Network Resource Optimization in Emerging Vehicular Edge Computing

その他（別言語等）のタイトル	車両のエッジコンピューティングにおけるネットワークリソースの最適化手法
著者	PENG Xiting
学位名	博士（工学）
学位の種別	課程博士
報告番号	甲第456号
研究科・専攻	工学専攻
学位授与年月日	2020-09-24
URL	http://doi.org/10.15118/00010336

Network Resource Optimization in Emerging Vehicular Edge Computing

Xiting Peng

Department of Information and Electronic Engineering
Muroran Institute of Technology

This dissertation is submitted for the degree of
Doctor of Philosophy of Engineering

September 2020

Declaration

I hereby declare that this dissertation is my own work and effort and that it has not been submitted anywhere for any award. Wherever contributions of others are involved, every effort is made to indicate this clearly, with due reference to the literature, and acknowledgement of collaborative research and discussions.

Xiting Peng
September 2020

Acknowledgements

I came to the Emerging Networks and Systems (ENeS) at Muroran Institute of Technology in Japan three years ago and I have experienced the hardest and most memorable three years in my life. I have met a lot of professors and friends here and all of you have accompanied me through a journey of my Ph.D. study. I thank you all that I have met here, no matter sad or happy, because these experiences taught me mature, saw I was growing up every step. Growth, which includes the vulnerability of the soul is.

Of course, first I would like to express my gratitude to my supervisor, Prof. Mianxiong Dong. In the past three years, thank you for your support and guidance in my scientific research. You have taught me a lot of knowledge that I will use all my life. My grateful sentiments come from the bottom of my heart. In addition, I also want to thank the two other teachers in my laboratory, Prof. Kaoru Ota and Prof. He Li who have given great support to my scientific research.

Besides my supervisor, I really appreciate my dissertation committee. Your insightful comments and encouragement are of great significance to me.

I sincerely thank the China Scholarship Council (CSC) for my financial support, so I am fully committed to scientific research. I also want to thank to the consuls' care in Consulate-General of the People's Republic of China in Sapporo. In particular, Consul Jue Wang of the Education Group, you made us, the international students studying alone in Japan, feel the care of our motherland.

I am full of gratitude to my friend Lu Zhou. During the most difficult time in my scientific research, it is that you selflessly helped and gave me many key suggestions to make my research go smoothly. When I encountered difficulties and frustrations in my research, it was you who always encouraged and supported me. Without your unselfish dedication could I achieve no results today. True friendship always foresees the needs of the other, rather than announcing what you need.

It is hard to find words to express the gratitude to my parents, you are my strongest psychological support. Without your support and encouragement, it would be very difficult for me to complete the Ph.D. It is that you have supported my whole process to study in Japan.

Last but not the least, all my love and thanks to my boyfriend Xiaoyu Zhang. Without your accompany and encouragement, it would be difficult for me to persist. It is also your strong belief to come to Japan to accompany me so that I always feel your love and warmth for me. Without you accompany, it will be very difficult for me to face all the difficulties in these three years. It has left us with many unforgettable moments for many sleepless nights we were working together before deadlines. There is still a lot to say, so stay in our future.

Xiting Peng
September 2020

Abstract

With the development of the Internet of Vehicles (IoV), more and more vehicular services are coming to people's daily life, including intelligent transportation services and infotainment services. While the emergence of numerous services could provide innovative and convenient services for drivers, how to process large-scale data effectively still needs in-depth research in the IoV scenario. Different from traditional networks, the vehicular network has poor-quality wireless links which may lead to poor communication quality. Therefore, moving data to the cloud for processing is not feasible in the vehicular network. Vehicular edge computing network (VECN) is a promising way to provide fast task processing services for vehicles by offloading these tasks to edge nodes close to vehicles. There are two challenges in the VECN architecture: one is that vehicular edge nodes always have limited computing resources. Therefore, lightweight algorithms need to be deployed to promote the development of VECN. And the other is that some edge nodes and vehicles do not belong to the same party. Therefore, how to conduct fair trade between them for providing reasonable resource allocation is an important issue. In this dissertation, we firstly propose the broad learning based lightweight traffic analysis system in the vehicular edge computing network. Secondly, a multi-attribute based double auction mechanism is designed to solve the problems of fair resource allocation in the VECN. Then we propose an online auction mechanism to satisfy the dynamics of users in the VECN scenario, which also considers the non-price attributes when constructing the matching between buyers and sellers. Finally, we conduct multiple experiments to verify the proposed schemes.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Background

1.1.1 The Evolution of Internet of Vehicles

The Internet of Vehicles (IoV) as the core of components of intelligent transportation integrated information platform, has become an emerging paradigm benefit from recent advancements in vehicular communications and networking. Three networks are combined in the IoV shown as Fig. 1.1: intra-vehicle network, inter-vehicle network, and vehicular mobile Internet. Intra-vehicle network refers to the communication terminal with communication between the devices in the vehicles, and between the terminal and vehicles, which belongs to the end system of IoV, that is, the intelligence sensor of the vehicle. Inter-vehicle network refers to the interconnected communication terminal between vehicles and vehicles, vehicles and roads, vehicles and people, which belongs to the management system of IoV. Inter-vehicle network is mainly responsible for the communication and roaming between the vehicle self-organizing network and various heterogeneous network. Vehicle mobile internet is a cloud-based vehicle operation information platform, which belongs to the cloud system of IoV and provides vehicle services, including ITS, auto service, car rental, mobile internet. Vehicle mobile internet is mainly used for data collection, calculation, monitoring and management of vehicles.

With agreed communication protocols and data interaction standards, such as IEEE 802.11p WAVE standard and potentially cellular technologies, the IoV usually can be seen as a large-scale distributed system through which an interconnection can be made between the things, vehicles and environments to exchange the data and information. Therefore, IoV plays a very important role in intelligent transportation systems such as managing

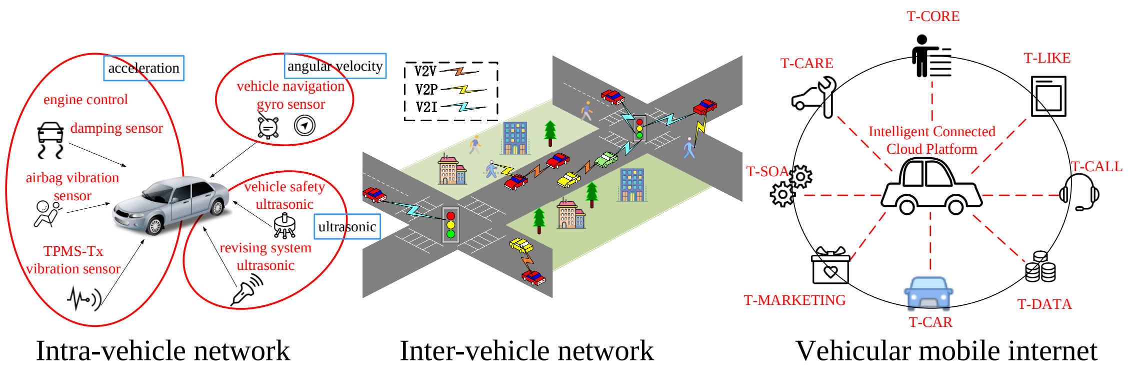


Fig. 1.1 Three Networks of IoV

traffic, developing intelligent dynamic information communication between the vehicles, road accident prevention, and road safety management.

The evolution of the Internet of Vehicles can be roughly divided into three stages, as shown in Fig. 1.2. The first stage is to provide vehicular information services through 2G, 3G, 4G, and other technologies, such as vehicular navigation, vehicle diagnosis, and infotainment. The second stage is to increase assisted driving applications through technologies like LTE-V2X, such as real-time traffic information sharing, traffic flow control, road congestion analysis, and accident warning. The third stage, autonomous driving applications and intelligent transportation system (ITS) will be gradually introduced through 5G technology, such as remote driving, formation driving, environment awareness, and high-precision map download.

With regard to the demand for the future autonomous driving phase, 3GPP has already started to standardize 5G V2X in 2016. 3GPP SA's V2X requirements standardization is divided into two stages: the first stage is the basic task requirements for LTE-V2X for assisted driving applications; the second stage is the 5G-V2X enhanced task requirements for autonomous driving applications.The four types of V2X applications defined by 3GPP SAI include: V2V, V2I, V2N, V2P.

- Vehicle to Vehicle (V2V): V2V application information is exchanged between vehicles close to each other. V2V application information, that contains vehicles' location and movement status, is used to provide safety precaution and improve driving safety.
- Vehicle to Infrastructure (V2I): V2I application information is exchanged between the vehicles and Road Side Unit (RSU). V2I application information is mainly used to provide real-time and effective traffic environment information for vehicles, such as traffic light information, traffic sign information, and road construction information.

- Vehicle to Network (V2N): V2N application information is exchanged between the vehicles and application servers that could be located on the cloud. V2N applications can supply could-based service access for vehicles and achieve greater coverage of IoV services to provide better services like improved transportation efficiency.

- Vehicle to Pedestrian (V2P): V2P application information is exchanged between vehicles and pedestrians close to each other. V2P applications can provide safety reminders for pedestrians or vehicles. Note that compared with V2V applications, V2P is more focused on protecting the safety of pedestrians.

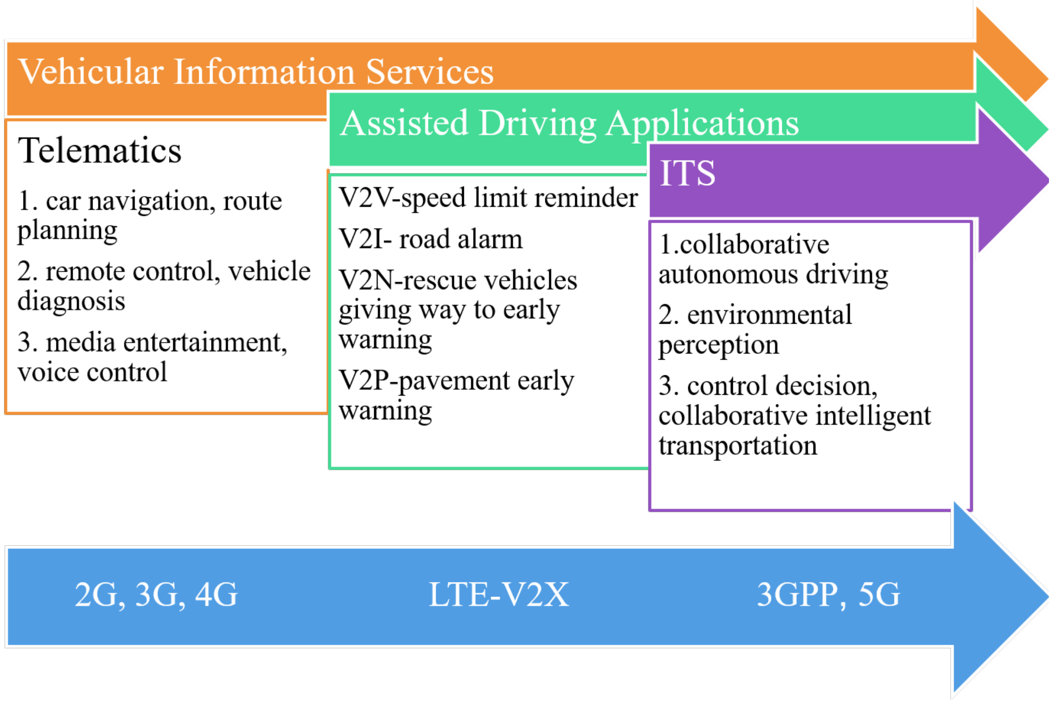

Vehicular Information Services
Telematics
1. car navigation, route planning
2. remote control, vehicle diagnosis
3. media entertainment, voice control
Assisted Driving Applications
V2V-speed limit reminder
V2I- road alarm
V2N-rescue vehicles giving way to early warning
V2P-pavement early warning
ITS
1.collaborative autonomous driving
2. environmental perception
3. control decision, collaborative intelligent transportation
2G, 3G, 4G
LTE-V2X
3GPP, 5G

Fig. 1.2 The Evolution of IoV

Taking into account the demand for higher levels of autonomous driving in the future, 3GPP SAI continues to define 5G-oriented enhanced V2X task requirements, which can be summarized into four major scenarios: formation driving, advanced driving, extended sensors, and remote driving.

