Potential Game Based Distributed IoV Service Offloading With Graph Attention Networks in Mobile Edge Computing

Qinting Jiang, Xiaolong Xu, *Senior Member, IEEE*, Muhammad Bilal, *Senior Member, IEEE*, Jon Crowcroft, *Fellow, IEEE*, Qi Liu, *Senior Member, IEEE*, Wanchun Dou, and Jingyan Jiang

***Abstract*— Vehicular services aim to provide smart and timely services (e.g., collision warning) by taking the advantage of recent advances in artificial intelligence and employing task offloading techniques in mobile edge computing. In practice, the volume of vehicles in the Internet of Vehicles (IoV) often surges at a single location and renders the edge servers (ESs) severely overloaded, resulting in a very high delay in delivering the services. Therefore, it is of practical importance and urgency to coordinate the resources of ESs with bandwidth allocation for mitigating the occurrence of a spike traffic flow. For this challenge, existing work sought the periodicities of traffic flow by analyzing historical traffic data. However, the changes in traffic flow caused by sudden traffic conditions cannot be obtained from these periodicities. In this paper, we propose a distributed traffic flow forecasting and task offloading approach named TFFTO to optimize the execution time and power consumption in service processing. Specifically, graph attention networks (GATs) are leveraged to forecast future traffic flow in short-term and the traffic volume is utilized to estimate the number of services offloaded to the ESs in the subsequent period. With the estimate, the current load of the ESs is adjusted to ensure that the services can be handled in a timely manner. Potential game theory is adopted to determine the optimal service offloading strategy. Extensive experiments are conducted to evaluate our approach and the results validate our robust performance.**

***Index Terms*— Service offloading, edge computing, graph attention network, game theory, flow forecasting.**

Manuscript received 26 March 2023; revised 6 November 2023; accepted 9 February 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 92267104 and Grant 62372242 and in part by the Natural Science Foundation of Jiangsu Province of China under Grant BK20211284. The Associate Editor for this article was K. Cengiz. *(Corresponding authors: Xiaolong Xu; Jingyan Jiang.)*

Qinting Jiang is with the School of Software, Nanjing University of Information Science and Technology, Nanjing 210044, China, and also with the SIGS, Tsinghua University, Shenzhen 518055, China (e-mail: q.jiang@nuist.edu.cn).

Xiaolong Xu is with the School of Software, Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: xlxu@ieee.org).

Muhammad Bilal is with the School of Computing and Communications, Lancaster University, Bailrigg, LA1 4WA Lancaster, U.K. (e-mail: m.bilal@ieee.org).

Jon Crowcroft is with the Department of Computer Science and Technology, University of Cambridge, CB3 0FD Cambridge, U.K. (e-mail: jon.crowcroft@cl.cam.ac.uk).

Qi Liu is with the School of Computer Science, Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: qi.liu@nuist.edu.cn).

Wanchun Dou is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China (e-mail: douwc@nju.edu.cn).

Jingyan Jiang is with the College of Big Data and Internet, Shenzhen Technology University, Shenzhen 518118, China (e-mail: jiangjingyan@sztu.edu.cn).

Digital Object Identifier 10.1109/TITS.2024.3369190

I. Introduction

IN MODERN metropolis, due to the high density of population and the increase in car ownership, urban traffic problems (e.g., traffic block) are becoming increasingly prominent. With the increasing development of wireless communication as well as artificial intelligence, Internet of vehicles (IoV) are capable of providing innovative services such as automatic driving and collision warning, thus alleviating the current traffic pressure, which significantly improves users' travel experience. Nevertheless, the limited computation resources make the vehicles fail to satisfy the high requirements for real-time service processing, posing a series of traffic security risks [1]. In addition, in the case of increasingly expensive fuel resources, the energy consumption generated by vehicular equipments will also increase the additional cost of users, resulting in the users' economic burden.

To tackle the contradiction of the resource-limited vehicles and the high requirements for real-time monitoring, a promising approach is to offload the vehicular services to the remote cloud [2]. By leveraging mighty computation power of the cloud, the computing capacity of the vehicles is extended. Thus the service processing speed is optimized to some extent [3]. Nevertheless, due to the extreme delay sensitiveness of the vehicular services, the unacceptable drawbacks of high latency and unstable connection between the vehicular users and the cloud center cannot be dismissed [4].

Mobile edge computing (MEC) has been widely practiced recently as the complement to cloud computing to address the issues of high latency and the unstable connection [5], [6]. The MEC provides approximative cloud computation power with the superiority of proximity to the users [7]. With lightweight data transmission, MEC can cut down the offloading delay significantly [8], [9]. Currently, a wide range of investigations on offloading decisions and resource provisioning in MEC has been studied, aiming at improving the quality of experience (QoE) of the end-users [10], [11]. Benefiting from MEC, the vehicle services are able to acquire the resources for execution in real time. Nevertheless, facing the increasing volumes of the vehicles and the high demands for real-time monitoring

and data analysis, the resource-constrained edge servers (ESs) fail to execute all the services of IoV simultaneously [12], [13]. As the supplementary of the ESs, centralized cloud provides these services with strict cloud processing capability. Furthermore, the connection reliability and the transmission latency have been significantly improved by the common usage of passive optical networks (PONs) [14]. Hence, it is particularly vital to take full advantage of the resources on the centralized cloud. The reasonable combination of MEC and cloud computing largely enhances users' experience and the QoS of the auxiliary services in driving [15].

Under the collaborative cloud-edge architecture, the users ought to select offloading destinations among the local device, ES and cloud server (CS). However, considering the mobility of the vehicles as well as the fixation of the ESs, interruptions caused by transitions of wireless connections occur in service offloading with great probability, which induces the high transmission delay. Moreover, the paroxysmal explosion of the traffic flow at a certain time point may cause ESs to be overloaded, making the services suffer from extremely high latency [16].

Currently, several methods have been studied to address the channel allocation and offloading decision problems in the service offloading process, aiming to optimize the overall transmission delay through different offloading schemes. Fan et al. [17] proposed a method that minimizes the total task processing delay for all vehicles by considering service scheduling, channel allocation, and computation resource allocation for vehicles and RSUs. On the other hand, other methods have focused on the resource allocation problem on edge servers, reducing the overall computation delay by allocating computing resources to users at a fine granularity. Tuyen et al. [18] employed convex optimization techniques to optimize the resource allocation in the service offloading process, thereby maximizing the offloading benefits for users. Existing methods for service offloading and resource allocation mainly address the optimization of service performance in quasi-static scenarios, where the total number of users and their distribution remain stable over a period of time. However, in reality, users exhibit strong mobility, and the number of users within the coverage area of different edge services changes in real-time. Furthermore, the distribution of users has temporal correlation. For example, during the rush hour, the total number of users at transportation hubs is much larger than in remote areas. Therefore, in scenarios where the user distribution dynamically changes, the load on edge servers becomes severely imbalanced. Idle servers at a particular moment result in resource wastage, while overloaded servers lead to excessive user waiting time, thereby affecting the overall service quality. Hence, to optimize the execution time (ET) and transmission time of each service, a reasonable service offloading strategy is required. Meanwhile, the edge server should maintain dynamic load balancing when facing the distribution shift of traffic flow [19].

With these observations, a distributed traffic flow forecasting and task offloading approach named TFFTO is designed to optimize the execution time and power consumption in service processing. In TFFTO, we combine potential game and graph attention networks to optimize the service offloading process. Potential game allows participants to optimize their strategies based on the actions of their opponents during the game. Such flexibility enables participants to better cope with uncertainty and environmental changes, thereby achieving improved results when the distribution of user quantities changes in real-time. Graph attention networks have the ability to integrate node information and possess a transductive property, enabling accurate prediction of large-scale traffic flows. The prime contributions of the paper are as follows.

- Construct a flow driven distributed computation offloading framework with collaborative cloud-edge computing in IoV for a dynamic service offloading scenario where the distribution of user quantities changes in real-time.
- Adopt the graph attention network (GAT) to improve the accuracy of traffic flow prediction. The prediction results are used to adjust the current load of ES, which significantly reduces the average service processing time.
- Design a potential game theory based distributed computation offloading algorithm to minimize the energy consumption of the vehicles and the service processing time.
- Conduct comparative experiments and the convergence analysis to demonstrate the validity of the proposed algorithm.

The remainder of this paper is organized as follows. In Section II, the recent researches related to our work are introduced. The system model and the optimization problem are presented in Section III. GAT is adopted for traffic flow prediction in Section IV, followed by the game formulation in Section V. The performance of the method is analyzed and the numerical results are provided in Section VI. Finally, Section VII summarizes this paper.

II. Related Work

The birth of IoV has promoted the development of a series of vehicular services such as collision warning and driverless driving [20]. These services not only provide great convenience for people's travel, but also need to consume vast computing resources to ensure low latency. The computing resources equipped with vehicles often fail to meet the delay requirements of vehicular services. Therefore, more powerful computing equipment is needed to support the operation of vehicular services. Since edge computing perfectly meets the needs of users in IoV for low-latency services, the applications of edge computing in IoV have been attracted great attention and extensively researched by scholars in recent years. As the storage and computing resources of edge servers are also limited, channels and computing resources must be allocated properly. otherwise, edge servers are likely to be overloaded [21].

A. Task Offloading and Resource Allocation in Quasi-Static Scenarios

Task offloading algorithm is a hot research topic in edge computing. By optimizing task offloading schemes, computational resources are allocated more effectively, leading

to a significant improvement in the resource utilization of edge servers and enhancing the overall user service experience. Currently, the optimization of task offloading strategies primarily depends on factors such as channel occupancy, server load status, and the maximum acceptable latency of the service itself. Some authors also perform fine-grained scheduling of computational resources on servers while task offloading, thereby further optimizing resource allocation. Feng et al. [22] proposed a distributed task offloading and data caching method to reduce the service latency and improve the storage utilization of edge servers. This method greatly reduces the network overhead by using dynamic programming. Tang and Wong [23] proposed a distributed task offloading algorithm based on the deep Q-network. The algorithm applied the model-free method, which ensured the users to make task offloading decisions without other users' information. However, offloading the whole task will lead to the waste of local computing resources, so a reasonable model segmentation method is needed to make the task more reasonably distributed. Gao et al. [24] proposed a model segmentation and task offloading scheme based on deep neural network. Through task partitioning, each sub-task can be processed on different devices, which enhances the flexibility of task scheduling and the utilization of computing resources.

