AI-Enhanced Load Balancing in Federated Edge Computing Networks: Challenges and Solutions

Sourav Mondal, Goutam Das, and Elaine Wong

Abstract—In federated edge computing networks, edge computing (EC) nodes from multiple service providers (SPs) are installed to serve the same customer base and each SP intends to maximize their economic utilities. Therefore, to address research problems like EC node placement, job request allocation, and load balancing, we need to involve some economic aspects along with network engineering aspects. This implies that we need to propose novel system models and formulate problems in a significantly different way than the existing ones. Thus, we are motivated to discuss about the primary aspects of these new problem formulations and propose a novel economic and game-theoretic model for load balancing among federated EC nodes in this paper. In this game formulation, instead of focusing on latency minimization, under-loaded EC nodes intend to maximize their economic utilities by receiving any extra workload and incentives from their overloaded neighbors while satisfying the expected latency target. Furthermore, we design a centralized control mechanism, tailor-made for ultra-reliable and low-latency (uRLL) applications, for implementing this load balancing framework by incorporating artificial intelligence (AI)-enhanced traffic prediction algorithms.

I. INTRODUCTION

EDGE computing technologies like multi-access edge computing, fog computing, and cloudlet computing act like a bridge between remote cloud servers and mobile edge devices by providing accessibility to computational and memory resources with a very low latency [1]. These EC nodes are mainly placed across the access and metro network segments of the Internet and create large-scale peer-to-peer (P2P) distributed systems. The placement of EC nodes is driven by several key factors like the underlying access network technology and architecture, available network bandwidth, the density

of mobile devices and their demand for computational and memory resources [2]. In EC networks, the job requests originating from mobile devices are assigned to the available EC nodes, which is referred to as the job request allocation problem. Nonetheless, sometimes certain EC nodes may get overloaded if too many job requests get assigned to them. Then some of their job requests need to be offloaded to their under-loaded neighbors, which is known as the load balancing problem [3]. A load balancing framework among EC nodes can be implemented via a centralized protocol where an oracle node makes the load balancing decisions. Such frameworks can provide quick and efficient solutions but, cannot handle very large systems. An alternative approach is to implement P2P or decentralized frameworks, which are efficient in handling large systems and robust against EC node failures and network disruptions. Nonetheless, in spite of several advantages of P2P or decentralized frameworks, a major challenge in implementing such frameworks is the exchange of exhaustive control packets among EC nodes. These issues can be overcome by incorporating AI-based techniques in designing network management protocols [4].

We critically observe that most of the recent works that address the load balancing problem among EC nodes, do not consider the coexistence of EC nodes from multiple SPs. When neighboring EC nodes from same as well as different SPs coexist over the same network, we call this network setup as a *federated EC network*. Due to the lack of frameworks or software for controlling and federating resources among EC nodes from multiple SPs, the authors of [5] identify that there is a need for proposing new schemes for the federation of EC nodes. Especially, designing of new pricing models for the federation of resources among EC nodes and policies for resource sharing among EC nodes schemes appear to be highly essential. When a group of SPs are willing to cooperate among them-

S. Mondal and E. Wong are with the Department of Electrical and Electronic Engineering, The University of Melbourne, VIC 3010, Australia. (e-mail: smondal@student.unimelb.edu.au, ewon@unimelb.edu.au).

G. Das is with the G. S. Sanyal School of Telecommunications, Indian Institute of Technology Kharagpur, 721302, India. (e-mail: gdas@gssst.iitkgp.ac.in).

Fig. 1: Fundamental aspects of the load balancing problem among EC nodes in federated environments.

selves to maximize their utilities, then cooperative game-theoretic models can be proposed where a group of SPs form stable coalitions [6]. However, if the SPs are completely non-cooperative, we can propose load balancing mechanisms based on non-cooperative game-theoretic models [7].