- Formation driving: Formation driving is to realize the automatic formation driving of multiple vehicles. All vehicles which are in formation receive the data periodically sent by the head car for the formation operation. Through the information interaction between vehicles, the distance between vehicles could be very small like several meters or even tens of centimeters, to reduce the fuel consumption of the rear vehicles. Besides, with the help of formation driving, the rear vehicles can achieve follow-up automatic driving.

- Advanced driving: Advanced driving is to realize semi-automatic or fully automatic driving. Each vehicle or RSU shares the data obtained through sensors with surrounding vehicles, to allow vehicles to coordinate their movement trajectories or operations. Besides, each vehicle shares its driving purposes with surrounding vehicles that can improve driving safety and traffic efficiency.
- Extended sensor: The extended sensor is to realize the exchange of data collected by local sensors or real-time video data between vehicles, RSU, pedestrian equipment, and the V2X application server. The interaction of these data is equivalent to expanding the detection range of vehicle sensors so that a vehicle can enhance its ability to perceive its environment and enables the vehicle to have a more comprehensive understanding of the surrounding conditions.
- Remote driving: Remote driving is to realize the driver or driving program to drive the vehicle remotely. It can be used to deal with some limited local driving conditions like passengers cannot drive the vehicle or the vehicle is in a hazardous environment. It can also be used in public transportation and other scenes with a relatively fixed driving track.

The above scenarios involve the transmission of V2X tasks such as raw sensor data sharing, the collaboration between vehicles, and operation confirmation. The applications in V2X have different requirements for data rate, reliability, delay, communication range, and driving speed. Therefore, 3GPP's requirements of future IoV can be summarized as follows: 1) massive data processing can reach 600 ~3000 messages per minute 2) fast response can reach less than 25ms 3) service success rate can reach 90% ~99.99% [49]. However, a large number of applications will generate massive data in a short time in vehicular network. For example, in some driver assistance systems, vehicles need to process massive data generated from vehicle sensors (camera, radar, and so on), which is difficult to do alone by resource-limited vehicles. Considering the poor communication quality, it is difficult to move these data to a remote server for real-time analysis. Edge/Fog computing is a promising way to achieve the requirements by offloading this task to the edge/fog node close to user devices. And the edge/fog node performs data processing tasks through local processing, which can provide the real-time services and reduce the burden of the server, thereby achieving high bandwidth and high reliability. Therefore, a new paradigm, edge/fog computing [47], is introduced into the vehicular network to provide real-time data processing through moving these tasks (e.g., anomaly detection of in-vehicle network) to surrounding edge/fog nodes, which is coined as Vehicular Edge/Fog Computing Network (VECN/VFCN). In the next subsection, we will give the origin, evolution, and architecture of edge/fog computing and the architecture of VECN/VFCN.

1.1.2 Architecture

Edge Computing

Edge computing can be traced back to the content delivery network (CDN) proposed byAkamai in 1998. CDN is an Internet-based cache network that relies on cache servers deployed in various locations. Through the load balancing, content distribution, scheduling, and other functional modules of the central platform, it directs user access to the nearest cache server. This will reduce network congestion and improve user access response speed and hit rate. CDN emphasizes the backup and cache of data, while the basic idea of edge computing is function cache. In 2009, Satyanarayananet al. [51] proposed the concept of Cloudlet. Cloudlet is a trusted and resource-rich host deployed at the edge of the network, connected to the Internet, and accessible by mobile devices to serve it. At the same time, edge computing emphasizes downlink, meaning that the functions on the cloud server are transferred to the edge server to reduce bandwidth and latency. In the background of the Internet of Everything, edge data has exploded developed. In order to solve the problem of computing load and data transmission bandwidth in the process of data transmission, calculation and storage, researchers have begun to explore adding data processing functions near the edge of data producers, that is, the uplink of the Internet of Everything service. Mobile edge computing(MEC), edge computing, and cloud-sea computing are typical examples.

Mobile edge computing refers to a new network structure that has become a standardized technology. It provides information technology services and cloud computing capabilities within the range of wireless access networks close to mobile users. Because mobile edge computing is located in the wireless access network and is close to mobile users, it can achieve lower latency and higher bandwidth to improve service quality and user experience. Mobile edge computing emphasizes the establishment of edge servers between cloud computing centers and edge computing devices and completes terminal data calculation tasks on edge servers. However, mobile edge terminal devices are generally considered to have no competing capabilities, and the terminal devices in the edge computing model have strong computing capabilities. Therefore, mobile edge computing as part of the edge computing architecture is similar to the architecture and layer of an edge computing server. Cisco

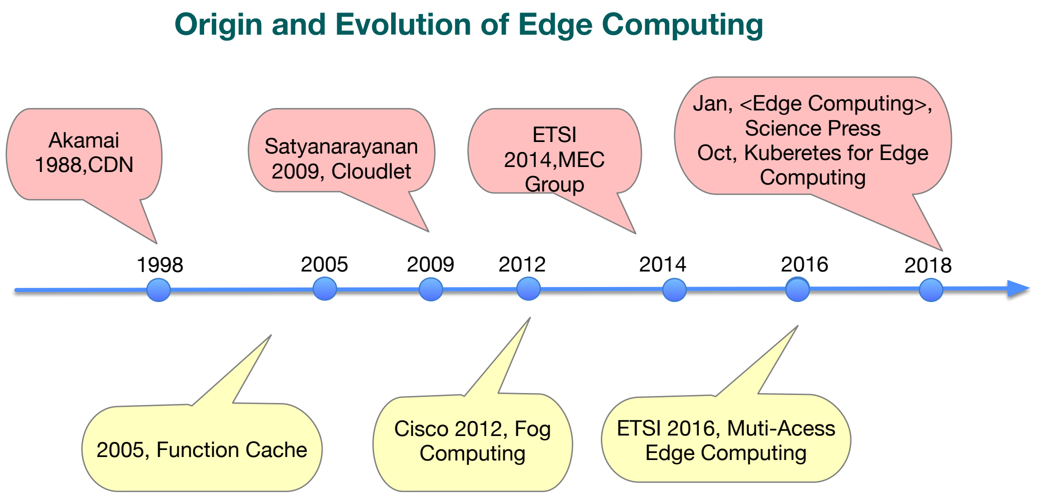

Origin and Evolution of Edge Computing
Akamai 1988,CDN
Satyanarayanan 2009, Cloudlet
ETSI 2014,MEC Group
Jan, <Edge Computing>, Science Press Oct, Kuberetes for Edge Computing
1998
2005
2009
2012
2014
2016
2018
2005, Function Cache
Cisco 2012, Fog Computing
ETSI 2016, Muti-Acess Edge Computing

Fig. 1.3 The Evolution of Edge Computing

[6] introduced fog computing in 2012 and defined fog computing as a highly virtualized computing platform that migrates cloud computing center tasks to network edge devices. By reducing the number of communications between cloud computing centers and mobile users, fog computing can alleviate the bandwidth load and energy pressure of the backbone link. Fog computing and edge computing are very similar, but fog computing focuses on the issue of distributed resource sharing between infrastructures. In addition to infrastructure, edge computing also focuses on edge devices, including computing, network, and management of storage resources, and the cooperation between edge and cloud.

The European Telecommunications Standards Institute (ETSI) established the Mobile Edge Computing Industry Specification Group in 2014 and officially announced the promotion of mobile edge computing standardization. The basic idea is to migrate the cloud computing platform from the inside of the mobile core network to the edge of the mobile access network to achieve the flexible use of computing and storage resources. In 2016, ETSI extended the concept of MEC to Multi-Access Edge Computing, further extending edge computing from telecommunications cellular networks to other wireless access networks(such as WiFi). By 2018, the traffic from games, videos, and streaming-based web content will account for 84 percent of IP traffic, which requires mobile networks to provide a better quality of experience. With edge-cloud architecture, network latency can be reduced by 50 percent. According to Gartner's report, the devices of IoT will be more than 20.8 billion by the end of 2020. This shows that the focus of research has gradually towards from cloud computing to edge computing. The whole origin and evolution of edge computing are as shown in Fig. 1.3.

Fog Computing

The concepts of fog computing and edge computing are very similar, just different companies named them differently. By adding a fog computing layer between the cloud server and the terminal device, the network device is used to provide computing, storage, and network communication services, making the system bandwidth speed of the network and the response speed of the system have been greatly improved. The term edge computing and fog computing seem interchangeable, and for a fact, they do share some key similarities:

- Both edge and fog computing systems shift processing of data closer to the source of data generation.
- In order to decrease latency, both of them are trying to reduce the amount of data sent to the cloud.
- Through the above strategies, both can improve the system response time, especially in remote mission-critical applications. By bringing the data processing closer to the source, companies are also improving the security as they don't need to send all the data across the public internet.

Moreover, we could regard two concepts of fog computing and edge computing as interchangeable in academia. For example, the authors in [58] say that mobile edge computing (MEC), interchangeably known as fog computing (originating from the cloudlet concept [52]), represents a vital solution to these limitations. Moreover, Yi *et al.* [67] viewed fog computing as edge computing, can address those problems by providing elastic resources and services to end users at the edge of network, while cloud computing are more about providing resources distributed in the core network. Edge computing is interchangeable with fog computing in [17], but edge computing focus more toward the things side, while fog computing focus more on the infrastructure side.

Therefore, we do not distinguish between edge and fog in this dissertation and we use the edge uniformly in the following chapters.

The Architecture of Edge/Fog Computing

As users have higher requirements for lower latency, lower bandwidth consumption, and higher security and privacy, cloud services cannot meet their requirements. Therefore, edge/fog computing is proposed as a promising way to assist cloud to achieve these requirements [53, 37]. Generally speaking, edge/fog computing means offloading some tasks that are previously carried out on the cloud to the devices that are close to the user side [54].

In some classic edge/fog computing scenarios, such as Vehicular Network, Smart Home, Body Things, and other IoT scenarios, the edge/fog nodes refer to the Road Side Unit (RSU), Hub (Gateway), smartphone, and WiFi access point, which always have little computing and storage capacities. By combining edge/fog computing and cloud computing, service providers not only can meet the requirements of large-scale data processing, but also meet the needs of real-time [35]. As shown in Fig.1.4, the classic edge/fog computing based architecture includes following layers [30]: Cloud layer, Edge/Fog layer, and User layer. In this architecture, the users send the data to the edge/fog node, rather than the cloud. And the edge/fog node performs data processing tasks, which can provide the real-time services and reduce the burden of the server. After completing the data processing, the edge/fog node sends the results to the cloud.