B. The Application of Game Theory in Edge Computing

In task offloading, the relationship between users and servers can be one-to-many, many-to-one, or many-to-many. Additionally, each user has the autonomy to make offloading decisions independently or be centrally scheduled by a central server. Therefore, optimizing task offloading decisions becomes an exponentially complex problem, which also exhibits game-like characteristics. As a result, many studies employ game theory to address this problem. Mitsis et al. [25] adopted Stackelberg game and established a multi-leader multi-follower model between servers and users to determine the optimal pricing strategy for servers and the optimal data offloading strategy for users. Teng et al. [26] employed non-cooperative game theory and combined it with a greedy approach to address the time complexity issue in the allocation and scheduling of multiple tasks to multiple servers. Chen et al. [27] defined the problem as a multi-user unloading decision game, and proposed a game-based decentralized task unloading method to maximize user QoE under resource constraints. In general, game theory can effectively reduce time complexity and accelerate decision convergence when dealing with large-scale decision-making problems

C. Task Offloading Under Real-Time Traffic Flow Variations

Optimizing the task offloading algorithm alone cannot completely improve the utilization of the computing resources on edge servers. The unbalanced temporal and spatial distribution of vehicle-mounted users leads to the load disproportion of edge servers, which are prone to overload in traffic rush hours and empty in flat peak hours. To rationally utilize the computing resources of each edge server, it is necessary to obtain the traffic distribution of each place before task offloading. Fang et al. [28] proposed a fine-grained task offloading method based on traffic flow prediction. In this method deep spatiotemporal residual network is leveraged to estimate the traffic volume in each region. With the periodic results based on traffic flow forecasts, genetic algorithm is used to select a reasonable task offloading strategy. Chen et al. [29] proposed a hybrid traffic flow forecast method by sparse auto-encoder to address the over-fitting and manual intervention problems of traffic flow forecast. By feature engineering, this method makes the periodic prediction of traffic flow more accurate and provides an effective reference for the placement of edge servers.

However, most studies to our knowledge do not take into account the scale of all edge servers and users in the IoV. These studies only consider the periodic flow prediction and load balancing of a single node. In fact, the number distribution of users is often instantaneous, and has both temporal and spatial correlation. Therefore, accurate traffic prediction should combine the feature information and location information of multiple nodes to make a global judgment. Moreover, the traffic flow prediction of the above studies only focuses on the periodic results and does not consider the sudden conditions such as traffic surge, so it is unable to deal with the scene of real-time traffic flow change. When there is a large demand for user services or a large number of edge servers, it is easy to cause high dimensions of state space and action space, resulting in the insufferable training time of the algorithm model. In order to solve the above problems, we propose a short-term traffic flow prediction scheme based on GAT to adjust the load of edge servers in time. Additionally, we model the service offloading process as a potential game to avoid the dimension explosion caused by the large decision space and state space of users.

TABLE I
NOTATIONS AND DESCRIPTION

Symbol	Metric	Description
y	$\times$	The RSU index $y \in M$
w	$\times$	The vehicle index $w \in W$
Y	$\times$	Set of the RSUs
W	$\times$	Set of the vehicles
R	m	The wireless signal coverage of the RSU
v_w	m/s	The mean velocity of the vehicle w
k_w	W	Transmission power of the vehicle w
β_w	$\times$	The coefficient of the vehicle w
q	Hz	The channel bandwidth
δ	W	The background noise power
τ	$\times$	The offloading decision set
$r_{w,y}$	bps	The upload data rate of the vehicle w

III. SYSTEM MODEL

In this section, we define the system model of this paper, including communication model and computation model.

A. The Framework of Task Offloading

As shown in Fig. 1, a framework composed of Y roadside units (RSUs), a base station (BS) equipped with the computation power P_B (total CPU revolutions in a second), cloud

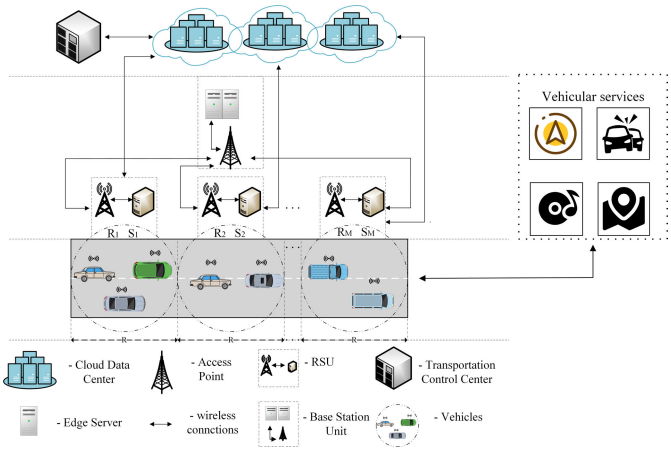

Fig. 1. A framework of collaborative services offloading in mobile edge computing.

computing servers and W vehicles are considered. The RSUs which are uniformly located along the highway possess the identical wireless signal coverage range R and the computation power P_R. Hence, the road can be divided into Y segments with all the vehicles randomly distributed in the researched highway. Additionally, the RSUs are connected with the BS and CSs by means of fiber optic cables. The vehicles either select to process the services independently or offload the services to the ESs/CSs. To obtain the useful insights and ensure tractable analysis, a quasi-static scenario is applied in which the speed of each vehicle in set $W = \{1, 2, 3, \ldots, S\}$ keeps invariable during the period of data transmission (e.g., several milliseconds). In addition, Code Division Multiple Access (CDMA) is adopted in this paper for communication and data transmission between devices. CDMA is a technology that encodes data using different spreading codes, allowing multiple users or devices to communicate simultaneously on the same frequency band. CDMA technology separates the data of specific users or devices from the interference of other users or devices by using the corresponding spreading code at the receiving end. The models of communication and computing in MEC are introduced respectively in the following parts. The meaning of some major symbols is listed in Table I.

B. Communication Model

The vehicle can either select the local computing, or choose the ES (e.g., RSU and BS)/the CS to execute the service. The computation offloading strategy of each vehicle is denoted as $d_w \in \{d_{w,W}, d_{w,R}, d_{w,B}, d_{w,C}\}$ where $w \in W$.

$$d_{i,j} \in \begin{cases} \{0,1\}, & \text{if } j \in \{W, B, C\}, \\ \{0,1,2,\ldots,Y\}, & \text{otherwise}, \end{cases} \tag{1}$$

where $\tau = \{W, B, C\}$ denotes the offloading decision set, which is composed of the vehicles (W), the BS (B) and the CS (C). Specifically, $d_{i,j} > 0$ if the vehicle w chooses to offload the services to $j \in \{R, B, C\}$ and $d_{w,W} > 0$ if the vehicle w selects the local processsing method. For the vehicle w, there is only one parameter greater than zero among the decision set $\{d_{w,W}, d_{w,R}, d_{w,B}, d_{w,C}\}$, others are all equal to zero, which means the service cannot be split and ought to be processed on one equipment. Provided the decision set $d = \{d_1, d_2, \ldots, d_W\}$ of all the vehicles, the uplink data rate of the vehicle w which unloads the service to the RSU, the BS or the CS via a wireless channel can be formulated as

$$\Psi_{w,y}(d) = qlog_2\left(1 + \frac{\mho}{\delta + \aleph}\right), \tag{2}$$

where $\mho = k_w D_w^y$ and $\aleph = \sum_{i!=w} \Xi(i \in W)\Xi(d_{i,j} = d_{w,j})k_i D_w^y$. $\Xi(h)$ is a judgement function. If h is true, $\Xi(h) = 1$. Otherwise, $\Xi(h) = 0$. Additionally, q is the channel bandwidth and k_w is the transmission power of the vehicle w. Moreover, D_w^y expresses the channel gain between the vehicle w and the edge device y, and δ_y represents the background noise power. In this work, the computation offloading is investigated with wireless interference, where the average summation throughput of users in the cellular communication scenario can be well captured.

From the communication model above, it is shown that if excessive vehicles select services offloading through the identical wireless channel concurrently, the transmitting procedure tends to suffer from severe interference which induces low data transmission rates.

C. Computation Model

Each vehicle has a computation service denoted as $\varphi_w = \{Q_w, G_w\}$, where Q_w refers to the input data size and G_w is the necessary computation resources (CPU revolutions in all) of the accomplishment of the service φ_w. Afterward, the system-wide overhead of the vehicle with a single task in terms of consumed energy and executed time under various computing models will be discussed.

1) Service Execution at Local Equipment: For the local execution model, the service φ_w of the vehicle w is executed on the local equipment. The computation power (total CPU revolutions in a second) of the vehicle w is denoted as p_w^L. It is supposed that various vehicles possess distinct computation powers. The time executing the service φ_w is expressed as

$$\sigma_{w,W}(d) = G_w \times \frac{1}{p_w^L}. \tag{3}$$

The consumed energy during the service processing is formulated as

$$\Omega_{w,W}(d) = \beta_w G_w, \tag{4}$$

where β_w is the coefficient of the vehicle w denoting the consumed energy per one CPU revolution. On the basis of (3) and (4), the overhead induced by computing locally in terms of consumed energy and executed time can be computed as

$$x_{w,W} = \lambda_w^t \sigma_{w,W}(d) + \lambda_w^e \Omega_{w,W}(d), \tag{5}$$

where λ_w^e and $\lambda_w^t \in \{0, 1\}$ express the weight coefficients of consumed energy and executed time for the decision making of vehicle w. When the battery is at a low state, the vehicle set $\lambda_w^e = 1$ and $\lambda_w^t = 0$. Similarly, when the service is sensitive to delay, the vehicle set $\lambda_w^t = 1$ and $\lambda_w^e = 0$ in the process of decision making. In other scenarios, proper

weighting parameters of the vehicle are resolved by employing the approach of multi-attribute utility.