In this article, we propose a novel economic and game-theoretic model for load balancing among federate EC nodes from different SPs. In this model, the EC nodes are interested in maximizing their economic utilities, rather than minimizing the overall latency. The primary motivation behind this approach is our realization that mobile users do not pay any extra price to the SPs for executing the job requests within the requested quality-of-service (QoS) latency. This implies that for job requests with a QoS latency target of 5 ms, user satisfaction is the same for an overall latency of 1 ms, 4 ms, or 5 ms. Therefore, any under-loaded EC node always finds it beneficial to receive some extra workload from its overloaded neighbors while satisfying the expected QoS latency, because the associated incentives help to maximize their individual utilities. Furthermore, we show that this model can be implemented through an AI-enhanced centralized framework, suitable for uRLL applications, where a neutral mediator supervises the market competition among EC nodes from different SPs. To satisfy the

stringent latency restrictions of uRLL applications, each EC node sends their predicted job request arrival rates to the mediator for making the load balancing decisions before the actual job request arrival and avoid the latency overheads for computing the load balancing strategies.

II. CHARACTERISTICS OF LOAD BALANCING PROBLEM AMONG FEDERATED EC NODES

In practice, the EC nodes are distributed over a large geographic area and the incoming job requests vary over time. Hence, load balancing among EC nodes is an important research challenge. In Fig. 1, we show the fundamental aspects of load balancing in federated EC networks. The very first important aspect is the *economic interaction among EC nodes from the same as well as different*, i.e., *heterogeneous SPs*. The economic interaction happens in terms of a few parameters like the payment received by the EC nodes from users, penalties paid by the EC nodes to the users for failing to meet QoS requirements and the mutual incentives exchanged for P2P computation offloading. In a federated environment, an overloaded EC node pays an additional incentive for offloading its extra workload to an under-loaded neighbor from a different SP. These payment parameters can be imposed by a neutral central authority like government or mediator. They

can also be dynamically determined through a suitable economic or bargaining policies.

The second aspect is *heterogeneous QoS requirements support*, because the job requests arriving to EC nodes may have different bandwidth demands for data transmission, CPU cycles for job processing, and maximum tolerance for end-to-end (E2E) latency. Such diverse requirements can be handled through *network slicing* in 5G and beyond networks. A very intriguing research challenge is the strategic partitioning of processors in EC nodes for P2P load balancing. The third aspect is *truthful information exchange among nodes*. Network parameters like job request arrival rates, job request service rates and QoS latency targets of the incoming job requests are private information for the EC nodes. Therefore, the EC nodes from different SPs may reveal false information to gain some profits from the market. Hence, to enforce truthful information revelation, we need to design incentive-compatible mechanisms for centralized frameworks or reinforcement learning based decentralized frameworks.

In recent times, we observe a growing interest to apply game theory in addressing network problems like load balancing as it provides a wide variety of mathematical models for strategic interaction among rational and independent agents [8]. Especially in federated EC networks, game-theoretic models appear to be highly essential as EC nodes from different SPs may cooperate or compete with each other. For example, the authors of [9] proposed a collaborative load balancing algorithm for latency improvement based on the Nash bargaining solution for a cooperative game among fog computing nodes.

On the other hand, the authors of [10] proposed a distributed non-cooperative load balancing game in small cell networks among the neighboring cloudlets who try to minimize their end-to-end latency subject to explicit energy and latency constraints. This model is very efficient under moderate load conditions, but under very high load conditions it performs poorly because some of the cloudlets may violate the latency constraints and the Nash equilibrium (NE) solution becomes infeasible. Moreover, to the best of our knowledge, most of the existing frameworks were not designed for federated EC networks, which we addressed in [11], [12]. In this game, an overloaded cloudlet paid incentives for offloading job requests to a cloudlet only from a different SP. Otherwise, incentives were not required

and the game-theoretic framework reduced to a more suitable distributed optimization framework.

III. SYSTEM MODEL

In this section, we discuss about the job request arrival and service processes at the set of N EC nodes $C = \{1, 2, \dots, N\}$ where $N \geq 2$. We also design an AI-enhanced control mechanism to implement this load balancing framework.

(a) Job Request Service Process: Usually each of the EC nodes contain a finite number of processors and we assume that job processing capabilities of each unit processor is the same. The average service rate of the EC node- i , $i \in C$ with a single processor is denoted by μ_i (jobs/s) and with n_i processors is denoted by $\mu_{ii} = n_i \mu_i$. We consider the processing model of the EC nodes as $M/M/1$ queuing systems based on the assumption that incoming job requests are maximally parallelized [10].