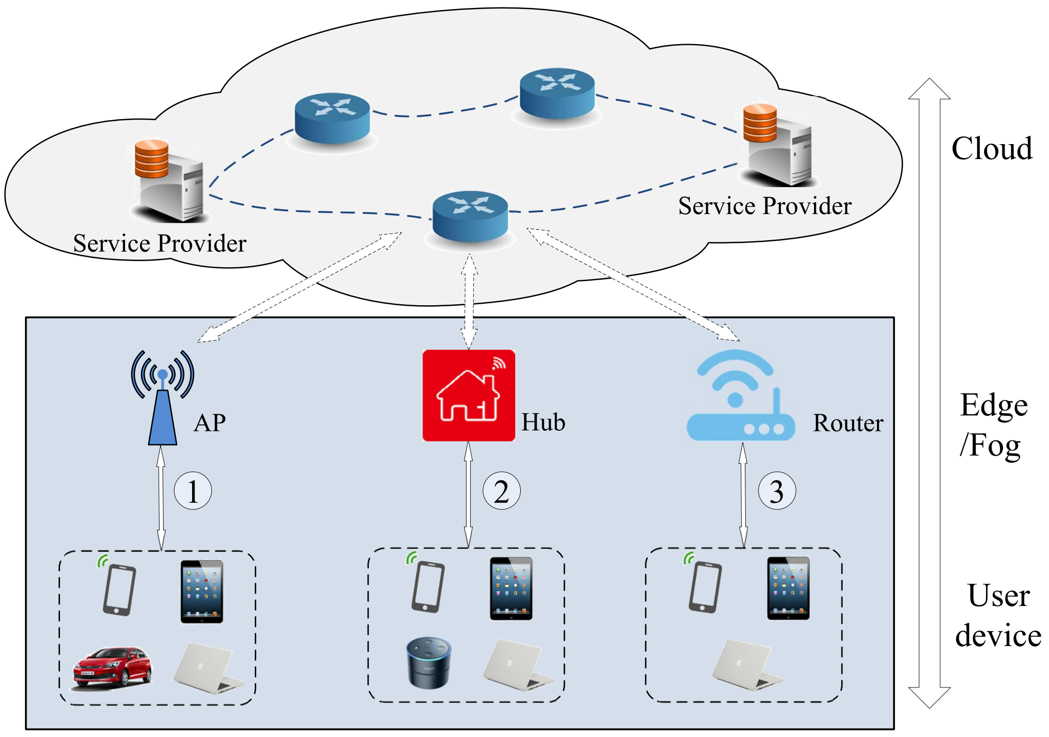


Fig. 1.4 The Architecture of the Edge/Fog Computing

The algorithms and systems we designed in this dissertation can be deployed both in the edge layer and the fog layer. Therefore, we no longer distinguish between edge computing and fog computing in the following parts. As mentioned above, we use edge computing uniformly.

Vehicular Edge Computing Network (VECN)

With the development of Internet of Vehicles, more and more vehicular applications are beginning to enter the lives of people, including the safety and entertainment related applications. The vehicular network is regarded as an efficient and useful way to provide innovative services and improve road capability by connecting vehicles and resource-rich processing servers. According to Gartner's market forecast, there will be more than 250 million connected vehicles by the end of 2020. By then, IoT (Internet of Things) service providers will benefit from this emerging market with annual revenues exceeding $300 billion, and global vehicular networking market will grow to 7.1 trillion US dollars [2].

However, a large number of applications will generate massive data in a short time. For example, in some driver assistance systems, vehicles need to process massive data generated from vehicle sensors (camera, radar, and so on), which is difficult to do alone by resource-limited vehicles. Considering the poor communication quality, it is difficult to move these data to a remote server for real-time analysis. Due to the property the real-time services of the edge computing we introduced above, we applied edge computing to the IoV to solve limitations in conventional vehicular networks. Therefore, a new paradigm, edge computing, is introduced into the vehicular network to provide real-time services for vehicles by offloading these tasks (e.g., anomaly detection of in-vehicle network) to surrounding edge nodes, which is coined as vehicular edge computing network (VECN).

Vehicular edge computing network (VECN) is a promising way to provide fast task processing services for vehicles by offloading these tasks to edge nodes close to vehicles [40]. VECN introduces the idea of the edge computing paradigm [31] into traditional vehicular network, as a supplement to cloud computing. The architecture of VECN is shown in Fig. 1.5. The whole network can be divided into three layers from top to bottom: cloud computing layer, edge computing layer, and vehicular computing layer. The cloud computing layer as remote cloud server at the core of the network is responsible for massive network data processing and macro analysis, and plays a role in decision-making at the network level; the middle layer is the edge computing layer, which is mainly responsible for real-time data fusion and localization processing, and undertakes decision-making tasks at the regional level; the bottom layer is the vehicular computing layer, which is mainly responsible for data collection and pre-processing, as a decision at the vehicular level.

1.2 Challenges

In this new architecture, we have encountered some challenges, as shown in Fig. 1.6. On one hand, vehicular edge nodes always have limited computing resources. On the other hand,

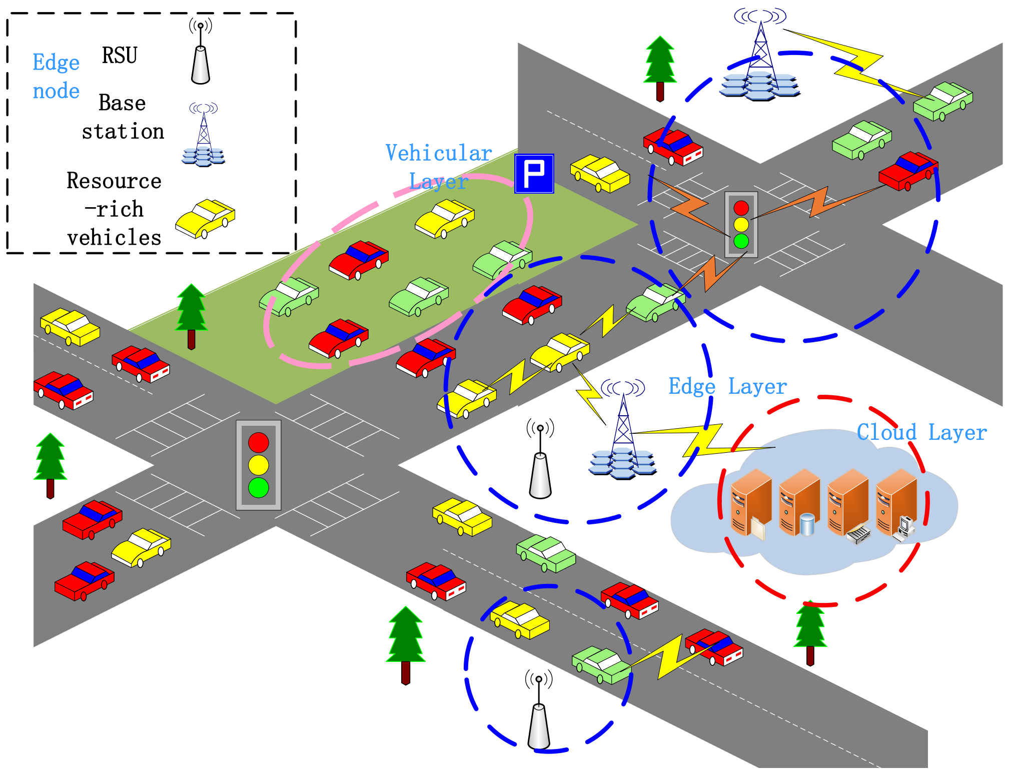


Fig. 1.5 The Architecture of VECN

some edge nodes and vehicles do not belong to the same part. To cope with these challenges and improve resource utilization in VECN, we need to develop a lightweight algorithm and conduct fair trade between vehicles and edge nodes for providing the reasonable resource allocation scheme in VECN.

1.2.1 Broad Learning Based Lightweight Traffic Analysis System in VECN

Current traffic classification models are mainly based on traditional machine learning, including SVM, random forest and decision tree. However, these traditional methods cannot be directly used at edge computing based traffic analysis architecture due to the low computing and storage capabilities of edge nodes. In order to maintain the classification accuracy, it is better to regularly update the trained model. Traditional methods require retraining all data when performing model updates, which consume large amounts of computing power. Usually, the edge node has these characteristics: poor computing power and low storage capacity [53], which cannot support frequent retraining of all data. To address the above challenges, we use

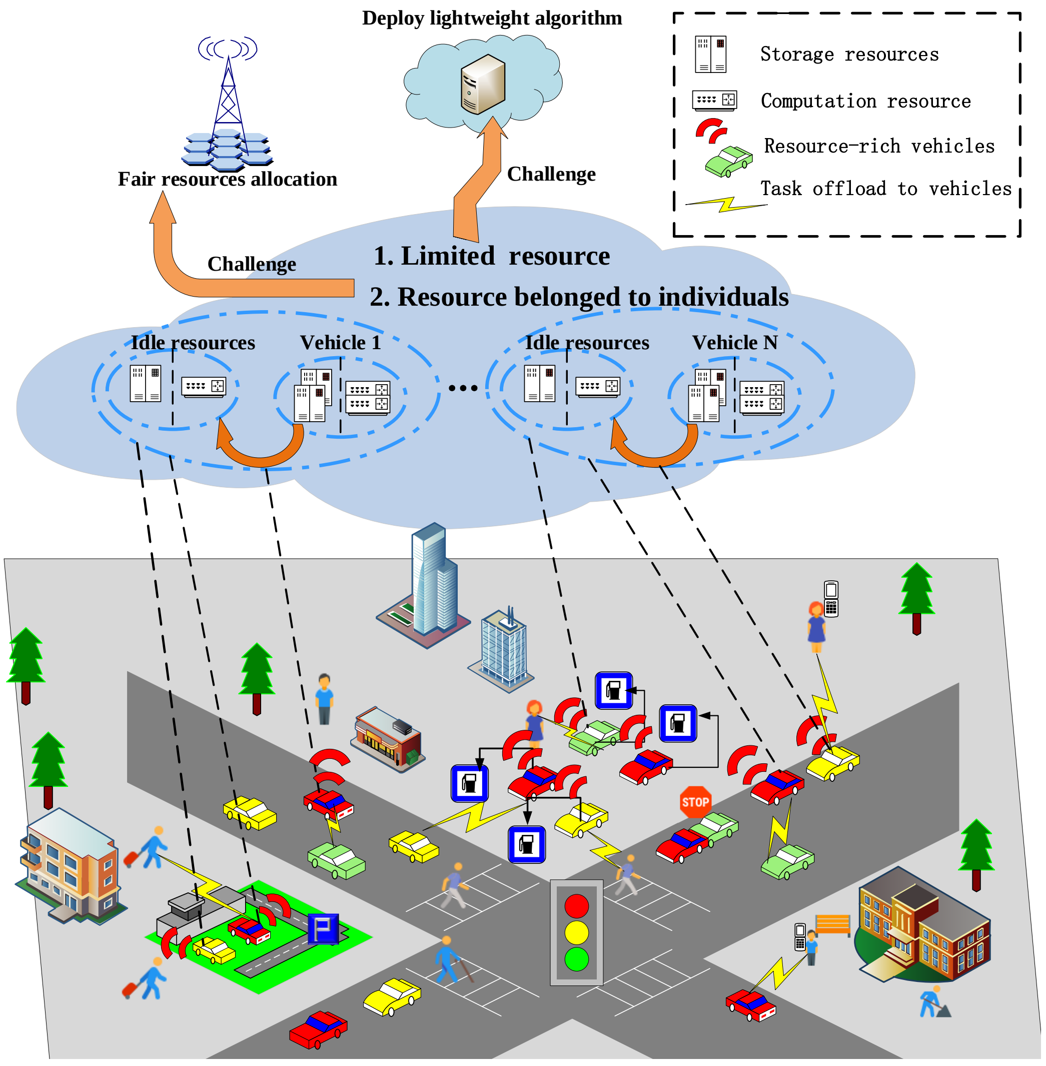


Fig. 1.6 The Challenges in VECN

a novel and lightweight neural network structure, broad learning system (BLS) [11], which has the faster training speed due to its flat network structure. More importantly, BLS can use incremental learning to constantly update the trained model when new data enters, and no retraining process is needed.