2) Service Execution at RSUs: The computation task φ_w of the vehicle w is offloaded to the nearest RSU y. Additional overhead in the aspects of consumed energy and time is produced by conveying the input data to the RSU. On account of the communication model, the time caused by conveying data and the consumed energy of the vehicle w are formulated as

$$\varrho_{w,y}(d) = Q_w \times \frac{1}{\Psi_{w,y}(d)}, \tag{6}$$

and

$$\Omega_{w,y}(d) = k_w Q_w \times \frac{1}{\Psi_{w,y}(d)}. \tag{7}$$

Remarkably, the output data size is quite micro compared with the input data, so the time of transmission can be neglected. The RSU y will execute the service φ_w after the data transmission process. The computation power allocated to the vehicle w by the RSU y is denoted as p_w^y. Since the RSU has limited computation power, the resource allocated to the vehicles must satisfy $\sum_{i=1}^{W} p_i^y \le P_R$. As a result, the service execution time of the vehicle y increases with more vehicles offloading their services to the RSU y, which conforms to the practical situation. The service ET of the vehicle w on the RSU y can be given as

$$\sigma_{w,y}(d) = G_w \times \frac{1}{p_w^y}. \tag{8}$$

Additionally, the time of vehicle w leaving the linking RSU y can be expressed as

$$T_{w,lev}^y = \frac{Ry - \varsigma_w}{v_w}, \tag{9}$$

where ς_w is the location of vehicle w and y denotes vehicle w running within the mth segment. In (9), $y = \lceil \frac{\varsigma_w}{R} \rceil$, $\lceil . \rceil$ is the ceiling function and v_w represents for the average velocity of the vehicle w. To ensure the service is accomplished before the vehicle w transfers from the RSU y of the current wireless connection to another adjacent unit, it must be satisfied that

$$\varrho_{w,y}(d) + \sigma_{w,y}(d) \le T_{w,lev}^y. \tag{10}$$

According to (6), (7) and (8) the overhead induced by executing the service on the RSU in the aspects of consumed energy and time can be computed as

$$x_{w,y} = \lambda_w^t \varrho_{w,y}(d) + \lambda_w^t \sigma_{w,y}(d) + \lambda_w^e \Omega_{w,y}(d). \tag{11}$$

3) Service Execution on the BS: Compared to the RSU, the BS possesses more powerful computation capacity. Nevertheless, there are fewer BS than RSUs in the researched segment, so it is more likely to be overloaded by users than the RSU. Additional overhead in the aspects of consumed energy and time is produced by conveying the input data to the BS. Since the BS is also closely located from the vehicles, the transmission time of output data can be dismissed. On account of the communication model, the time caused by data transmission and the consumed energy of the vehicle w are formulated as

$$\varrho_{w,B}(d) = Q_w \times \frac{1}{\Psi_{w,y}(d)}, \tag{12}$$

and

$$\Omega_{w,B}(d) = k_w Q_w \times \frac{1}{\Psi_{w,y}(d)}. \tag{13}$$

BS executes the service φ_w after the data transmission process. The computation power allocated to the vehicle w is denoted as p_w^B. Due to the fact that the BS has limited computation power, the resources allocated to the vehicles must satisfy $\sum_{i=1}^{W} p_i^B \le P_B$. The service ET of vehicle w on the BS is calculated as

$$\sigma_{w,y}(d) = G_w \times \frac{1}{p_w^B}. \tag{14}$$

According to (12), (13) and (14), the overhead of BS processing model in the aspects of consumed energy and time are computed as

$$x_{w,B} = \lambda_w^t \varrho_{w,B}(d) + \lambda_w^t \sigma_{w,B}(d) + \lambda_w^e \Omega_{w,B}(d). \tag{15}$$

4) Service Execution on the CS: By virtue of the fiber and core networks, the vehicle w offloads task φ_w to the cloud located hundreds of kilometers away. Additional overhead in the aspects of consumed energy and time is produced by delivering the computation input data to the cloud. Furthermore, although the output data size is quite smaller, the time for feeding back the output data from cloud to the vehicle w should be taken into consideration. The time caused by data transmission and the consumed energy of the vehicle w are formulated as,

$$\varrho_{w,C}(d) = \left(Q_w + Q_w^o\right) \times \frac{1}{\Psi_{w,y}(d)} + \alpha, \tag{16}$$

and

$$\Omega_{w,C}(d) = k_w \left(\left(Q_w + Q_w^o\right) \times \frac{1}{\Psi_{w,y}(d)} + \alpha \right). \tag{17}$$

In (16) and (17), Q_w^o is the size of output data. The delay from the vehicle w to the cloud is α. The cloud executes the task φ_w after the data transmission process. The computation power which the vehicle w obtains from the cloud is denoted as p_w^C. The computation power of the cloud is powerful enough to satisfy all the needs of the services, so there are no constraints and limitations in the resources allocation. The service ET of the vehicle w on the cloud is formulated as

$$\sigma_{w,C}(d) = G_w \times \frac{1}{p_w^C}. \tag{18}$$

According to (16), (17) and (18), the overhead of the cloud processing model in the aspects of consumed energy and time are computed as

$$x_{w,C} = \lambda_w^t \varrho_{w,C}(d) + \lambda_w^t \sigma_{w,C}(d) + \lambda_w^e \Omega_{w,C}(d). \tag{19}$$

According to the system model above, a game theoretic approach will be developed to devise an efficient computation offloading scheme in the following sections.

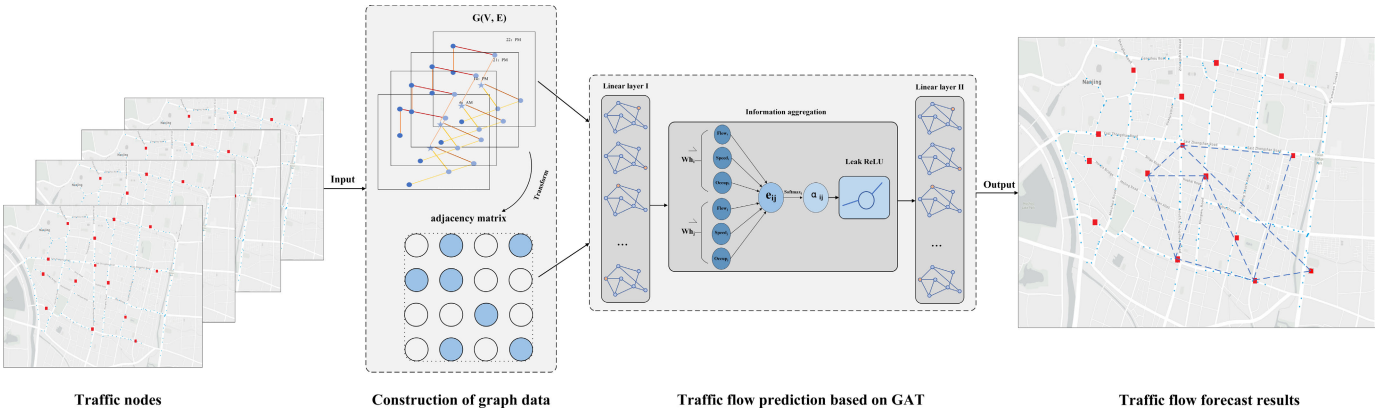

Fig. 2. The traffic flow forecasting scheme with graph attention neural networks.

IV. GAT Based Traffic Flow Forecasting

Due to the advantages of GAT in prediction [30], [31], GAT is adopted in this section to forecast the short-term traffic flow. According to the prediction results, the current load of the edge server is adjusted to prepare for the next phase of service offloading. The design of the traffic flow forecasting scheme with GAT is shown in Fig. 2.

A. The Construction of Graph Neural Network

Denote the set $C = \{c_1, c_2, \quad \cdots, c_n\}$ as the current load status of all edge servers in the selected area. To predict the traffic flow in the future time period, the selected area is divided into multiple road nodes which serves as hubs for traffic flow collection and prediction. Each road node is connected with the neighboring nodes by the different weights of edges based on the degree of correlation between the two nodes. Thus, the whole region can be regarded as an undirected connected graph.

In each time period, the road node collects three indicators of current traffic, denoted as the set $f\{flow, speed, occup\}$. In set f, $flow$ stands for the total traffic flow. $speed$ is the current average speed of vehicles and pedestrians. $occup$ is the average occupancy. The three indicators are the features of each node.

The set of node eigenvectors is denoted as

$$h = \left\{\vec{h}_1, \vec{h}_2, \quad \cdots, \vec{h}_N\right\}, \quad \vec{h}_i \in \mathbb{R}^F, \tag{20}$$

where N is the number of nodes. F is the number of node features which includes the set f, the geographical location of nodes and the local information of nodes. The size of the matrix h is $F \times N$ which represents the characteristics of all nodes. $\mathbb{R}$ represents the characteristics of only one node, so the size is $F \times 1$.

The objective function of traffic flow prediction is denoted as

$$h' = \left\{\vec{h'}_1, \vec{h'}_2, \quad \cdots, \vec{h'}_N\right\}, \quad \vec{h}_i \in \mathbb{R}^{F'}, \tag{21}$$

where $\vec{h'}_i$ is the prediction result of the eigenvectors in next time period of node i.

After the traffic flow results for the next time period are obtained, the current load of edge servers needs to be adjusted. Specifically, if the prediction result shows that the area covered by the edge server signal will face a surge in traffic flow, then some of the services on the server will be transferred to the adjacent servers for achieving the load balance of next phase. The load measurement function is defined as

$$K_i = \log_2\left(1 + \frac{c_i h_i}{\sum_{j=1}^{Y} c_j h_j}\right), \tag{22}$$

where h_i is the future traffic flow in the signal region of edge server i. The smaller the value of K_i is, the more favorable the current load is for the edge server. Meanwhile, the values of different K_i should be close. When the load of edge server is adjusted based on the forecast, the output result is the new load set $C' = \left\{c'_1, c'_2, \quad \cdots, c'_n\right\}$.