(b) Job Request Arrival Process: The average job request arrival rate to EC node- i is denoted by λ_i . Each EC node determines to process a major share of the incoming job requests internally and offload excess workload to a neighboring EC node, through some internal scheduling algorithm (beyond the scope of this paper). The average job request arrival varies over time and we assume that each λ_i is *independently and uniformly distributed* over the support $\Lambda_i = [0, \lambda_i^{max}]$, $\forall i \in C$. In practice, although the job request arrival process to EC nodes is a non-stationary process, but it displays some *pseudo-stationary* characteristics as the mean job request arrival rate varies gradually.

(c) QoS Latency Requirements: In this paper, rather than individual job requests, we are considering a batch of incoming job requests to EC nodes. Hence, the computational and latency requirements of all the incoming job requests to EC node- i are denoted by the consolidated tuple (μ_i, λ_i, D_Q) . We assume that all job requests belong to a similar type of low-latency applications and the value of QoS target latency D_Q is also the same. We consider a more generalized model with multiple classes of job requests with heterogeneous D_Q values as our future work. This implies that the duration of each timeslot is also equal to D_Q and if any highly overloaded EC node fails to process some of its incoming job requests within D_Q , it will drop those job requests and pay a penalty for that. Note that

Fig. 2: A schematic diagram among N federated EC nodes and the timing diagram illustrating the overall control design.

an $M/M/1$ queue provides only an upper bound for the processing latency of an EC node with the aggregated processing rate of all the processors.

(d) User Mobility Model: We assume that the mobile users do not leave the coverage area of an EC node within 1-10 ms. Hence, we can consider the *quasi-static mobility model* for mobile users. This implies that mobile users are almost stationary to the corresponding EC node during computation offloading period, but may move on later.

A. AI-enhanced Centralized Control Mechanism

In a centralized framework, the federated EC nodes are *non-cooperative* and *rational utility maximizers*, but the mediator is neutral and supervises the interactions among the federated EC nodes only to ensure a fair market competition, as shown in Fig. 2. The EC nodes predict their future job request arrival rates, denoted as $\hat{\lambda} = (\hat{\lambda}_1, \hat{\lambda}_2, \dots, \hat{\lambda}_N) \in \Lambda$, the average round-trip data transmission latency among mobile devices and the corresponding EC node- i , denoted by t_{ui} and the inter-EC node round-trip data transmission latency, denoted by $t_{ij}, \forall i, j \neq i \in C$. We assume that EC-nodes share this information *truthfully* as the mediator can impose an incentive-compatible mechanism [12]. We further assume that the incoming job request traffic is *bursty* in nature and has *short-term dependency* characteristics. Hence, we use the auto-regressive integrated moving average (ARIMA)-based traffic prediction algorithm [13]. Each EC node makes an initial prediction by observing the moving average over a finite set of historical data samples and sends their predicted information to the mediator two time-slots earlier. The accuracy of the average job request arrival

traffic can be improved further by observing the error iteratively in subsequent time-slots. Based on the revealed information, the mediator computes the NE load balancing strategies for each EC node. The fundamental stages of the overall control design are summarized below:

- Each EC node- i continuously observes a finite set of historical data samples and uses an ARIMA-based algorithm to predict the average incoming job request arrival rate $\hat{\lambda}_i^{(n+1)}$ for the $(n+1)^{\text{th}}$ time-slot at the $(n-1)^{\text{th}}$ time-slot.
- Each EC node also estimates the parameters t_{ui} and t_{ij} by using the given stochastic parameters of the wireless and optical interfaces between mobile devices and EC nodes.
- Each EC node communicates their latest predictions on $\hat{\lambda}_i^{(n+1)}$, t_{ui} and t_{ij} to the centralized computational facility installed by the mediator.
- The mediator employs a gradient projection algorithm to compute the NE computation offloading strategies and sends back to the EC nodes while n^{th} time-slot is ongoing.
- Thus, the overloaded EC nodes offload a fraction of their total incoming workload to their underloaded neighbors when the $(n+1)^{\text{th}}$ time-slot actually begins.
- During the $(n+1)^{\text{th}}$ time-slot, each EC node calculates the prediction error on the job request arrival rate to aid the learning algorithm further for improving the accuracy in next time-slots.