In Chapter 3, we propose a broad learning based lightweight traffic analysis system in VECN. In one hand, edge computing can provide a distributed architecture, which can save the precious bandwidth resources and provide the safer service environment by offloading the analysis task to the network edge. In the other hand, we use broad learning system to incrementally train the traffic data, which is more suitable for the edge computing because it has fast training speed and can support incremental learning. The proposed system is composed of two major modules: basic training on the cloud and incremental training (model updating) on the edge node. Firstly, we use some traffic data to train a basic model and send the basic model to the edge node. Secondly, when the accuracy of the trained model is not enough to provide better services, model updating will be executed on the edge node by the incremental way. We implement the edge computing based traffic analysis system on the Raspberry Pi, and the experimental results show that our method has the faster training speed compared with Convolutional Neural Network (CNN). Except for the initial stage, our method does not need to upload the traffic data to the cloud.

1.2.2 Multi-attribute Based Static/Offline Double Auction in VECN

The existing edge-related auction mechanisms could not be directly used in VECN, which only consider the price information. Due to poor-quality wireless links, vehicular network has a large delay when transmitting large-scale data [77]. One edge node is difficult to provide large-scale data processing services for vehicles who are far away. Therefore, location information should be considered when determining the matching in the VECN auction. Moreover, due to different types of tasks and different computing capabilities of edge nodes, vehicles have different preferences over edge nodes. For example, vehicles want to choose edge nodes with higher capabilities when processing safety-related tasks. Therefore, other non-price attributes, such as reputation and computing capability, also need to be considered in VECN auction.

In Chapter 4, we design a multi-attribute based static/offline double auction mechanism in VECN scenario. The proposed auction mechanism not only considers the price but also considers non-price attributes when determining the winners, which could construct more reasonable matching between edge nodes and vehicles. In addition, our auction mechanism could satisfy the following economic properties: computational efficiency, individual rationality, budget balance, and truthfulness [25, 66]. To verify our auction mechanism, we simulate the VECN scenario using VISSIM, an open source framework for running vehicle network simulation. Then we implement the proposed auction system by JAVA, and verify the effectiveness and efficiency of the double auction mechanism by the driving data extracted from VISSIM.

1.2.3 Dynamic/Online Resources Allocation in VECN

Since edge nodes and vehicles have their own interests, a double auction scheme which could consider interests of both parties is needed in vehicular edge computing network scenario. During the double auction process, vehicles submit their bids and edge nodes give their asks to trusted auctioneer. Then this auctioneer executes winner determination and pricing algorithms to determine the final winners and price. Since edge nodes and vehicles do not know the bids/asks of other vehicles and edge nodes, the double auction can give a fair trading platform.

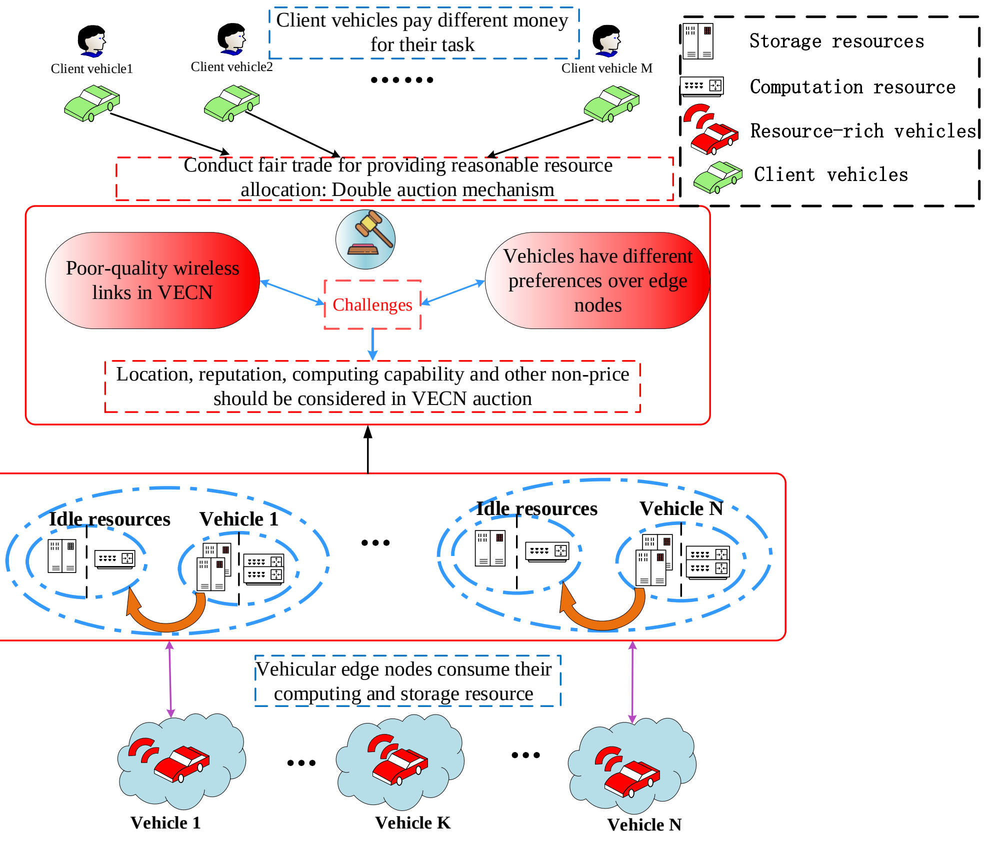

Client vehicles pay different money for their task
Client vehicle1
Client vehicle2
Client vehicle M
Storage resources
Computation resource
Resource-rich vehicles
Client vehicles
Conduct fair trade for providing reasonable resource allocation: Double auction mechanism
Poor-quality wireless links in VECN
Challenges
Vehicles have different preferences over edge nodes
Location, reputation, computing capability and other non-price should be considered in VECN auction
Idle resources
Vehicle 1
Idle resources
Vehicle N
Vehicular edge nodes consume their computing and storage resource
Vehicle 1
Vehicle K
Vehicle N

Fig. 1.7 The Challenges of Resource Allocation in VECN

There are already some works study edge-related auction schemes. In [36], a reverse auction scheme was studied to encourage vehicles to share resources. In [24, 23], the authors considered deploying edge computing into mobile blockchain and proposed an auction mechanism for providing a fair trading environment. In [57], the authors studied the resource allocation in industrial IoT. They assumed each edge node can serve multiple mobile devices, and proposed two double auction schemes.

Considering the dynamic flexibility in VECN, buyers and sellers can join or leave the auction mechanism at any time according to their own wishes, which will not be controlled by the auctioneer. Therefore, in the vehicular edge computing network scenario, the auctioneer needs to make decisions in a more dynamic environment, which could be achieved by the online auction mechanism.

Moreover, these auction schemes only use the price for constructing the matching between buyers and sellers. However, in VECN scenario, the communication quality could be poor because of the mobility of vehicles, which means long-distance communication will have a large delay. Even if there existing a matching between a vehicle and an edge node according to existing edge-related auction mechanisms, the winning edge node still cannot provide services for the winning vehicle if there is a long distance between them (e.g., cannot complete the transmission of massive data between vehicles and edge nodes in time). Therefore, location is an important factor to construct reasonable matching in VECN auction scenario. In addition to location, other non-price attributes are also important constraints in VECN auction scenario (as discussed in Section 4.3).

In Chapter 5, to cope with these challenges as shown in Fig.1.7, we design a online resource auction scheme for an vehicular edge computing network with non-price attributes, which is in line with a dynamic situation that vehicular edge nodes and client vehicles can join or leave the auction system at any time. Moreover, in addition to the bids/asks, we also consider the non-price attributes when constructing the matching between buyers and sellers. We simulate VECN using a vehicular network simulator for verifying our work. Then we evaluate our work from the perspective of running time, charges/rewards, and utilities of users. Experimental result shows that our scheme could meet properties of computational efficiency, individual rationality, budget-balance, and truthfulness.

1.3 Organization

The rest of this dissertation is organized as follows. We briefly introduce the edge computing based traffic analysis system using broad learning in Chapter 2. This chapter also includes broad learning algorithm and incremental learning algorithm to analyze traffic data, which has fast training speed. We discuss a multi-attribute based static/offline double auction mechanism in Chapter 3, which considers both the price and non-price attributes for constructing reasonable matching. The proposed auction mechanism could satisfy computational efficiency, individual rationality, budget balance, and truthfulness. Chapter 4 shows the importance of non-price attributes in VECN auction scenario, and the proposed online double auction mechanism which could satisfy the dynamics of users in the auction system. We conclude this dissertation in Chapter 5.

Chapter 2

Broad Learning Based Lightweight Traffic Analysis System in VECN

2.1 Motivation

With the development of the next-generation communication technologies (e.g., the fifth generation (5G) wireless systems), people's demand for high-speed data services is gradually satisfied [48, 43], including high-quality wireless video streaming, massive Internet of Vehicles (IoV) services, and real-time VR/AR applications. However, the use of these high-speed data services and applications in IoV also makes traffic denser and traffic types more diverse [19], which presents new challenges in designing a real-time traffic analysis architecture that can process large amounts of traffic data and an accurate analysis method that suitable for a wide variety of traffic types.

Accurate identification of network traffic types is the basis for providing better services. By executing accurate traffic analysis, service providers can monitor the operation of the entire network [27]. They can realize users' behaviors by analyzing users' traffic, which can provide personalized services. Moreover, in the complex network environment, traffic analysis can also ensure the security of the network. It can effectively deal with many security issues, such as intrusion detection. Therefore, how to provide an accurate traffic classification method is crucial.

Current methods of traffic analysis are executed at the cloud, which needs to upload all users' traffic to the cloud [50]. However, many applications in the fifth generation (5G) wireless systems, such as video applications, can generate massive traffic within a short time period, which will consume lots of precious bandwidth resources if all traffic data is uploaded to the server. Therefore, current cloud-based traffic analysis system does not apply to traffic-intensive applications in the fifth generation (5G) wireless systems. Moreover, service providers just need the classification results for understanding the network operation and providing better services, rather than all traffic data. Therefore, it is better to propose a novel distributed traffic analysis architecture, which only needs to send the classification results to service providers.