B. Real-Time Prediction of Traffic Flow

Due to the typical graph structure of transportation networks, graph neural networks (GNN) are capable of effectively capturing the relationships and interactions between nodes compared with other neural networks, thereby providing accurate predictions of traffic flow. GNNs integrate information from both nodes and edge, enabling precise predictions in large-scale transportation networks. GAT is an improved deep graph neural network which adds attention mechanism (AM) into the conventional graph neural network to realize the aggregation of domain nodes with distinction. AM is a technique that enables models to focus on critical information and fully absorb it. The core logic of the AM is to shift from focusing on the whole to focusing on the key. Thus, AM is leveraged to measure the correlation between the domain node and the central node.

The specific process of GAT are divided into three steps. The first step is to calculate the correlation degree between nodes, which is formulated as

$$e_{ij} = \alpha\left(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j\right) = \vec{a}^T\left[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j\right], \tag{23}$$

where $\vec{h}_i \in \mathbb{R}^F$ is the characteristics of node i, $\mathbf{W} \in \mathbb{R}^{F \times F'}$ is the learnable linear transformation parameter, $\alpha(\cdot) : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \to \mathbb{R}$ is the AM. Specifically, $\|$ stands for connecting two vectors sequentially.

The second step is to leverage softmax function to normalize the correlation between nodes. The purpose of normalization is to make it easy for the coefficient of attention between different nodes to be compared. The normalized function is defined as

$$\alpha_{ij} = \text{softmax}_j\left(e_{ij}\right) = \frac{\exp\left(e_{ij}\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(e_{ik}\right)}. \tag{24}$$

The domain nodes are aggregated with different information by applying the coefficient of attention to complete the convolution operation in the third step. The polymerization formula is

$$\vec{h}_i' = f\left(\sum_{j\in\mathcal{N}_i}\alpha_{i_j}\mathbf{W}\,\vec{h}_j\right), \tag{25}$$

where $f(\cdot)$ is a nonlinear activation function. The convolution kernel parameter $\mathbf{W}$ of GAT is shared by all the nodes in the domain. The coefficient of attention α_{i_j} represents the closeness of node i and node j. The larger the α_{i_j}, the closer the connection between the two nodes when the information of the surrounding neighborhood nodes is aggregated.

Based on the GAT, a novel traffic flow forecasting (TFF) algorithm is proposed. The core of the algorithm is to introduce the AM in the graph algorithm. By calculating the "attention coefficient" between the current node and its neighbors, the "attention coefficient" is weighted when the neighbors are aggregated. The graph neural network can pay more attention to the important nodes, so as to reduce the impact of edge noise. The details of TFF are given in Algorithm 1. To be specific, from line 1 to line 3, we construct an undirected connected graph of the studied sections according to the input information. From line 6 to line 10, correlations between different nodes are calculated. Then, different information is aggregated to domain nodes through attention coefficient to realize node updating. Repeating line 6 through 10 until the model has been trained for a predetermined number of times.

Algorithm 1 Traffic Flow Forecasting

Data: The indicators set $f\{flow, speed, occup\}$, the node eigenvectors set $h = \left\{\vec{h}_1, \vec{h}_2, \quad \cdots, \vec{h}_N\right\}$.
Result: The flow forecasting set $h' = \left\{\vec{h'}_1, \vec{h'}_2, \quad \cdots, \vec{h'}_N\right\}$.

1 **Start:**
2 The original data set is constructed into an undirected graph $G(V, E)$ and the edge weights are defined.
3 **end start**
4 **for** *each episode* **do**
5 Confirm novel learnable linear transformation parameter $\mathbf{W}$.
6 **for** $i = 0$ *to* t **do**
7 Calculate the correlation degree e_{ij} between node i and node j,
8 leverage $softmax$ function $\alpha_{ij} = \text{softmax}_j\left(e_{ij}\right) = \frac{\exp(e_{ij})}{\sum_{k\in\mathcal{N}_i}\exp(e_{ik})}$ to normalize the correlation between nodes,
9 the domain nodes are aggregated with different information by applying the coefficient of attention to complete the convolution operation.
10 **end**
11 Obtain the optimization set h'.
12 **end**

V. Potential Game Based Service Offloading

In this section, we fomulate the services offloading process in IoV as a potential game and prove that the potential game can achieve Nash equilibrium.

A. Game Formulation

Denote $d_{\ell w} = (d_1, \ldots, d_{w-1}, \ldots, d_{w+1}, \ldots, d_W)$ as the service execution decisions by all the other vehicles besides vehicle w. Given the decisions $d_{\ell w}$ of other vehicles, vehicle w will choose a proper offloading decision to obtain the minimum of the overhead function, i.e.,

$$\min_{d_w\in\tau}\chi_w\left(d_w, d_{-w}\right).$$

According to (5), (11), (15) and (19), the overhead function of the vehicle w can be obtained as

$$\chi_w\left(d_w, d_{\ell w}\right) = \begin{cases} x_{w,W}, & \text{if } d_{w,W} > 0, \\ x_{w,R}, & \text{if } d_{w,R} > 0, \\ x_{w,B}, & \text{if } d_{w,B} > 0, \\ x_{w,C}, & \text{if } d_{w,C} > 0. \end{cases} \tag{26}$$

The issue above is transformed as a strategic game

$$\Gamma = \left\{W, \{\tau_w\}_{w\in W}, \{\chi_w\}_{w\in W}\right\},$$

where $\chi_w\left(d_w, d_{\ell w}\right)$ denotes the overhead function of the vehicle w and τ_w is the set of strategies for the vehicle w. Following, the game Γ will be called as the resource and channel allocation game.

Definition 1: A strategy set $d^* = \left(d_1^*, \ldots, d_W^*\right)$ is called a Nash equilibrium for resource and channel allocation game if no vehicles are able to improve its benefit (reduce the value of the overhead function) by changing the strategy unilaterally at the equilibrium d^*.

$$\chi_w\left(d_w^*, d_{\ell w}^*\right) \le \chi_w\left(d_w, d_{\ell w}^*\right), \quad \forall d_w \in \tau_w. \tag{27}$$

According to (26), it can be divided into three game relations: (1) the game between local computing and computation offloading, (2) the game between MEC and cloud, (3) the internal game of MEC. Each vehicle experiences the three games during the decision-making process and finally chooses the optimal decision mode. In order to deal with the first layer of the game, the definition of beneficial computation offloading (BCO) is handed out.

Definition 2: Given the decision set d, the decision d_w of vehicle w which chooses computation offloading is beneficial if there exists one approach among executing the service on the RSU, the BS and the cloud that incur lower overhead than executing the service locally (i.e., $x_{w,y} < x_{w,W}$, or $x_{w,B} < x_{w,W}$, or $x_{w,C} < x_{w,W}$).

Lemma 1: Provided a decision set d, vehicle w realizes BCO if the received interference $\Theta_w^j = \sum_{i!=w} \Xi(i \in W)\Xi(d_{i,j} = d_{w,j})k_i D_{i,y}$, $j \in \{C, R, B\}$ *satisfies that* $\Theta_w^j \leq \partial_{w,j}$.

Proof 1: According to(2), (3), (4), (5), (11), (15), (19) and Definition 2, the condition $x_{w,j} \leq x_{w,W}$, $j \in \{C, R, B\}$ *is equivalent to*

$$\lambda_w^t \varrho_{w,j}(d) + \lambda_w^t \sigma_{w,j}(d) + \lambda_w^e \Omega_{w,j}(d) \leq \lambda_w^t \sigma_{w,W}(d) + \lambda_w^e \Omega_{w,W}(d). \tag{28}$$

Substitute (2), (3) and (4) into the above inequality respectively. The inequality is transformed as

$$\frac{\left(\lambda_w^t + \lambda_w^e k_w\right) Q_w}{\Psi_{w,y}(d)} + \lambda_w^t \sigma_{w,y}(d) \leq \lambda_w^t \sigma_{w,W}(d) + \lambda_w^e \Omega_{w,W}(d).$$

That is,

$$\Psi_{w,y}(d) \geq \frac{\left(\lambda_w^t + \lambda_w^e k_w\right) Q_w}{\lambda_w^t \sigma_{w,W}(d) + \lambda_w^e \Omega_{w,W}(d) - \lambda_w^t \sigma_{w,y}(d)}.$$

Expand out $\Psi_{w,y}(d)$ *by the formula. It turns out that*

$$\sum_{i!=w} \Xi(i \in W)\Xi(d_{i,y} = d_{w,y})k_i D_{i,y} \leq \frac{k_w D_{w,y}}{\frac{\left(\lambda_w^t + \lambda_w^e k_w\right) Q_w}{2^{\omega\left(\lambda_w^t \sigma_{w,W}(d) + \lambda_w^e \Omega_{w,W}(d) - \lambda_w^t \sigma_{w,y}(d)\right)}} - 1} - \delta. \tag{29}$$

Let $\Theta_w^y = \sum_{i!=w} \Xi(i \in W)\Xi(d_{i,y} = d_{w,y})k_i D_{i,y}$ *and* $\partial_{w,y} = \frac{k_w D_{w,y}}{\frac{\left(\lambda_w^t + \lambda_w^e k_w\right) Q_w}{2^{\omega\left(\lambda_w^t \sigma_{w,W}(d) + \lambda_w^e \Omega_{w,W}(d) - \lambda_w^t \sigma_{w,y}(d)\right)}} - 1} - \delta$. *Then (29) is equivalent to* $\Theta_w^y \leq \partial_{w,y}$.

similarly, substitute (15) and (19) into the above inequality. It can finally obtain that $\Theta_w^B \leq \partial_{w,B}$ *and* $\Theta_w^C \leq \partial_{w,C}$ *respectively.*

According to Definintion 2, if vehicle w satisfies

$$\begin{cases} \Theta_w^y \geq \partial_{w,y}, \\ \Theta_w^B \geq \partial_{w,B}, \\ \Theta_w^C \geq \partial_{w,C}, \end{cases} \tag{30}$$

it processes the service locally. Otherwise, it achieves BCO and will offload its task to the RSU, the BS or the cloud based on specific conditions. Lemma 1 shows that the vehicle is advisable to offload its task to other equipment when the received interference $\Theta_{w,j}$, $j \in \{R, B, C\}$ *on a certain wireless channel is lower enough. Nevertheless, high interference causes lower transmission rates and unbearable delay. Local processing is more reliable under this condition.*

The vehicles fall into two categories after the first stage of the game. Those who fail to obtain the BCO dispose the service on the vehicles, exiting the second phase of the game. Meanwhile other vehicles start the game between edge computing and cloud computing. According to (11), (15) and (19), if vehicle w selects the cloud to execute the service, it must satisfy the condition that $x_{w,C} \leq \min\left\{x_{w,B}, x_{w,y}\right\}$.