IV. ECONOMIC AND NON-COOPERATIVE LOAD BALANCING GAME AMONG EC NODES

In the federated EC node deployment scenario, the complete job request offloading strat-

egy space of all EC nodes is defined as a matrix $\Phi = (\Phi_1^T, \Phi_2^T, \dots, \Phi_N^T)^T \in \mathbb{R}^{N \times N}$, where $\varphi_i = (\varphi_{i1}, \varphi_{i2}, \dots, \varphi_{iN}) \in \Phi_i \subset \mathbb{R}^N$, $\varphi_{ij} \in \Phi_{ij} = [0, 1] \subset \mathbb{R}$, and $\sum_{j=1}^N \varphi_{ij} = 1, \forall i \in C$. Each φ_{ij} denotes the fraction of job requests EC node- i offloads to its neighboring EC node- j . Each EC node receives a linearly proportional price (Ω_1) per workload from the connected mobile devices. Each EC node pays a linearly proportional price per workload (Ω_2) for offloading job requests to a neighboring EC node from a different SP and also, receives a linearly proportional price for executing its neighbor's offloaded jobs. The EC nodes can also cooperate or bargain among themselves to decide the value of Ω_2 , which leads to a cooperative or bargaining game-theoretic model and is part of our future work. Each EC node- i has to pay this price to EC node- j only when it belongs to a different SP. In addition to these, if any EC node violates the QoS target latency D_Q , then it pays a linear penalty price with a proportionality cost factor (Ω_3) [10].

The objective of each federated EC nodes is to maximize their *individual utility* $\mathcal{U}_i^N(\varphi_i, \varphi_{-i}), \forall i \in C$, where $\varphi_{-i} = (\varphi_1, \dots, \varphi_{i-1}, \varphi_{i+1}, \dots, \varphi_N)$. We define the utility function as $\mathcal{U}_i^N(\varphi_i, \varphi_{-i}) = (\text{revenue earned from mobile users}) + \gamma_{ji} \times (\text{incentives earned, if received job requests from neighbors}) - \gamma_{ij} \times (\text{incentives paid, if offloaded job requests to neighbors}) - (\text{penalty, if QoS latency target is violated})$, where $\gamma_{ij} = 1$ if neighboring EC nodes belong to different SPs, else $\gamma_{ij} = 0$. Note that, we consider the utility function only under the *condition of stable operation*, i.e., $[\mu_{ii} - (1 - \sum_{j \neq i} \varphi_{ij}) \lambda_i - \sum_{j \neq i} \varphi_{ji} \lambda_j] >$

Fig. 3: Variation of utility function of EC node- i against φ_{ji} .

$0, \forall i, j \neq i \in C$ and a necessary condition $\Omega_2 \geq \Omega_3 \left[\max\{t_{ui}\} + \frac{1}{\max\{\mu_{ii}\}} + \max\{t_{ij}\} - D_Q \right], \forall i, j \neq i \in C$ is satisfied such that the feasibility of pure strategy Nash equilibrium (NE) solution is ensured [11]. Now, we can calculate each of the terms present in utility as follows:

- *revenue earned from mobile users* = $\Omega_1 \times (\text{total workload from associated mobile users})$,
- *incentives earned for receiving or paid for offloading job requests* = $\Omega_2 \times (\text{amount of workload received or offloaded})$,
- *penalty if QoS latency violated* = $\Omega_3 \times \max\{0, (\text{latency of the job requests} - D_Q)\}$.

A. Computation of Pure Strategy NE Solution

The value of $\mathcal{U}_i^N(\varphi_i, \varphi_{-i})$ is maximum when the end-to-end latency of the incoming job requests is equal to D_Q because the latency penalty is zero. Thus, any overloaded EC node intend to offload its extra workload to avoid the latency penalty and keep the utility at the maximal point. Similarly, any under-loaded EC node is interested in receiving extra workload and associated incentives without exceeding D_Q , to push its utility towards the maximal point, as shown in Fig. 3. To incorporate this property, we introduce an additional set of shared constraints $\varphi_{ij} \leq \psi_{ij}$, where φ_{ij} is decided by an overloaded EC node, but ψ_{ij} is decided by the corresponding under-loaded EC node. Therefore, we identify this game formulation as a *generalized Nash equilibrium problem*. We can label an EC node as overloaded if $\mathbb{T}_i = (t_{ui} + \frac{1}{\mu_{ii} - \lambda_i}) > D_Q$ and under-loaded if $\mathbb{T}_i = (t_{ui} + \frac{1}{\mu_{ii} - \lambda_i}) \leq D_Q$. Based on the load conditions in various scenarios, the utility and constraints of different EC nodes should be organized and the pure strategy NE can be computed through an optimization problem formulation as shown in [12]. Now, we explain the load balancing game through some basic examples with illustrations in Fig. 4. We consider a set of three EC nodes $C = \{1, 2, 3\}$, where EC node-1 and 2 belong to SP-A and EC node-3 belongs to SP-B with $\mu_{11} = \mu_{22} = \mu_{33} = 1000$ jobs/s, $D_Q = 10$ ms, $t_{u1} = t_{u2} = t_{u3} = 2$ ms, and $t_{12} = t_{21} = 0.5$ ms, $t_{23} = t_{32} = 0.7$ ms, and $t_{31} = t_{13} = 0.9$ ms. In addition, we arbitrarily choose $\Omega_1 = 5 \times 10^4$, $\Omega_2 = 3 \times 10^4$, and $\Omega_3 = 9 \times 10^4$, such that the necessary game design condition is satisfied.