Current traffic classification models are mainly based on traditional machine learning, including SVM, random forest and decision tree. However, these traditional methods cannot be directly used at edge computing based traffic analysis architecture due to the low computing and storage capabilities of edge nodes. In order to maintain the classification accuracy, it is better to regularly update the trained model. Traditional methods require retraining all data when performing model updates, which consume large amounts of computing power. Usually, the edge node has these characteristics: poor computing power and low storage capacity, which cannot support frequent retraining of all data. To address the above challenges, we use a novel and lightweight neural network structure, broad learning system (BLS), which has the faster training speed due to its flat network structure. More importantly, BLS can use incremental learning to constantly update the trained model when new data enters, and no retraining process is needed.

In this chapter, we design a novel broad learning based lightweight traffic analysis system. In one hand, vehicular edge computing can provide a distributed architecture, which can save the precious bandwidth resources and provide the safer service environment by offloading the analysis task to the network edge. In the other hand, we use broad learning system to incrementally train the traffic data, which is more suitable for the vehicular edge computing because it has fast training speed and can support incremental learning. The proposed system is composed of two major modules: basic training on the cloud and incremental training (model updating) on the edge node. Firstly, we use some traffic data to train a basic model and send the basic model to the edge node. Secondly, when the accuracy of the trained model is not enough to provide better services, model updating will be executed on the edge node by the incremental way. We implement the edge computing based traffic analysis system on the Raspberry Pi, and the experimental results show that our method has the faster training speed compared with Convolutional Neural Network (CNN). Except for the initial stage, our method does not need to upload the traffic data to the cloud.

The main contributions of this chapter are summarized as follows:

- We propose a distributed traffic analysis architecture, vehicular edge computing based traffic analysis system, which can save the bandwidth resources and protect users' privacy from being inferred by malicious service providers.

- In order to solve the problem of poor computing power and low storage capacities of edge nodes, we use a novel and lightweight neural network structure (broad learning system) to analyze traffic data, which has fast training speed and can support incremental learning.
- We implement our system on the Raspberry Pi and perform comprehensive experiments using real-world dataset to validate its performance.

The rest of this chapter is organized as follows. In section 2.2, we present related works about the traffic analysis and the broad learning system. Section 2.3 presents the proposed edge computing based traffic analysis system. We discuss the experimental evaluation in section 2.4. Finally, we conclude this section in section 2.5.

2.2 Related Work and Preliminary Knowledge

In this section, we will introduce related work about traffic analysis and the technologies that we used in detail, including the broad learning system and the dynamic stepwise updating algorithm.

2.2.1 Traffic analysis

Nowadays, traffic analysis has received wide attention from both the academia and the industry [38]. Researchers have proposed many methods for traffic classification, which can be classified as follows: port-based identification, deep packet inspection, and machine learning. However, the first two technologies have strong limitations, which has the low accuracy when identifying varied traffic types [10]. Therefore, machine learning based methods are widely adopted by the academia and the industry.

Machine learning based traffic analysis methods have a wide range of applications in different fields, which is mainly based on traditional machine learning. In [33], the authors proposed a traffic analysis model based on S-SVM, which can be used to accurately identify a variety of network traffic data. In [62], random forest was used to identify apps by fingerprinting network traffic. Besides, many machine learning based methods are available for traffic classification. For instance, in [3], the authors used six common algorithms, such as Linear Regression, Decision Tree, and Multi-layer Perceptron, for malware traffic classification and compared their performances when confronted with real network data.

However, these methods are executed on the cloud, which needs to upload users' traffic to the server. In addition to wasting resources, uploading traffic to the cloud will also cause the leakage of users' privacy [14]. Therefore, in this study, we design a novel distributed system using edge computing to ensure that traffic data is analyzed at edge nodes which are trustworthy to users. Furthermore, edge node always has the poor computing power and low storage capacity, which cannot directly deploy traditional machine learning based methods because these methods need to retrain all data when performing model updating. Deep learning based traffic analysis methods [60] can save and update the trained model in an incremental way. However, due to the complexity of the deep structure, it costs more time. Therefore, we use a lightweight neural network structure, broad learning system, to train the traffic data on the edge nodes, which has fast training speed and do not need to retrain all data when regularly updating the trained model.

2.2.2 Broad Learning System

Broad learning system [11] is an effective network structure which does not consume much time to retrain the model. Broad learning system (BLS) is developed with fewer parameters and the training speed can be faster due to its flat network structure, which is composed of input layer, feature nodes/enhancement nodes, and output layer. Fig. 2.1 shows its standard structure: Firstly, the mapped features are extracted from original input for generating the feature nodes. Then through activation functions which can be either nonlinear or linear, the feature nodes are mapped into enhancement nodes. Finally, feature/enhancement nodes are used to generate the output through pseudoinverse.

We set I_X^m and K_X^n as m groups of feature nodes and n groups of enhancement nodes, where X represents the initial input. These nodes can be generated by the following way:

$$I_X^m = \Big[\varphi(XA_{e1} + \theta_{e1}), ..., \varphi(XA_{em} + \theta_{em})\Big] \tag{2.1}$$

$$K_X^n = \Big[\chi(I_X^m A_{h1} + \theta_{h1}), ..., \chi(I_X^m A_{hn} + \theta_{hn})\Big] \tag{2.2}$$

where $A_{ei}, \theta_{ei}, \varphi$ are the random weights, bias and activation function of feature nodes, and $A_{hj}, \theta_{hj}, \chi$ are corresponding parameters of enhancement nodes. According to these generated nodes, we can compute the weights $W_n^m = (I_X^m|K_X^n)^+Y$, where $(I_X^m|K_X^n)^+$ is the pseudoinverse of the matrix $(I_X^m|K_X^n)$ and Y is the label of the input data.

In traffic analysis scenarios, we need to update traffic classification model frequently. Introducing incremental learning into BLS is a good option which can avoid retaining all traffic data. When new traffic data comes into the trained model, the output-layer weights of the network can be updated without retraining the network model.

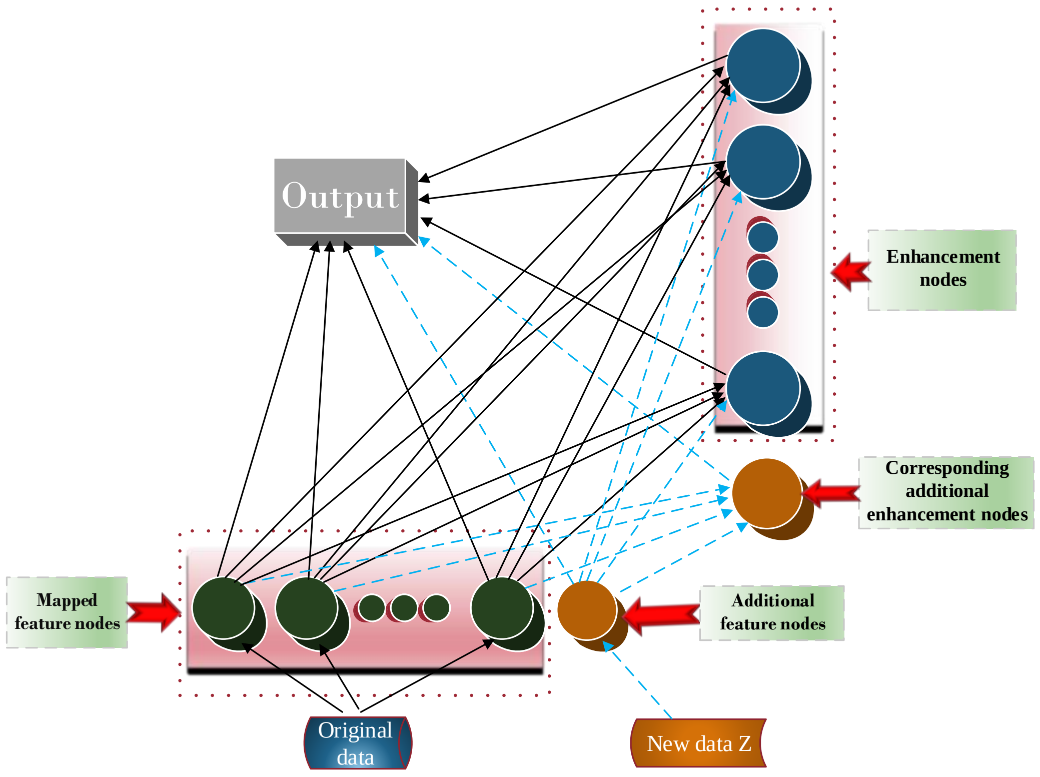


Fig. 2.1 The Architecture of the Broad Learning System

When new data Z inputs, the corresponding feature nodes I_Z^m and enhancement nodes K_Z^n could be calculated using the same way as the Equation (2.1) and (2.2). The new weights ${}^Z W_n^m$ of the BLS model can be updated by the dynamic stepwise updating algorithm based on the previous weights W_n^m:

$$ {}^Z W_n^m = \left[W_n^m + B(Y_z - (I_Z^m | K_Z^n) W_n^m) \right] \tag{2.3}$$

where Y_z is the label of the new data Z. Note that,

$$ B = \begin{cases} (C^T)^+, & \text{if } C \neq 0 \\ (I_X^m | K_X^n)^+ D (I + D^T D)^{-1}, & \text{if } C = 0 \end{cases} \tag{2.4}$$

where $C^T = (I_Z^m | K_Z^n) - D^T (I_X^m | K_X^n)$, and $D^T = (I_Z^m | K_Z^n)(I_X^m | K_X^n)^+$.

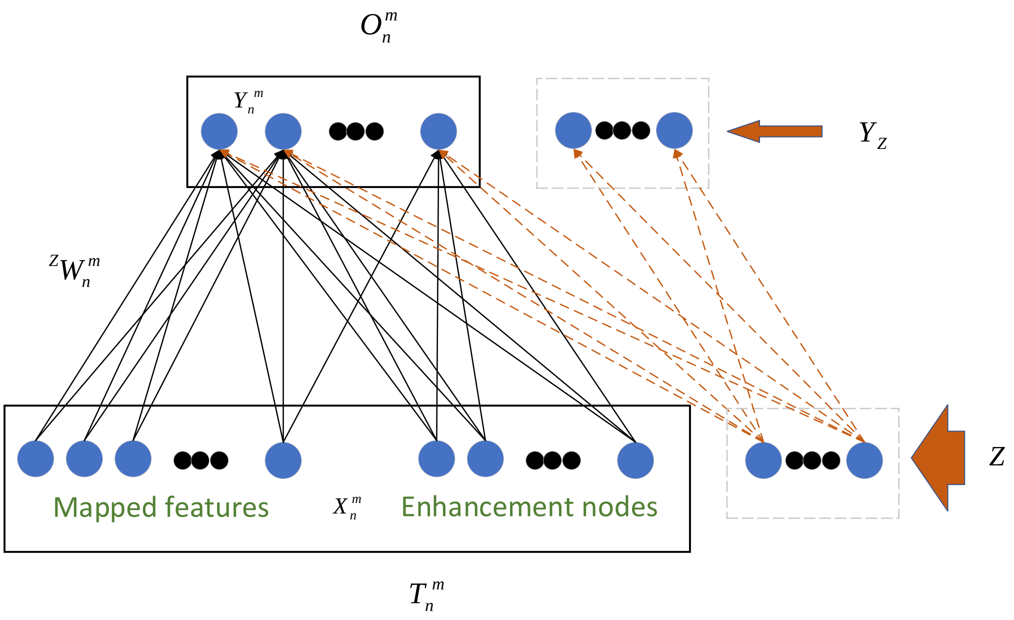


Fig. 2.2 Dynamic Stepwise Updating Algorithm

2.2.3 Dynamic Stepwise Updating Algorithm

Dynamic stepwise updating algorithm is designed to realize the fast updating of the weights of the trained model. With the help of the pseudoinverse of a partitioned matrix, updating the weights of the system on-the-fly can be achieved. We give a brief introduction to the dynamic stepwise updating algorithm by taking an example of adding new data to the trained model, as shown in Fig. 2.2. We set X_n^m as the m groups of feature mapping nodes and n groups of enhancement nodes of the original neural network, which is equal to $(I_X^m|K_X^n)$ as shown in Section 2.2.2. When the new data Z inputs, we can generate the new incremental features (feature/enhancement nodes) Z_n^m. Therefore, we can update the node matrix:

$$T_n^m = \begin{bmatrix} X_n^m \\ Z_n^m \end{bmatrix} \tag{2.5}$$

Correspondingly, we can get the pseudoinverse of the new node matrix T_n^m by $(X_n^m)^+$:

$$(T_n^m)^+ = \left[(X_n^m)^+ - BD^T \middle| B\right] \tag{2.6}$$

Similarly,

$$B = \begin{cases} (C^T)^+, & \text{if } C \neq 0 \\ (X_n^m)^+ D(I + D^T D)^{-1}, & \text{if } C = 0 \end{cases} \tag{2.7}$$

where $C^T = Z_n^m - D^T X_n^m$, and $D^T = Z_n^m (X_n^m)^+$.
As we know, the following equation holds (the new updating weights):

$$^zW_n^m = (T_n^m)^+ \cdot (O_n^m) \tag{2.8}$$

and

$$W_n^m = (X_n^m)^+ \cdot (Y_n^m) \tag{2.9}$$

where

$$O_n^m = \begin{bmatrix} Y_n^m \\ Y_z^T \end{bmatrix} \tag{2.10}$$

Y_n^m is the original label and Y_z is the label of the new data Z.

Then, according to the equations above, we can calculate the new weights $^zW_n^m$ of the updated model only through the pseudoinverse of the new input as follows:

$$^zW_n^m = \left[W_n^m + B(Y_z - Z_n^m W_n^m) \right] \tag{2.11}$$

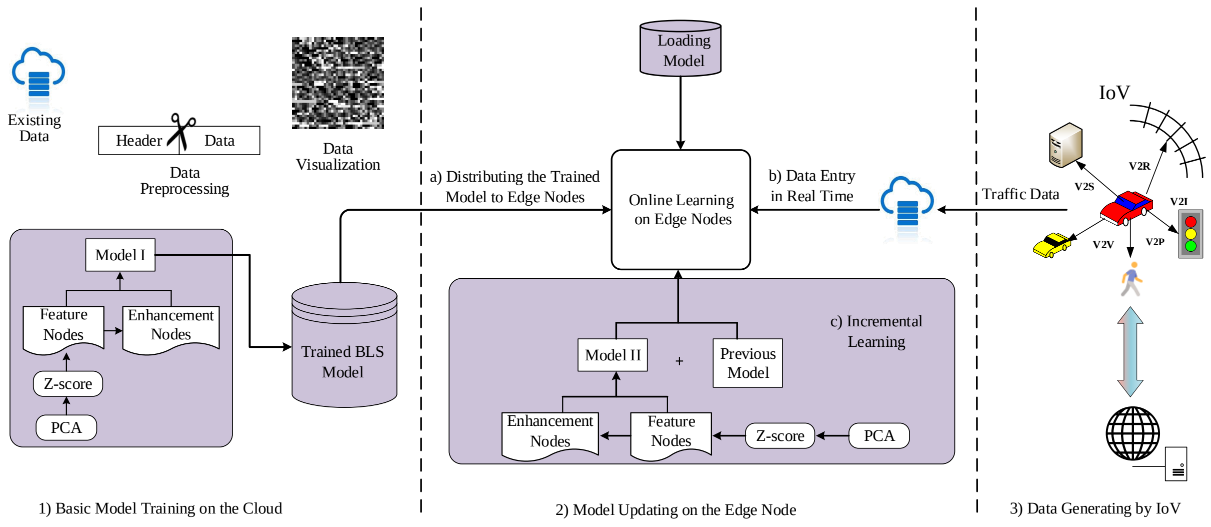


Fig. 2.3 Edge Computing based Traffic Analysis

Algorithm 1: Basic Model Training

```
Input: Traffic Data $X$, Label $Y_{a,c}$, maptimes $t_1$, enhencetimes $t_2$, Batchsize $g$
Output: Trained Basic Model $\mathscr{M}$
$X^{'} \leftarrow$ PCA (X); $X^{''}_{a,b} \leftarrow$ Z-score ($X^{'}$);
// Feature Nodes I
for each $i \in [1, t_1]$ do
    $A_{b,g}, \theta \leftarrow$ Random ();
    $I^{i}_{a,g} \leftarrow \varphi(X^{''}_{a,b} A_{b,g} + \theta); // \varphi$ is activation function
    $I_{a,m} \leftarrow$ Add $I^{i}_{a,g}$; // $m = t_1 * g$
end
// Enhancement Nodes K
for each $i \in [1, t_2]$ do
    $A^{'}_{m,g}, \theta^{'} \leftarrow$ Random ();
    $K^{i}_{a,g} \leftarrow \chi(I_{a,m} A^{'}_{m,g} + \theta^{'}); // \chi$ is activation function
    $K_{a,n} \leftarrow$ Add $K^{i}_{a,g}$; // $n = t_2 * g$
end
// Updating weight W
$P_{m+n,a} \leftarrow$ Pseuedoinverse ($I_{a,m}, K_{a,n}$);
$W_{m+n,c} \leftarrow P_{m+n,a} Y_{a,c}$;
$\mathscr{M} \leftarrow$ Save the Model ($W_{m+n,c}$, $P_{m+n,a}$, $I_{a,m}$, $K_{a,n}$)
return $\mathscr{M}$
```

2.3 Broad Learning Based Lightweight Traffic Analysis System in VECN

We present the traffic analysis system from the perspective of the architecture and the concrete training process.

2.3.1 System Design

To thwart the problems that cannot be solved by cloud computing and protect users' data privacy, we propose a distributed traffic analysis system with the help of edge computing and use a novel and lightweight network structure (broad learning system) to train the traffic data, which can better fit the vehicular edge computing based architecture. As shown in Fig. 2.3, the proposed edge computing based traffic analysis system is mainly composed of two stages: basic model training on the cloud and the model updating on the edge nodes, which are designed to reduce the bandwidth resources and protect users' privacy without affecting the analysis results.

Basic model training on the cloud

Considering that the edge nodes usually have poor computing power and lower memory capacity, it is impossible to process large amounts of traffic data for basic model training. Therefore, we use the cloud to perform this task which is the same as current architecture. After the basic model training, the cloud sends the basic model to edge nodes for model updating using the incremental learning, which does not need to interact with the cloud.

In order to make broad learning system more suitable for the edge computing based architecture, we modify the basic broad learning system algorithm. In our system, the cloud performs the following two steps as shown in Algorithm 1: (1) Analyzing the existing traffic data and generating the basic model. Our approach is not to completely abandon the previous architecture, but to further optimize on the basis of the existing architecture. Since the cloud has collected many traffic data, we can reuse these data for basic model training instead of recollecting data, which can further save the bandwidth resources. Firstly, we use principal component analysis (PCA) to extract the important feature data (input data). Then feature nodes and enhancement nodes can be generated based on original data, random weights/bias, and activation functions. By computing the pseudoinverse of the generated nodes, we can obtain final weights, which constitute the trained model.

Model updating on the edge node

In our system, the edge nodes refer to the devices between the user side and the cloud, which are close to the users' device and can be controlled by the users instead of the service providers. In general, the common edge nodes refer to the access point, router, hub, and switch devices. Usually, the edge nodes have poor computing power and lower storage capacity. Therefore, it is reasonable to deploy the lightweight model on the edge nodes.

After receiving the trained model, the edge nodes load this model and continue to train the model by the way of incremental learning as shown in Algorithm 2. The edge node does not need to process previous data, and it only needs to generate additional feature/enhancement nodes for new data. Data preprocessing is completed by PCA, and the generated way of additional feature/enhancement nodes is the same as Algorithm 1. These newly generated nodes will be used to update the BLS model by the dynamic stepwise updating algorithm. Through the incremental learning, we can constantly update the model and maintain the accuracy of the trained model.

Algorithm 2: Incremental Learning on the edge Node

```
Input: Basic Model $\mathscr{M}$, New Traffic $Z$, Label $Y'_{a',c}$
Output: The Updated Model $\mathscr{M}'$
$W_{m+n,c}, P_{m+n,a}, I_{a,m}, K_{a,n}$ ← Load Basic Model $\mathscr{M}$
$Z'$ ← PCA (Z); $Z''_{a',b}$ ← Z-score ($Z'$);
// Feature Nodes $I'$
$I'_{a',m}$ ← Use the same method as Algorithm 1
// Enhancement Nodes $K'$
$K'_{a',n}$ ← Use the same method as Algorithm 1
// Calculate $B_{m+n,a'}$
$D^T_{a',a} \leftarrow (I'_{a',m}|K'_{a',n})P_{m+n,a}$;
$C^T_{a',m+n} \leftarrow (I'_{a',m}|K'_{a',n}) - D^T_{a',a}(I_{a,m}|K_{a,n})$;
if $C_{m+n,a'}$ != 0 then
    $B_{m+n,a'}$ ← Pseuedoinverse ($C^T_{a',m+n}$);
end
else
    $B_{m+n,a'} \leftarrow P_{m+n,a}D_{a,a'}((D^T_{a',a}D_{a,a'} + diag(a')).I)$
end
// Updating weight W
$W_{m+n,c} \leftarrow W_{m+n,c} + B_{m+n,a'}(Y'_{a',c} - (I'_{a',m}|K'_{a',n})W_{m+n,c})$;
$P_{m+n,a+a'} \leftarrow ((P_{m+n,a} - B_{m+n,a'}D^T_{a',a})|B_{m+n,a'})$;
return $\mathscr{M}'$
```

2.3.2 Algorithm Implementation

Although broad learning system can efficiently reduce the training time due to its simple network structure, it is difficult to directly deploy the broad learning algorithm on devices with the limited performance. Fortunately, it can be realized with the following modifications as shown in Algorithm 1 and 2:

Incremental learning for fast retraining

We can combine broad learning system with incremental learning [16] to construct the fast retraining model, which does not need to retrain all traffic data. Incremental learning means that a learning system can continuously learn new knowledge from new samples and preserve most of the knowledge that has been learned before. Therefore, we can split traffic data into some parts, and train one part on the edge nodes each time. Based on the incremental learning, we can quickly train incoming new data and continuously update the trained model for maintaining high classification accuracy without retraining all traffic data.

Principal component analysis for data reduction

Since the input data always has higher dimension, it will inevitably take up a lot of training time if we directly input high-dimensional data into the training algorithm. Therefore, instead of inputting all traffic data into the training algorithm, we use data reduction method to process these traffic data and extract the important features. Principal component analysis is a common method for data reduction. It uses an orthogonal transformation to convert the high-dimensional data, which can convert correlated variables into linearly uncorrelated variables. We can set suitable parameters for principal component analysis, which has the negligible influence to the classification accuracy. Note that there exist two types of principal component analysis: basic principal component analysis and incremental principal component analysis which is a useful way to solve the limitation of memory [12, 5]. Since we perform the dimension reduction in an offline way, these two methods has the same effect. Therefore, we choose basic principal component analysis for the convenience of experimental analysis.