Expand the above inequalities, it is obtained that

$$\frac{\left(Q_w + Q_w^o\right)}{\Psi_{w,y}(d)} - \frac{Q_w}{\Psi_{w,y}(d)} \leq \frac{\lambda_w^t \sigma_{w,B}(d) - \lambda_w^e k_w \alpha + \lambda_w^t\left(\sigma_{w,C}(d) + \alpha\right)}{\lambda_w^t + \lambda_w^e k_w}, \tag{31}$$

and

$$\frac{\left(Q_w + Q_w^o\right)}{\Psi_{w,y}(d)} - \frac{Q_w}{\Psi_{w,y}(d)} \leq \frac{\lambda_w^t \sigma_{w,y}(d) - \lambda_w^e k_w \alpha + \lambda_w^t\left(\sigma_{w,C}(d) + \alpha\right)}{\lambda_w^t + \lambda_w^e k_w}. \tag{32}$$

Thus, if the uplink data rates of vehicle w caters to above inequalities, it selects CSs to offload the service, otherwise edge computing becomes its optimal option. The inequalities imply that under the circumstance of all other parameters being fixed, the offloading decision of vehicle w depends on the number of the existing vehicles on the wireless link. The network congestion will increase if excessive vehicles choose the same wireless link simultaneously, thus slowing down the data upload rate of the vehicle.

Those who choose to execute the task on the cloud will retreat from the third layer of the game. The remaining vehicles participate in the internal game of edge computing, which means if $x_{w,B} \leq x_{w,y}$, vehicle w selects the BS to offload the service, otherwise, it finally selects the RSU to process the task. According to $x_{w,B} \leq x_{w,y}$, we can get

$$\frac{1}{\Psi_{w,y}(d)} - \frac{1}{\Psi_{w,y}(d)} \leq \frac{\lambda_w^t \sigma_{w,y}(d) - \lambda_w^t \sigma_{w,B}(d)}{\left(\lambda_w^t + \lambda_w^e k_w\right) Q_w}. \tag{33}$$

B. Structural Properties

Potential game, as a formidable instrument, is resorted to testify the resource and channel allocation game satisfying Nash equilibrium next.

Definition 3: If a game satisfies the function $\phi(d)$ *such that for every* $w \in W$, $d_{\ell w} \in \prod_{i \neq w} \tau_i$, d'_w, $d_w \in \tau_w$, *if*

$$\chi_w\left(d'_w, d_{\ell w}\right) \leq \chi_w\left(d_w, d_{\ell w}\right), \tag{34}$$

we have

$$\phi_w\left(d'_w, d_{\ell w}\right) \leq \phi_w\left(d_w, d_{\ell w}\right), \tag{35}$$

it is a potential game.

Lemma 2: The resource and channel allocation game has the nature of the potential game.

Proof 2: Construct the potential function $\phi(d_w, d_{\ell w})$ *as*

$$\begin{aligned} &\phi_w(d_w, d_{\ell w}) \\ &= \frac{1}{2}\sum \Xi(\mu \in W) \sum_{\iota!=\mu} \Xi(\iota \in W) k_\mu D_{\mu,y} k_\iota D_{\iota,y} * \\ &\quad \times \Lambda\left\{d_{\mu,k}, d_{\iota,k}\right\} \Upsilon\left\{d_{\mu,k}\right\} \\ &\quad + \sum \Xi(\mu \in W) k_\mu D_{\mu,y} \partial_\mu \Upsilon\left\{d_{\mu,W}\right\}, \\ &\quad y \in \{1, 2, \ldots, Y\}, k \in \{R, B, C\}. \end{aligned} \tag{36}$$

In (36), $\Lambda\{a, b\}$ *is an indicator function where* $\Lambda\{a, b\} = 1$ *if* $a = b$, *otherwise* $\Lambda\{a, b\} = 0$. $\Upsilon\{h\}$ *is a judgement function. If* $h > 0$, $\Upsilon\{h\} = 1$, *else* $\Upsilon\{h\} = 0$. *The wireless interference* ∂_μ *for choosing offloading decisions can be*

obtained according to the condition $\min\{x_{\mu,y}, x_{\mu,C}, x_{\mu,B}\} \leq x_{\mu,W}$. *So we can get that*

$$\partial_\mu = \begin{cases} \partial_{\mu,y}, & \text{if } x_{\mu,y} \leq x_{\mu,C} \text{ and } x_{\mu,y} \leq x_{\mu,B}, \\ \partial_{\mu,B}, & \text{if } x_{\mu,B} \leq x_{\mu,y} \text{ and } x_{\mu,B} \leq x_{\mu,C}, \\ \partial_{\mu,C}, & \text{if } x_{\mu,C} \leq x_{\mu,y} \text{ and } x_{\mu,C} \leq x_{\mu,B}. \end{cases} \tag{37}$$

Then referring to the equations above, it can be proven that the resource and channel allocation game has the nature of a potential game. To prove this, the update process is partitioned into four cases.

Case 1: $d_{\mu,W} > 0$, $d'_{\mu,k} > 0$, $k \in \{R, B, C\}$. *In such case it is obtained that*

$$\sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d'_{\mu,k}, d_{\iota,k} \right\} \leq \partial_\mu. \tag{38}$$

So we have

$$\begin{aligned} &\phi\left(d_\mu, d_{\ell\mu}\right) - \phi\left(d'_\mu, d_{\ell\mu}\right) \\ &= k_\mu D_{\mu,y} \partial_\mu - \frac{1}{2} \sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d'_{\mu,k}, d_{\iota,k} \right\} \\ &\quad - \frac{1}{2} \sum_{\mu!=\iota} \Xi(\mu \in W) k_\mu D_{\mu,y} \Lambda \left\{ d_{\iota,k}, d'_{\mu,k} \right\} \\ &= k_\mu D_{\mu,y} \left(\partial_\mu - \sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d'_{\mu,k}, d_{\iota,k} \right\} \right) > 0. \end{aligned} \tag{39}$$

Case 2: $d_{\mu,y} > 0$, $d'_{\mu,B} > 0$. *According to (34), it is obtained that*

$$\begin{aligned} &\sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d'_{\mu,B}, d_{\iota,B} \right\} \\ &\leq \sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d_{\mu,y}, d_{\iota,y} \right\}. \end{aligned} \tag{40}$$

Let $\pi'_{\mu,j} = \sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d'_{\mu,j}, d_{\iota,j} \right\}$ *and* $\pi_{\mu,j} = \sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d_{\mu,j}, d_{\iota,j} \right\}$, $j \in (y, B, C)$.

$$\begin{aligned} &\phi\left(d_\mu, d_{\ell\mu}\right) - \phi\left(d'_\mu, d_{\ell\mu}\right) \\ &= k_\mu D_{\mu,y} \left(\pi_{\mu,y} - \pi'_{\mu,B} \right) > 0. \end{aligned} \tag{41}$$

Case 3: $d_{\mu,y} > 0$, $d'_{\mu,C} > 0$. *According to (33), it is obtained that*

$$\begin{aligned} &\sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d'_{\mu,C}, d_{\iota,C} \right\} \\ &\leq \sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d_{\mu,y}, d_{\iota,y} \right\}. \end{aligned} \tag{42}$$

$$\begin{aligned} &\phi\left(d_\mu, d_{\ell\mu}\right) - \phi\left(d'_\mu, d_{\ell\mu}\right) \\ &= k_\mu D_{\mu,y} \left(\pi_{\mu,y} - \pi'_{\mu,C} \right) > 0. \end{aligned} \tag{43}$$

Case 4: $d_{\mu,B} > 0$, $d'_{\mu,C} > 0$. *According to (32), it is obtained that*

$$\begin{aligned} &\sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d'_{\mu,C}, d_{\iota,C} \right\} \\ &\leq \sum_{\iota!=\mu} \Xi(\iota \in W) k_\iota D_{\iota,y} \Lambda \left\{ d_{\mu,B}, d_{\iota,B} \right\}. \end{aligned} \tag{44}$$

$$\begin{aligned} &\phi\left(d_\mu, d_{\ell\mu}\right) - \phi\left(d'_\mu, d_{\ell\mu}\right) \\ &= k_\mu D_{\mu,y} \left(\pi_{\mu,B} - \pi'_{\mu,C} \right) > 0. \end{aligned} \tag{45}$$

Algorithm 2 Game Theoretical Based Distributed Computation Offloading

Result: Decision profile d and the total minimum utility value x

1 **Start:**
2 Each vehicle w performs the service on its own device. That is, $d_{w,W}(0) = 1$.
3 **end start**
4 **Repeat:**
5 **for** *vehicle* w *in each time slot* t **do**
6 BS collects information of all vehicles and compute their departure time $T^y_{w,lev} = \frac{Ry - l_w}{v_w}$
7 BS collects information of all channels and sends them to the vehicles.
8 Compute the data uplink rates $\Psi_{w,y}(d_{w,y})$,$\Psi_{w,y}(d_{w,B})$,$\Psi_{w,y}(d_{w,C})$ and total time T^y_w, T^B_w, T^C_w, repectively
9 **end**
10 **if** $min\{T^y_w, T^B_w, T^C_w\} \geq T^y_{w,lev}$ **then**
11 Vehicle w performs the service on local device and quits the game at this phase, i.e., $W = W \setminus \{w\}$
12 **end**
13 The remaining vehicles play the three-tier game and obtain the optimal response set $C_w(t)$
14 **if** $C_w(t) \neq \phi$ **then**
15 The vehicles transmit request signal to the BS, competing for the opportunity of updating.
16 **if** *acquires permission signal from the BS* **then**
17 Vehicle w chooses the decision $d_w(t+1) \in C_w(t)$ in next phase
18 **else**
19 Choose the original $d_w(t+1) = d_w(t)$ in next phase.
20 **end**
21 **end**
22 **Until** $C_w(t) = \phi$ and the BS sends the END message.