(a) All EC nodes are under-loaded: In the first scenario, we consider the job request arrival rates

Fig. 4: Pure Nash equilibrium strategies of federated EC nodes in the load balancing game under different scenarios.

are $\lambda_1 = 750$ jobs/s, $\lambda_2 = 580$ jobs/s and $\lambda_3 = 695$ jobs/s. Therefore, the end-to-end latencies of job requests arriving at EC node-1 is $\mathbb{T}_1 = 6$ ms < 10 ms, at EC node-2 is $\mathbb{T}_2 = 4.38$ ms < 10 ms, and at EC node-3 is $\mathbb{T}_3 = 5.28$ ms < 10 ms. Clearly, all EC nodes are under-loaded, i.e., each EC node can process and satisfy the QoS target latencies of their individual job requests. Hence, the EC nodes do not offload any job requests to each other and the NE solution of our proposed economic and non-cooperative load balancing game is $\varphi_{12}^* = \varphi_{21}^* = \varphi_{13}^* = \varphi_{31}^* = \varphi_{23}^* = \varphi_{32}^* = 0$. Consequently, the economic utilities of the EC nodes are $\mathcal{U}_1^3 = \Omega_1 \frac{\lambda_1}{\mu_{11}} = 30000$ cost units, $\mathcal{U}_2^3 = \Omega_1 \frac{\lambda_2}{\mu_{22}} = 47500$ cost units, and $\mathcal{U}_3^3 = \Omega_1 \frac{\lambda_3}{\mu_{33}} = 46000$ cost units, because the incentive and penalty terms are absent from the utility calculation in this case.

(b) One under-loaded and two overloaded EC nodes: In this scenario, we consider $\lambda_1 = 600$ jobs/s, $\lambda_2 = 950$ jobs/s and $\lambda_3 = 920$ jobs/s with $D_Q = 10$ ms. Therefore, the end-to-end latencies of job requests arriving at EC node-1 is $\mathbb{T}_1 = 4.5$ ms < 10 ms, at EC node-2 is $\mathbb{T}_2 = 22$ ms > 10 ms, and at EC node-3 is $\mathbb{T}_3 = 14.5$ ms > 10 ms. This implies that EC node-1 is under-loaded but EC node-2 and 3 are overloaded. Therefore, to avoid the penalty for violating QoS latency target, both EC node-2 and 3 offload their excess workloads to EC node-1, as shown in Fig. 4a. The pure strategy NE solution is $\varphi_{12}^* = \varphi_{13}^* = \varphi_{23}^* = \varphi_{32}^* = 0$, $\varphi_{21}^* = 0.0789$, $\varphi_{31}^* = 0.0489$, i.e., EC node-2 offloads job requests at a rate