2.4 Evaluation

We implement our system on the classic microcontroller devices, Raspberry Pi, to evaluate the system performance and compare BLS based method with other machine learning method. We use a similar metric to other works on traffic analysis, which selects several classic traffic types [33].

2.4.1 Prototype

For evaluating the effectiveness/efficiency of our proposed scheme, we have deployed our scheme on the Raspberry Pi with a 64-bit ARM CPU at 1.2 GHz and 1GB of ARM memory. As a classic device, Raspberry Pi has the similar computing and storage capabilities with the edge nodes that are defined in our paper [21, 41]. Therefore, it is reasonable to evaluate the proposed edge computing based traffic analysis system using this device. In our experiment, we use 40 percent of traffic data for training the basic model. On the cloud, our system will set a threshold for the basic model's accuracy, and it will stop the training process if the accuracy exceeds the threshold. Moreover, 40 percent of traffic data is used for incremental learning on the edge nodes. In the process of incremental training, we will assign these data to multiple training stages to evaluate the effectiveness of the incremental learning. The model will be saved after each training stage is finished. Finally, the remaining 20 percent of the data is the test set.

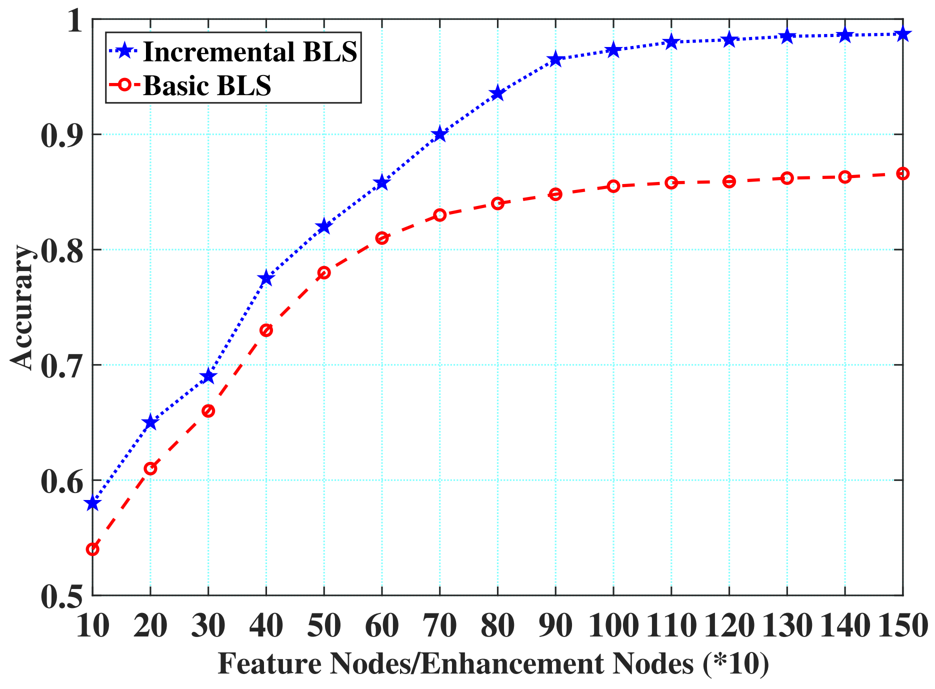
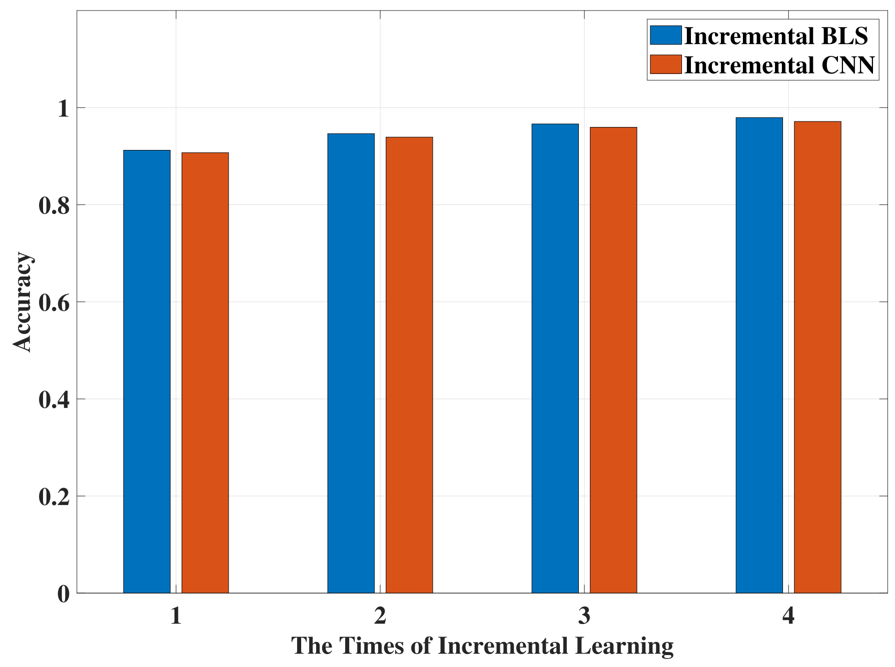


Fig. 2.4 The Impact of Feature/Enhancement Nodes

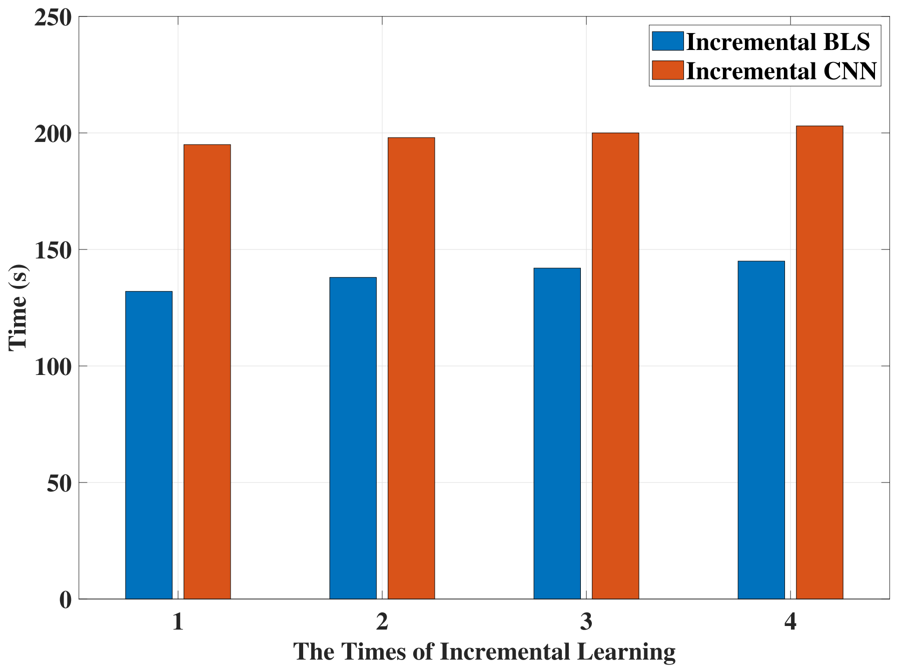
2.4.2 Classification Performance

We deploy the proposed system on the edge nodes, Raspberry Pi 3, to evaluate its performance. Firstly, we use classification accuracy as the evaluation parameter. The numbers of feature/enhancement nodes are important factors affecting the classification accuracy, which can be adjusted for obtaining accurate results. Note that, the training time will grow with the increase of these nodes. Therefore, we should find a tradeoff between the training time and the accuracy. Fig. 2.4 shows the classification accuracy of basic model and incremental model under different number of nodes. The accuracy of the trained model is significantly improved on the basis of the basic model by incremental training of new data.

We evaluate the effect of feature/enhancement nodes by varying their numbers from 100 to 1500. When we set the value of both the feature nodes and the enhancement nodes to 1000, the classification accuracy of the basic model and the incremental model are 85.5% and 98%. We can see that with the increase of the numbers of nodes, the classification accuracy



(a) Classification Accuracy



(b) Training Time

Fig. 2.5 Accuracy and Efficiency Comparison with CNN

Table 2.1 Training Time under Different Number of Feature/Enhancement Nodes

Basic Model		Incremental Model	
The Number of Nodes	Time	The Number of Nodes	Time
100	1.2s	100	54s
200	1.4s	200	63s
300	1.8s	300	72s
400	2.4s	400	81s
500	2.9s	500	89s
600	3.4s	600	102s
700	4.2s	700	109s
800	5.0s	800	112s
900	5.8s	900	125s
1000	6.5s	1000	132s
1100	7.0s	1100	142s
1200	7.6s	1200	150s
1300	8.2s	1300	158s
1400	8.7s	1400	170s
1500	9.3s	1500	176s

of the basic model and the incremental model will significantly grow when these nodes are small. However, in the later stage, even if the nodes continue to increase, the growing of the classification accuracy is limited. That is, when the classification accuracy reaches a threshold, the increase of the nodes does not work. Moreover, the increase of the nodes means the training time will grow, as shown in Table 2.1. As shown in this table, although basic model training takes a short time, the training time of the incremental learning will significantly increase with the growing of feature/enhancement nodes. Therefore, we need to set a reasonable parameter (such as 1000 or 1100) instead of increasing these nodes as many as possible.

2.4.3 Performance Comparison with Deep Learning

We compare the broad learning system based traffic analysis model with the deep learning based traffic analysis model, which is the most popular machine learning method. Note that, we use the classic deep structure, Convolutional Neural Network (CNN). Like the Broad learning system, CNN can also save the trained model and continue to train the model by the way of incremental learning. Therefore, we can use the same dataset and set the same parameters as the broad learning system, which can provide a consistent environment for performance comparison. Moreover, we use the best settings which are chosen from the validation set in BLS and CNN, and obtain the experimental results on the test set.

In our setting, we construct a CNN model which contains four convolutional layers and two full-connected layers. Data pre-processing uses the same steps as the broad learning system, including data visualization (38x38). When the accuracy of the basic CNN model is consistent with the basic model of the broad learning system, the basic CNN model is distributed to the same Raspberry Pi for incremental learning. The training results are shown in Fig. 2.5a, where $i = 1, 2, 3, 4$ represents each incremental stage. This figure shows that these two models have similar training accuracy in each incremental stage. Then we evaluate the training time of each incremental stage, and the results are shown in Fig. 2.5b. Our scheme only needs about 70% of the training time of the CNN model, which can show the superiority of the proposed model.