The core idea of the proof is to testify when vehicle w *updates its present* d_i *to the optimized decision* d'_i, *the majorization in* χ_w *is mapped to the majorization in the* ϕ_w.

Based on the game formulation and the asynchronous optimization of the resource and channel allocation game, a novel game theoretical based distributed traffic flow forecasting and task offloading (TFFTO) algorithm is designed. To be specific, TFFTO is primarily composed of three stages. From line 4 to line 9 in the first stage, the total transmission and executed time of each task through computing the task locally and selecting the MEC/CS is calculated respectively. Then, the departure time $T^y_{n,left}$ of each vehicle is calculated. From line 10 to line 12, the services which fail to meet the maximum time constraints, i.e., $min\{T^y_w, T^B_w, T^C_w\} \geq T^y_{w,lev}$, $\forall w \in W$, are

discarded from the game set and forced to be processed locally. Finally, from line 14 to line 21, the remaining vehicles, whose tasks are able to be processed locally or be offloaded to the MEC/CS, play the three-tier game. Repeat the process above until the game reaches a Nash equilibrium. The details of TFFTO are given in Algorithm 2.

C. Time Complexity Analysis

In this subsection, we approximate the time complexity of the proposed algorithm. Multi-user service offloading is an NP-Hard problem, and we optimize the time complexity of this problem using a combination of GAT and potential game approaches. After training, the GAT model only involves the forward propagation process, and the time complexity depends on the number of nodes and edges, as well as the number of neighbors for each node. Typically, the time complexity is $O(N+E)$, where N is the number of nodes and E is the number of edges. The time complexity of potential games is related to the state space and action space, and in simpler cases, it can be $O(n)$, while in special cases, it may reach exponential complexity. Therefore, the overall time complexity of the proposed algorithm in this paper is similar to the existing complexity.

VI. Performance Evaluation

The numerical results to evaluate the TFFTO proposed by us are presented in this section.

A. Experiment Setup

In the experiment, we utilize the real dataset collected from the vehicles in September 2014 from Nanjing to evaluate the performance of TFF in TFFTO. There are 436 RSUs collecting traffic metrics in the dataset.

In the simulation experiment scenario of service offloading, we intercept a specific area of Nanjing and take the results of traffic flow prediction as the input set. The CS is 500 km away from the tested road and 8 RSUs are installed uniformly along the road, each with a wireless signal coverage range of 200 m. The transmission power $k_w = 2$ W and the background noise $\delta = 1.5 \times 10^{-8}$ W. The channel gain is set as $D_{w,y} = \varsigma_{w,y}^{-f}$ based on the model of wireless interference in urban environment, where $\varsigma_{w,y}$ is the relative position of the vehicle w and RSU y and $f = 4$ is the general loss coefficient.

The data size of each service is $Q_w = 5000$ K and the total quantity of CPU revolutions is $G_w = 1$ Gigacycles. The computation power p_w^L of vehicle w is stochastically appointed from the set $\{0.4, 0.7, 1.0\}$ GHz. The computation power distributed for the vehicle w by the CSs is $p_w^C = 10$ GHz. The computation powers of each RSU and the BS are assigned as $P_R = 10$ GHz and $P_B = 20$ GHz respectively. As to the decision weights of vehicle w for both the consumed energy and the computation time, we set that $\lambda_w^e = 1 - \lambda_w^t$, where λ_w^t is stochastically appointed from the set $\{1, 0.5, 0\}$. Some parameters above are set according to [32] and [33]. Others are set based on the real environment of the road section being studied in Nanjing.

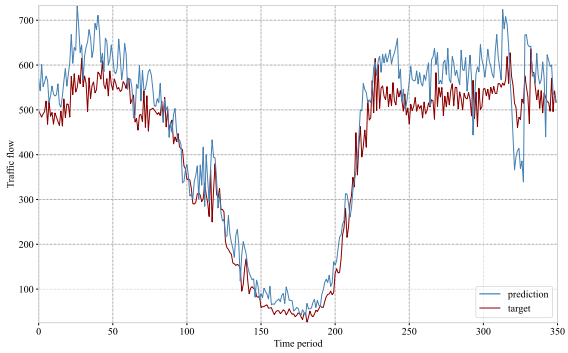

Fig. 3. Traffic flow forecast value and actual value.

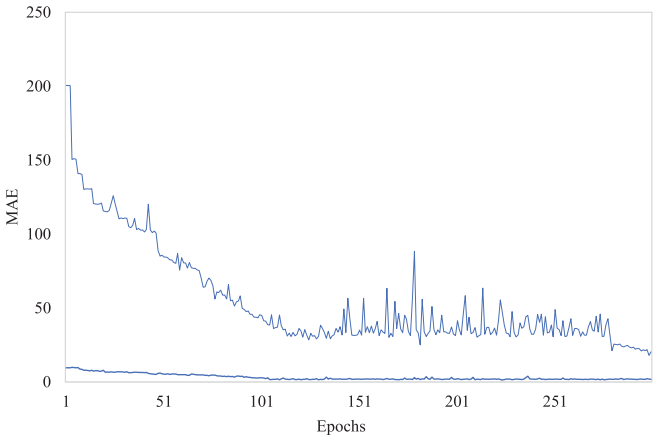

Fig. 4. Mean absolute error value.

B. Numerical Results of TFF

In the following section, the performance of TFF will be measured from four perspectives: the difference between the predict value and the true value, Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and test-loss.

1) Numerical Analysis on Prediction Accuracy: Fig. 3 shows the comparison of the predicted average flow over a period of time with the real value of the selected region. The horizontal axis represents time segments and the vertical axis represents the current traffic flow on the road section. The data shows that the predicted and true values of traffic flow are roughly in line, with the prediction accuracy remaining roughly constant over time. There is some distortion in the peak and trough of traffic flow, and the predicted value is slightly greater than the true value. Overall, the accuracy of the flow forecast is very high.

2) Analysis on MAE and MAPE: Fig. 4 and Fig. 5 show the variations of MAE and MAPE respectively. MAE in the results is 1000 times the actual value, so the MAE value of the TFF algorithm is very low, which proves that the prediction results of this algorithm have high accuracy. As the epoch times increase, the value of MAPE gradually declines and eventually converges to around 2, indicating that the average error of the algorithm is very small.

3) Analysis on Test-Loss: Test-loss is an important index based on MAPE and MAE to measure the distortion rate of forecast results. In Fig. 6, each point on the polyline represents

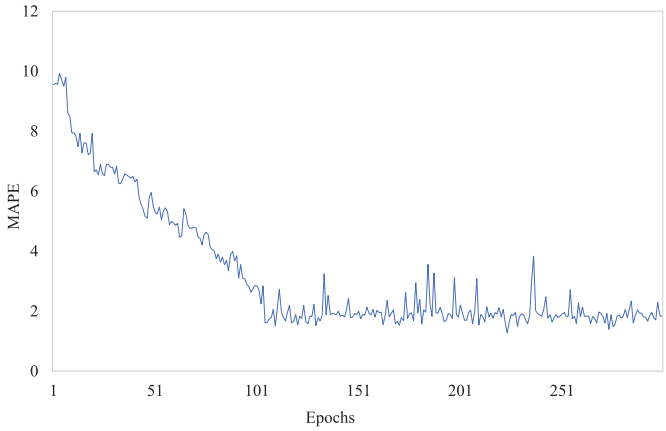

Fig. 5. Mean absolute percentage error value.

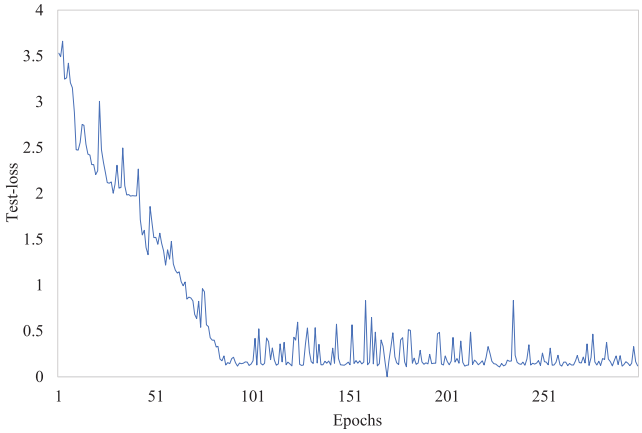

Fig. 6. Test-loss value.

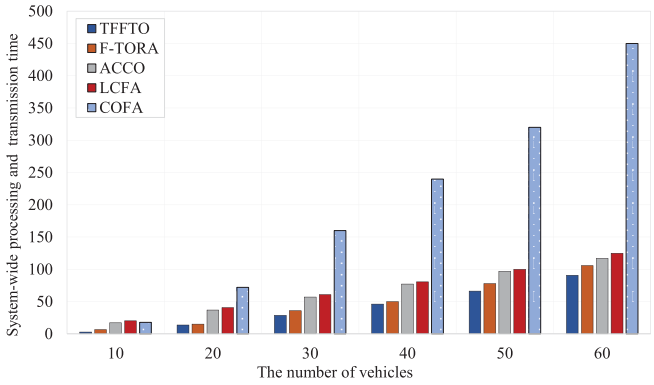

Fig. 7. Comparative analysis on system-wide processing and transmission time.