$\varphi_{21}^* \lambda_2 = 0.0789 \times 950 = 1349.64$ jobs/s to EC node-1 but do not pay any incentives as they belong to the same SP. On the contrary, EC node-3 offloads job requests at a rate $\varphi_{31}^* \lambda_3 = 0.0489 \times 920 = 45.002$ jobs/s and pays an incentive $\Omega_2 \varphi_{31}^* \frac{\lambda_3}{\mu_{11}} = 35.991$ cost units to EC node-1 as they belong different SPs. Now, the modified end-to-end latencies at the EC nodes are $\leq D_Q$. Consequently, the economic utilities of the EC nodes are $\mathcal{U}_1^3 = \Omega_1 \frac{\lambda_1}{\mu_{11}} + \Omega_2 \varphi_{31}^* \frac{\lambda_3}{\mu_{11}} = 31349.64$ cost units, $\mathcal{U}_2^3 = \Omega_1 \frac{\lambda_2}{\mu_{22}} = 47500$ cost units, and $\mathcal{U}_3^3 = \Omega_1 \frac{\lambda_3}{\mu_{33}} - \Omega_2 \varphi_{31}^* \frac{\lambda_3}{\mu_{11}} = 44650.36$ cost units, because only revenue and incentive terms are present. Note that, if EC node-1 was loaded slightly more, say $\lambda_1 = 800$ jobs/s, then it would not be able to accept the entire load ($\varphi_{21}^* \lambda_2 + \varphi_{31}^* \lambda_3$) while satisfying D_Q . Thus, the NE solution is $\varphi_{21}^* = 0.0623$, $\varphi_{31}^* = 0$ as it prioritizes the EC node-2 from the same SP. This happens because our game acts an optimization problem among EC nodes from the same SP.

- If an under-loaded EC node has sufficient capacity, it accepts the entire extra workload from its all overloaded neighbors.
- If an under-loaded EC node does not have sufficient capacity, it accepts the maximum possible workload only from its neighboring EC nodes with the same SP.

(c) Two under-loaded and one overloaded EC nodes: This is an alternative scenario to the previous one where we consider $\lambda_1 = 970$ jobs/s, $\lambda_2 = 820$ jobs/s and $\lambda_3 = 700$ jobs/s. Therefore, the end-to-end latencies of job requests arriving at EC node-1

is $\mathbb{T}_1 = 35.33 \text{ ms} > 10 \text{ ms}$, at EC node-2 is $\mathbb{T}_2 = 7.55 \text{ ms} < 10 \text{ ms}$, and at EC node-3 is $\mathbb{T}_3 = 5.33 \text{ ms} < 10 \text{ ms}$. This implies that EC node-1 is overloaded but EC node-2 and 3 are under-loaded. Thus, EC node-1 offloads its excess workloads to EC node-2 and 3, as shown in Fig. 4b. The pure strategy NE solution is $\varphi_{21}^* = \varphi_{23}^* = \varphi_{31}^* = \varphi_{32}^* = 0$, $\varphi_{12}^* = 0.0235$, $\varphi_{13}^* = 0.0745$, i.e., EC node-2 receives job requests at a rate $\varphi_{12}^* \lambda_1 = 0.0235 \times 970 = 22.71 \text{ jobs/s}$ from EC node-1 but do not receive any incentives as they belong to the same SP. On the contrary, EC node-3 receives job requests at a rate $\varphi_{13}^* \lambda_1 = 0.0745 \times 970 = 72.29 \text{ jobs/s}$ and receives an incentive $\Omega_2 \varphi_{13}^* \frac{\lambda_1}{\mu_{33}} = 2167.95$ cost units from EC node-1 as they belong different SPs. Now, the modified end-to-end latencies at the EC nodes are $\leq D_Q$. Therefore, the economic utilities of the EC nodes are $\mathcal{U}_1^3 = \Omega_1 \frac{\lambda_1}{\mu_{11}} - \Omega_2 \varphi_{31}^* \frac{\lambda_3}{\mu_{11}} = 46332.05$ cost units, $\mathcal{U}_2^3 = \Omega_1 \frac{\lambda_2}{\mu_{22}} = 41000$ cost units, and $\mathcal{U}_3^3 = \Omega_1 \frac{\lambda_3}{\mu_{33}} + \Omega_2 \varphi_{31}^* \frac{\lambda_3}{\mu_{11}} = 37167.95$ cost units.

Note that the share of workload distributed between EC node-2 and 3 is dependent on their respective workloads and EC node-2 alone cannot process all the extra workload from EC node-1. If we had, say $\lambda_1 = 970 \text{ jobs/s}$, $\lambda_2 = 750 \text{ jobs/s}$, and $\lambda_3 = 700 \text{ jobs/s}$, then the NE solution would be $\varphi_{12}^* = 0.0979$, $\varphi_{13}^* = 0$ as this helps EC node-1 to avoid paying extra incentives. Again, if EC node-2 and 3 could not accept the total workload ($\varphi_{12}^* + \varphi_{13}^*) \lambda_1$, say $\lambda_1 = 970 \text{ jobs/s}$, $\lambda_2 = 850 \text{ jobs/s}$ and $\lambda_3 = 840 \text{ jobs/s}$, then they had to partially accept the job requests such that D_Q is just satisfied. Then the NE solution would be $\varphi_{12}^* = 0.0172$, $\varphi_{13}^* = 0.0198$.

- If an overloaded EC node has two under-loaded neighbors from the same and different SPs who can receive the extra workload, then it prefers to offload to the same SP.
- An overloaded EC node offloads the maximum possible workload to its neighbor from the same SP and then to different SPs.

(d) All EC nodes are overloaded: In this scenario, we consider $\lambda_1 = 910 \text{ jobs/s}$, $\lambda_2 = 978 \text{ jobs/s}$ and $\lambda_3 = 953 \text{ jobs/s}$. Therefore, the end-to-end latencies of job requests arriving at EC node-1 is $\mathbb{T}_1 = 13.11 \text{ ms} > 10 \text{ ms}$, at EC node-2 is $\mathbb{T}_2 = 47.45 \text{ ms} > 10 \text{ ms}$, and at EC node-3 is $\mathbb{T}_3 = 23.27 \text{ ms} > 10 \text{ ms}$. Clearly, all EC nodes are overloaded and fails to satisfy the QoS target latencies of their individual job requests and they cannot improve the latency performance by mutual job

Fig. 5: Comparison of NE utility values of federated EC nodes with actual and predicted in incoming job request arrival rates.

request offloading. The NE solution of our proposed economic and non-cooperative load balancing game is $\varphi_{12}^* = \varphi_{21}^* = \varphi_{13}^* = \varphi_{31}^* = \varphi_{23}^* = \varphi_{32}^* = 0$. Hence, the corresponding economic utilities of the EC nodes are $\mathcal{U}_1^3 = \Omega_1 \frac{\lambda_1}{\mu_{11}} - \Omega_3 \frac{\lambda_1}{\mu_{11}} \{t_{u1} + \frac{1}{\mu_{11} - \lambda_1}\} = 44426.2$ cost units, $\mathcal{U}_2^3 = \Omega_1 \frac{\lambda_2}{\mu_{22}} - \Omega_3 \frac{\lambda_2}{\mu_{22}} \{t_{u2} + \frac{1}{\mu_{22} - \lambda_2}\} = 44723.05$ cost units, and $\mathcal{U}_3^3 = \Omega_1 \frac{\lambda_3}{\mu_{33}} - \Omega_3 \frac{\lambda_3}{\mu_{33}} \{t_{u3} + \frac{1}{\mu_{33} - \lambda_3}\} = 45653.57$ cost units, respectively.

Clearly, the above numerical examples help us to understand the fundamental insights on load balancing strategies among non-cooperative EC nodes in a federated environment. We could highlight the fact that the EC nodes prefer to receive or offload job requests from or to EC nodes with the same SP. Different combinations of the aforementioned characteristics are applicable against all possible scenarios with $N \geq 3$ EC nodes.

B. AI-enhanced Framework Evaluation

In Fig. 5, we show the impact of the prediction accuracy of the incoming job request arrival rates on the NE utility values of the federated EC nodes. For the job request arrival rate prediction, we use the moving-average method based ARIMA algorithm and consider that job request arrival rates to each EC node remains stationary for 30 seconds. We consider the same three EC nodes in Sec. IV-A with $\mu_{ii} = 1000 \text{ jobs/s}$ and λ_i varying within 0-1000 jobs/s. Therefore, from the plot, we observe that whenever the actual job request arrival rates of the EC nodes change, the NE utility values of the EC

nodes based on the predicted job request arrival rate are slightly erroneous. However, within a few time-slots, each EC node can accurately predict the actual job request arrival rate. Therefore, the NE utilities of the EC nodes with predicted job request arrival rates match to the NE utilities with actual job request arrival rates.

V. CONCLUSIONS AND FUTURE RESEARCH

In this paper, we have proposed a novel economic and game-theoretic model for load balancing among non-cooperative federated EC nodes. We have implemented this model as a centralized framework with AI-enhanced traffic prediction methods for uRLL applications. We strongly believe that this work will encourage more future research in several related directions, e.g., truthful incentive-compatible mechanism design, implementation of decentralized load balancing frameworks, computation of load balancing strategies by using reinforcement learning, to name a few.