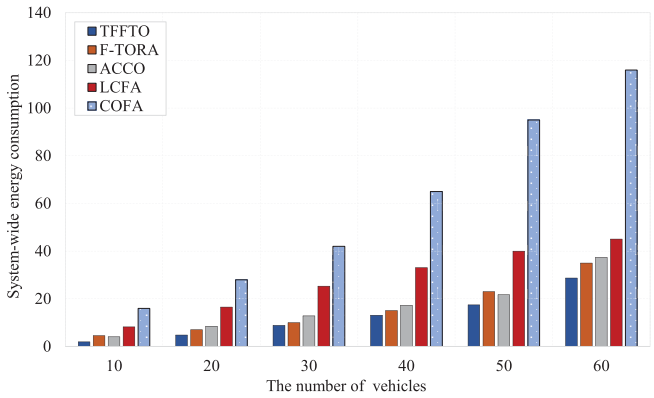

Fig. 8. Comparative analysis on system-wide energy consumption.

a test-loss value of prediction. As the epoch times increase, most of the points are clustered along the axis near 0, and only a few points have values above 0.5, which proves that the prediction value of the TFF algorithm has high accuracy.

C. Numerical Results of TFFTO

In the following sections, the numerical results of the game theory based TFFTO are compared with that of four other service processing modes.

- Local computing for all the vehicles (LCFA): all the vehicles choose to process the service locally.
- Computation offloading for all the vehicles (COFA): all the vehicles choose to offload the task to the nearest RSU server.
- The task offloading and resource allocation method F-TORA in [34].
- An approximation collaborative computation offloading algorithm (ACCO) proposed in [35].

It is worth noting that ACCO is another method based on game theory. The five methods (TFFTO, F-TORA, ACCO, LCFA and COFA) are applied to delay sensitive services, high energy consumption services, and services with a mixture of the two features respectively to verify that the game theory method based on GAT proposed by us is obviously better than the single game theory based method and other service offloading methods when faced all kinds of service requests. Moreover, the convergence of TFFTO will also be evaluated.

1) Comparative Analysis on Average System-Wide Processing and Transmission Time: The comparison of system-wide processing and transmission time between TFFTO and other four methods (F-TORA, ACCO, LCFA and COFA) is shown in Fig. 7. With fewer than 20 vehicles, the system-wide processsing and transmission time based on TFFTO is nearly 5 times lower than that of ACCO. As the number of the vehicles grows, this value falls but is still optimized by nearly 30 percent when the number of vehicles reaches 50. At all times, the result of TFFTO is vastly superior to those obtained by the other two methods LCFA and COFA. Compared with F-TORA, the results of TFFTO is slightly better when the traffic scale is small, but TFFTO's advantages become more prominent when the number of vehicles increases. Therefore, in the face of delay-sensitive services, TFFTO can effectively reduce the total delay of the system.

2) Comparative Analysis on Average System-Wide Energy Consumption: Fig. 8 shows the system-wide energy consumption based on TFFTO, F-TORA, ACCO, LCFA and COFA respectively. Although the degree of optimization is not as great as that of the time dimension, TFFTO still outperforms the other four methods. At any traffic scale, the energy consumption generated by using TFFTO is significantly lower than that generated by using LCFA and COFA. Additionally, under TFFTO, the system-wide energy consumption is reduced

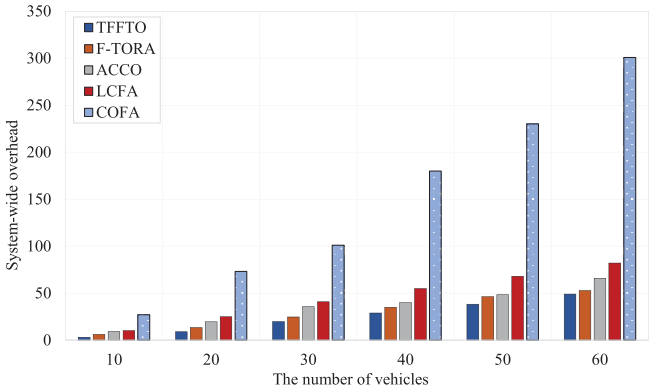

Fig. 9. Comparative analysis on system-wide overheard.

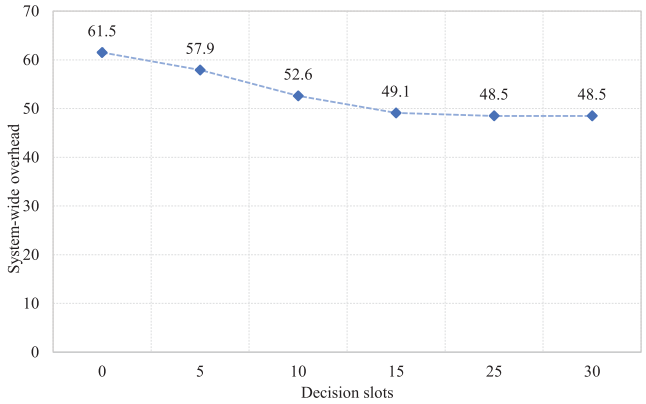

Fig. 10. The variety of system-wide overhead during each decision slot.

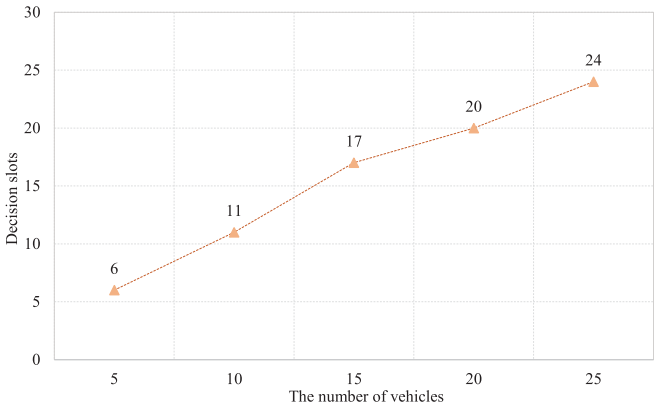

Fig. 11. Average decision slots for different numbers of vehicles.

by 25 percent compared with that of ACCO and reduced by 15 percent compared with that of F-TORA.

3) Comparative Analysis on Average System-Wide Overhead: Fig. 9 shows the comparison of the average system-wide overhead generated by TFFTO and four other service offloading methods: F-TORA, ACCO, LCFA and COFA. The experiments are conducted with $W = \{10, 20, 30, 40, 50, 60\}$ vehicles respectively. The average system-wide overheads of the five service offloading schemes are in close proximity when there are few vehicles at first. With the increase of the vehicles, the overhead generated by LCFA grows steadily and shows a state of approximately linear growth. The result is very close to reality, although the computing capacity and the task size vary from vehicle to vehicle, they are generally close, which means the overhead induced by executing the service locally is almost the same for each vehicle. Hence, as the number of the vehicles rises, the total overhead is approximately equal to the overhead of a vehicle times the number of the vehicles. The overhead caused by COFA grows slowly in the first few experiments. Nevertheless, it explodes when the number of the vehicles reaches thirty, which shows that ESs tend to become resources constrained due to overloaded. The performance of TFFTO algorithm is nearly 17 percent better than LCFA and 80 percent better than COFA, which can largely improve the QoS. It is evident that the TFFTO outperforms all other two computation models, the overhead has been growing slowly and steadily, undistributed by the number of the vehicles, which proves the algorithm is able to ensure users' experience during heavy traffic. Additionally, TFFTO and F-TORA achieve almost the same results on a smaller vehicle scale, and TFFTO is slightly better than F-TORA at a certain vehicle scale. When the number of the vehicles is less than 20, the overhead based on ACCO is twice more than that of TFFTO. As the number of vehicles grows sustainably, the BS tends to be saturated with users, which means an increasing number of vehicles is opting for local execution. Nevertheless, the system-wide overhead of TFFTO is still nearly 25 percent optimized than that of ACCO. As a result, TFFTO outperforms the ACCO in terms of reducing system overhead. To sum up, TFFTO is superior to the other four schemes at any traffic scale.

4) Analysis on Convergence: Fig. 10 shows that the system-wide overhead decreases at each decision slot and finally converges to a fixed value after a finite step decision, which proves that the TFFTO algorithm possesses perfect convergence and can reach Nash equilibrium. In Fig. 11, as the number of the vehicles rises, the average quantity of decision slots for convergence grows approximately linearly which demonstrates that TFFTO converges rapidly and is less affected by the number of the vehicles. Therefore, the time complexity of the proposed service offloading algorithm based on potential games can be approximated as linear complexity. As a result, the scheme has perfect stability and convergence.

VII. Conclusion and Future Work

A novel approach based on graph attention networks and game theory is proposed to solve the service offloading issue of the vehicles. Specifically, graph attention networks are leveraged to predict the traffic flow in different time periods. Then the service offloading issue is formulated as a resource and channel allocation game which is proved to possess the nature of a potential game. Furthermore, a game theory based distributed computation offloading algorithm is designed to optimize the computation offloading problem. The numerical results demonstrate that TFFTO outperforms its representative counterparts.

In the future, we are committed to consider the computation offloading problem in the 5G scenario, where a task can be split into multi-subtasks and vehicles can transmit data to each other.

REFERENCES

[1] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task offloading in vehicular edge computing networks: A load-balancing solution," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2092–2104, Feb. 2020.

[2] J. Li et al., "Maximizing user service satisfaction for delay-sensitive IoT applications in edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 5, pp. 1199–1212, May 2022.

[3] Z. Ning et al., "Intelligent edge computing in Internet of Vehicles: A joint computation offloading and caching solution," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2212–2225, Apr. 2021.