ACKNOWLEDGMENT

The authors would like to thank *Melbourne School of Engineering (MSE)* for providing financial support to S. Mondal through *PhD Write Up Award*.

REFERENCES

- [1] K. Dolui and S. K. Datta, "Comparison of Edge Computing Implementations: Fog computing, Cloudlet and Mobile Edge Computing," in *2017 Global Internet of Things Summit (GloTS)*, June 2017, pp. 1–6.
- [2] S. Mondal, G. Das, and E. Wong, "Efficient Cost-Optimization Frameworks for Hybrid Cloudlet Placement over Fiber-Wireless Networks," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 11, no. 8, pp. 437–451, August 2019.
- [3] Y. Jiang, "A Survey of Task Allocation and Load Balancing in Distributed Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 585–599, Feb 2016.
- [4] D. Rafique and L. Velasco, "Machine Learning for Network Automation: Overview, Architecture, and Applications [Invited Tutorial]," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 10, no. 10, pp. D126–D143, 2018.
- [5] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, "All One Needs to Know About Fog Computing and Related Edge Computing Paradigms: A Complete Survey," *Journal of Systems Architecture*, 2019.
- [6] C. Anglano, M. Canonico, P. Castagno, M. Guazzone, and M. Sereno, "A Game-Theoretic Approach to Coalition Formation in Fog Provider Federations," in *2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)*, 2018, pp. 123–130.
- [7] S. Mondal, G. Das, and E. Wong, "Computation Offloading in Optical Access Cloudlet Networks: A Game-Theoretic Approach," *IEEE Communications Letters*, vol. 21, no. 7, pp. 1481–1484, July 2018.
- [8] S. Penmatsa and A. T. Chronopoulos, "Game-theoretic Static Load Balancing for Distributed Systems," *Journal of Parallel and Distributed Computing*, vol. 71, no. 4, pp. 537 – 555, 2011.
- [9] H. Xiao, Z. Zhang, and Z. Zhou, "GWSA Collaborative Load-Balancing Algorithm for Internet-of-Things," *Sensors*, vol. 18, no. 8, p. 2479, 2018.
- [10] L. Chen, S. Zhou, and J. Xu, "Computation Peer Offloading for Energy-Constrained Mobile Edge Computing in Small-Cell Networks," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1619–1632, Aug 2018.
- [11] S. Mondal, G. Das, and E. Wong, "A Game-Theoretic Approach for Non-Cooperative Load Balancing Among Competing Cloudlets," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 226–241, 2020.
- [12] S. Mondal, G. Das, and E. Wong, "Centralized and Decentralized Non-Cooperative Load-Balancing Games among Competing Cloudlets," *CoRR*, vol. abs/2006.00390, 2020. [Online]. Available: <https://arxiv.org/abs/2006.00390>
- [13] L. Ruan, M. P. I. Dias, and E. Wong, "Enhancing Latency Performance through Intelligent Bandwidth Allocation Decisions: A Survey and Comparative Study of Machine Learning Techniques," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 12, no. 4, pp. B20–B32, 2020.

Sourav Mondal received the B.Tech from Kalyani Govt. Engineering College and the M.Tech from the Indian Institute of Technology Kharagpur in 2012 and 2014, respectively. Currently, he has joined the University of Melbourne to pursue his doctoral studies and is working on Strategic Deployment of Artificial Intelligence-Enhanced Cloudlets for Low-latency Human-to-Machine Applications.

Goutam Das has obtained his M.Tech. degree from the Indian Institute of Technology, Kharagpur (India) in 2001, and his Ph.D. degree from the University of Melbourne (Australia) in 2008. Currently he is working as an Assistant Professor in the Indian Institute of Technology, Kharagpur. His research interest is in the area of both optical as well as wireless networking.

Elaine Wong received B.E. and Ph.D. degrees from the University of Melbourne, Melbourne, Victoria, Australia. She is currently a Professor at the University of Melbourne. Her research interests include optical-wireless integration, broadband applications of vertical-cavity surface-emitting lasers, wireless sensor body area networks, and emerging optical and wireless technologies for Tactile Internet.