[4] Y. Wang et al., "A game-based computation offloading method in vehicular multiaccess edge computing networks," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4987–4996, Jun. 2020.

[5] S. Josilo and G. Dan, "Computation offloading scheduling for periodic tasks in mobile edge computing," *IEEE/ACM Trans. Netw.*, vol. 28, no. 2, pp. 667–680, Apr. 2020.

[6] G. Cui et al., "Interference-aware SaaS user allocation game for edge computing," *IEEE Trans. Cloud Comput.*, vol. 10, no. 3, pp. 1888–1899, Jul. 2022.

[7] F. Zhou and R. Q. Hu, "Computation efficiency maximization in wireless-powered mobile edge computing networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3170–3184, May 2020.

[8] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint computation offloading and user association in multi-task mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12313–12325, Dec. 2018.

[9] Z. Ning et al., "Distributed and dynamic service placement in pervasive edge computing networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 6, pp. 1277–1292, Jun. 2021.

[10] T. Q. Dinh, Q. D. La, T. Q. S. Quek, and H. Shin, "Learning for computation offloading in mobile edge computing," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6353–6367, Dec. 2018.

[11] Y. Li, X. Wang, X. Gan, H. Jin, L. Fu, and X. Wang, "Learning-aided computation offloading for trusted collaborative mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 12, pp. 2833–2849, Dec. 2020.

[12] J. Zheng, Y. Cai, Y. Wu, and X. Shen, "Dynamic computation offloading for mobile cloud computing: A stochastic game-theoretic approach," *IEEE Trans. Mobile Comput.*, vol. 18, no. 4, pp. 771–786, Apr. 2019.

[13] S. Jošilo and G. Dán, "Selfish decentralized computation offloading for mobile cloud computing in dense wireless networks," *IEEE Trans. Mobile Comput.*, vol. 18, no. 1, pp. 207–220, Jan. 2019.

[14] H. Cao and J. Cai, "Distributed multiuser computation offloading for cloudlet-based mobile cloud computing: A game-theoretic machine learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 752–764, Jan. 2018.

[15] T. Bahreini, H. Badri, and D. Grosu, "Mechanisms for resource allocation and pricing in mobile edge computing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 3, pp. 667–682, Mar. 2022.

[16] M. Li, C. Huang, and D. Wang, "Robust stochastic configuration networks with maximum correntropy criterion for uncertain data regression," *Inf. Sci.*, vol. 473, pp. 73–86, Jan. 2019.

[17] W. Fan et al., "Joint task offloading and resource allocation for vehicular edge computing based on V2I and V2V modes," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 4, pp. 4277–4292, Apr. 2023.

[18] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, May 2018.

[19] S. Deng, C. Zhang, C. Li, J. Yin, S. Dustdar, and A. Y. Zomaya, "Burst load evacuation based on dispatching and scheduling in distributed edge networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 8, pp. 1918–1932, Aug. 2021.

[20] Z. Hong, H. Huang, S. Guo, W. Chen, and Z. Zheng, "QoS-aware cooperative computation offloading for robot swarms in cloud robotics," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 4027–4041, Apr. 2019.

[21] Y. Wu, K. Ni, C. Zhang, L. P. Qian, and D. H. K. Tsang, "NOMA-assisted multi-access mobile edge computing: A joint optimization of computation offloading and time allocation," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12244–12258, Dec. 2018.

[22] H. Feng, S. Guo, L. Yang, and Y. Yang, "Collaborative data caching and computation offloading for multi-service mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 9408–9422, Sep. 2021.

[23] M. Tang and V. W. S. Wong, "Deep reinforcement learning for task offloading in mobile edge computing systems," *IEEE Trans. Mobile Comput.*, vol. 21, no. 6, pp. 1985–1997, Jun. 2022.

[24] M. Gao, R. Shen, L. Shi, W. Qi, J. Li, and Y. Li, "Task partitioning and offloading in DNN-task enabled mobile edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 22, no. 4, pp. 2435–2445, Apr. 2023.

[25] G. Mitsis, E. E. Tsiropoulou, and S. Papavassiliou, "Price and risk awareness for data offloading decision-making in edge computing systems," *IEEE Syst. J.*, vol. 16, no. 4, pp. 6546–6557, Dec. 2022.

[26] H. Teng, Z. Li, K. Cao, S. Long, S. Guo, and A. Liu, "Game theoretical task offloading for profit maximization in mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 22, no. 9, pp. 5313–5329, Sep. 2023.

[27] Y. Chen, J. Zhao, Y. Wu, J. Huang, and X. S. Shen, "QoE-aware decentralized task offloading and resource allocation for end-edge-cloud systems: A game-theoretical approach," *IEEE Trans. Mobile Comput.*, vol. 23, no. 1, pp. 769–784, Jan. 2024.

[28] Z. Fang, X. Xu, F. Dai, L. Qi, X. Zhang, and W. Dou, "Computation offloading and content caching with traffic flow prediction for Internet of Vehicles in edge computing," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Oct. 2020, pp. 380–388.

[29] C. Chen, Z. Liu, S. Wan, J. Luan, and Q. Pei, "Traffic flow prediction based on deep learning in Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3776–3789, Jun. 2021.

[30] Y. Liu, C. S. Chen, C. W. Sung, and C. Singh, "A game theoretic distributed algorithm for FeICIC optimization in LTE–A HetNets," *IEEE/ACM Trans. Netw.*, vol. 25, no. 6, pp. 3500–3513, Dec. 2017, doi: 10.1109/TNET.2017.2748567.

[31] X. Zheng et al., "How framelets enhance graph neural networks," 2021, *arXiv:2102.06986*.

[32] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.

[33] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[34] X. Xu et al., "Game theory for distributed IoV task offloading with fuzzy neural network in edge computing," *IEEE Trans. Fuzzy Syst.*, vol. 30, no. 11, pp. 4593–4604, Nov. 2022.

[35] H. Guo and J. Liu, "Collaborative computation offloading for multiaccess edge computing over fiber-wireless networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4514–4526, May 2018.

Qinting Jiang is currently pursuing the B.S. degree in software engineering with the School of Software, Nanjing University of Information Science and Technology. His research interests include deep learning and edge computing.

Xiaolong Xu (Senior Member, IEEE) received the Ph.D. degree in computer science and technology from Nanjing University, China, in 2016. From April 2017 to May 2018, he was a Research Scholar with Michigan State University, USA. He is currently a Professor with the School of Software, Nanjing University of Information Science and Technology. His research interests include edge computing, the Internet of Things (IoT), cloud computing, and big data.

Muhammad Bilal (Senior Member, IEEE) received the Ph.D. degree in information and communication network engineering from the School of Electronics and Telecommunications Research Institute (ETRI), Korea University of Science and Technology, in 2017. From 2017 to 2018, he was with Korea University, where he was a Post-Doctoral Research Fellow with the Smart Quantum Communication Center. In 2018, he joined the Hankuk University of Foreign Studies, South Korea, where he is currently an Associate Professor with the Division of Computer and Electronic Systems Engineering. In 2023, he joined Lancaster University as a Senior Lecturer (Associate Professor) with the School of Computing and Communications. He is a prolific author, known for his wide-ranging contributions to numerous articles published in internationally renowned, top-tier journals. His pioneering work has also led to the successful acquisition of multiple U.S. and Korean patents. His research interests include network optimization, cyber security, the Internet of Things, vehicular networks, information-centric networking, digital twins, artificial intelligence, and cloud/fog computing. He is an esteemed member of the editorial boards for several prominent journals, including IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE INTERNET OF THINGS JOURNAL, IEEE Future Directions in Technology, Policy, and Ethics Newsletter, *Alexandria Engineering Journal* (Elsevier), and *Physical Communication* (Elsevier). He also serves as the Co-Editor-in-Chief of the *International Journal of Smart Vehicles and Smart Transportation*. In addition, he has actively contributed as a Technical Program Committee Member for leading international conferences, such as the IEEE Vehicular Technology Conference (VTC), the IEEE International Conference on Communications (ICC), ACM SigCom, and the IEEE Consumer Communications and Networking Conference (CCNC).

Jon Crowcroft (Fellow, IEEE) received the degree in physics from Trinity College, University of Cambridge, Cambridge, U.K., in 1979, and the M.Sc. and Ph.D. degrees in computing from University College London, London, U.K., in 1981 and 1993, respectively. From 2016 to 2018, he was the Programme Chair with the Alan Turing Institute, U.K. National Data Science and AI Institute, London, U.K. He is currently a Researcher with the Alan Turing Institute. Since October 2001, he has been a Marconi Professor of Communications Systems with the Department of Computer Science and Technology, University of Cambridge. His research interests include internet support for multimedia communications, scalable multicast routing, practical approaches to traffic management, the design of deployable end-to-end protocols, opportunistic communications, social networks, privacy-preserving analytics, and techniques and algorithms to scale infrastructure-free mobile systems. He is a fellow of the Royal Society, ACM, British Computer Society, IET, and the Royal Academy of Engineering.

Qi Liu (Senior Member, IEEE) received the B.S. degree in computer science and technology from Zhuzhou Institute of Technology, China, in 2003, and the M.S. and Ph.D. degrees in data telecommunications and networks from the University of Salford, U.K., in 2006 and 2010, respectively. His research interests include context awareness, data communication in MANET and WSN, and smart grids. His recent research work focuses on intelligent agriculture and meteorological observation systems based on WSN.

Wanchun Dou is currently a Full Professor with the State Key Laboratory for Novel Software Technology, Nanjing University. His research interests include big data, cloud computing, and service computing.

Jingyan Jiang received the Ph.D. degree in computer science from Jilin University, China, in 2020. From 2020 to 2022, she was a Post-Doctoral Research Fellow with Tsinghua University Shenzhen Graduate School. She is currently an Assistant Professor with Shenzhen Technology University, Shenzhen, China. Her research interests include edge intelligence and federated learning.