2.4.4 Running Time Comparison between Distributed Architecture and Cloud-based Architecture

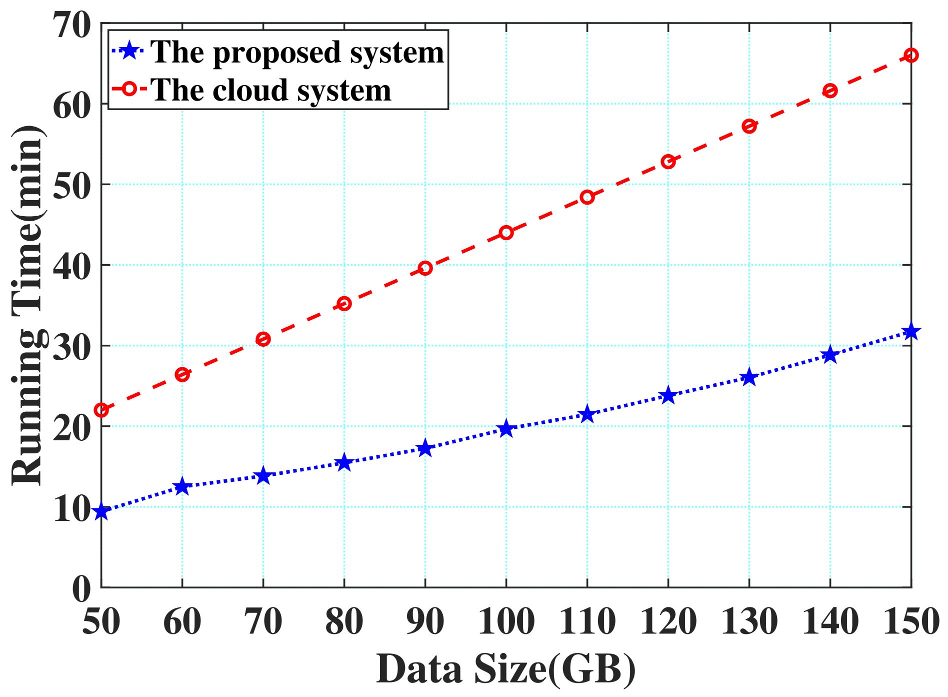


Fig. 2.6 The First Incremental Stage

To show the performance superiority of the distributed traffic analysis architecture, we compare the running time of the proposed traffic analysis system which executes the incremental learning on the edge nodes with the cloud based traffic analysis system. Since the basic model is trained on the cloud in both systems, we only need to compare the time spent in incremental learning process. We consider some scenarios that need to analyze large amounts of traffic data, such as residential area, university, and industrial park. We generalize these scenarios and generate a distributed traffic analysis system where multiple edge nodes are deployed. In these scenarios, edge nodes are not specially deployed devices, which refer to the access point, router, hub, or the users' device. These edge nodes are existing devices in our daily lives, which will not generate extra costs when introducing the edge computing based traffic analysis system.

In our setting, multiple edge nodes are deployed within a certain range (residential area, university, and industrial park). We fix the number of edge nodes at 100 when evaluating the impact of data size on running time under different traffic analysis systems (edge computing based traffic analysis system and cloud based traffic analysis system). In both systems, cloud is used to train the basic model. And the trained model is distributed to these edge nodes for further training using incremental learning at edge computing based traffic analysis system, while the basic model will continue to be trained on the cloud at cloud based traffic analysis system. Note that the updating process of the trained model will take place when the model is not accurate and accumulates a large amount of traffic data.

In our simulation, we use the transmission rate of the fifth generation (5G) mobile network when sending traffic data to the cloud, and the transmission rate is set to $442Mbps$. This data is obtained from the 5G real network simulation experiment of Qualcomm Technologies in San Francisco, which is the fastest file transfer speed in current fifth generation mobile networks testing. They simulate a hypothetical NSA 5G new air interface network, operating on a 28 GHz millimeter-wave spectrum with a bandwidth of 800 MHz [1]. Note that $442Mbps$ is the median 5G file download speed, which will be much faster than the file upload speed in general. Therefore, we choose a transmission rate that is much larger than the true upload speed for comparison, which can show the performance superiority of the distributed architecture.

We evaluate the impact of data size on running time of the edge computing based traffic analysis system and the cloud based computing system, and compare the running time (transmission time and computing time) when executing incremental learning after collecting different amounts of traffic data. In our experiment, we evaluate the effect of the amount of data when performing model updating by varying their size from 50GB to 150GB. Note that the incremental learning is performed on edge nodes in parallel, which can speed up data processing. Therefore, it is reasonable to choose the maximum processing time of the single edge node for comparison. We conduct multiple experiments and average the experimental results.

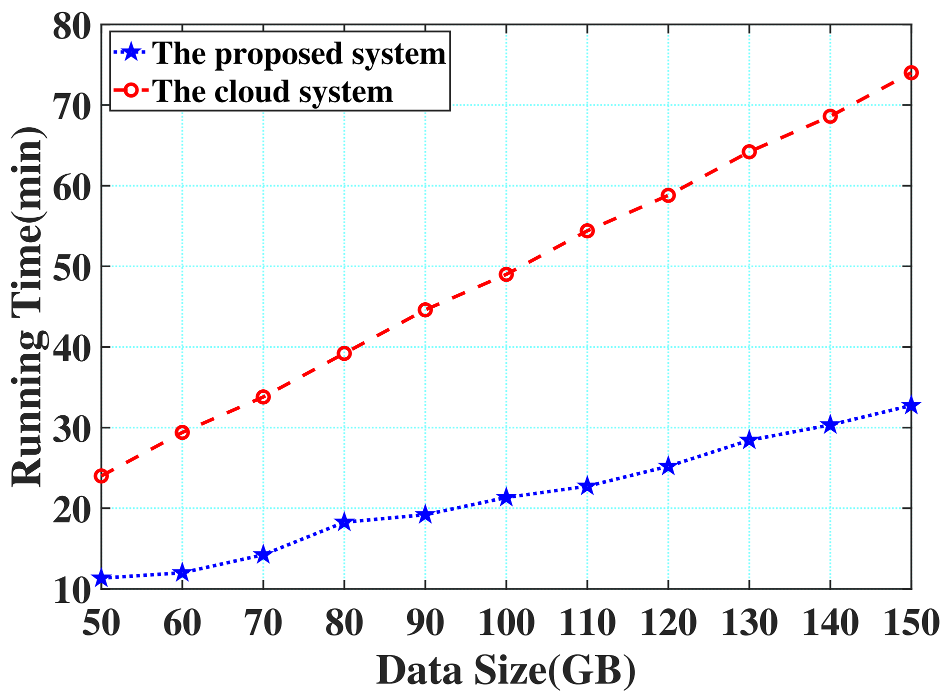


Fig. 2.7 The Second Incremental Stage

Fig. 2.6-2.9 show the results of the running time of the edge computing based traffic analysis system and the cloud based computing system in different incremental stages. The red line chart shows the running time of the cloud based traffic analysis system, and the other is the results of the edge computing based architecture. From these figures, we can see that the distributed architecture always need less time when processing large-scale traffic data in each incremental stage and there is a big gap in the running time between the two systems even if we set a very high transmission rate, which can show the superiority of the edge computing based traffic analysis system. As the amount of traffic data increases, the gap between the two systems is becoming more and more obvious. This is because the impact of transmission time is getting bigger and bigger with the increase of the amount of traffic data, which can demonstrate the effectiveness of our system when processing large-scale traffic data. These figures also show the running time of different incremental stages, and our system uses less running time than the cloud based traffic analysis system at each stage, which can show the stability of our system under different stages of incremental learning.

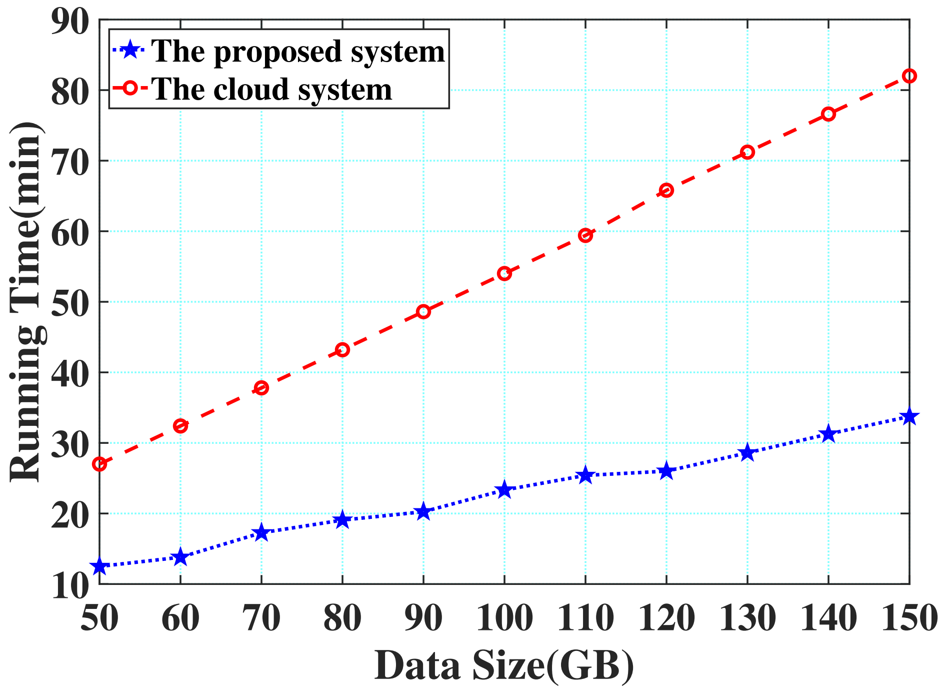


Fig. 2.8 The Third Incremental Stage

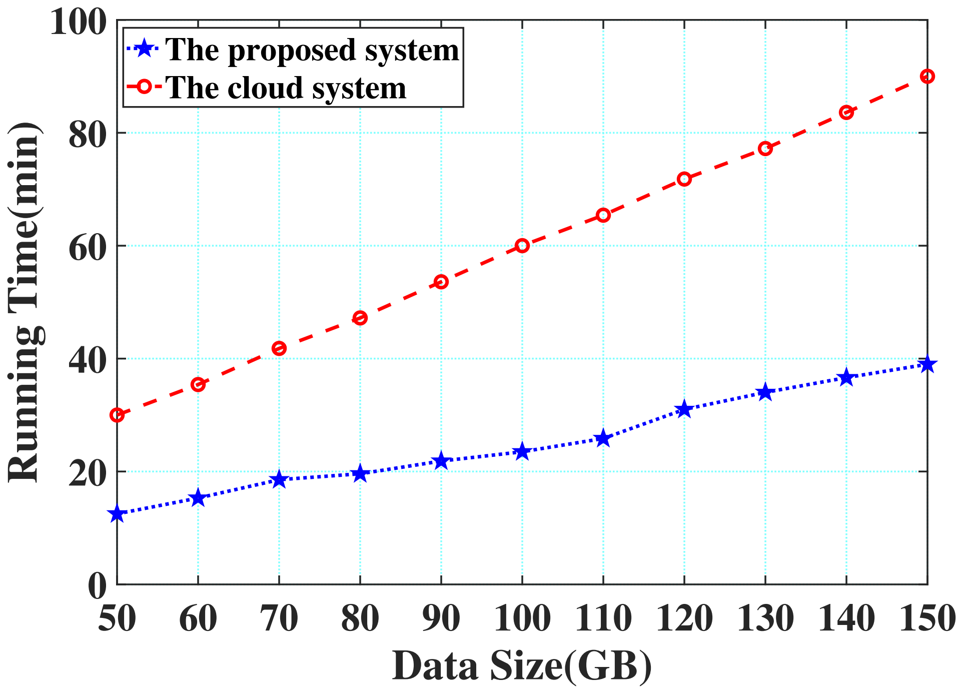


Fig. 2.9 The Fourth Incremental Stage

2.5 Conclusion

We propose a novel edge computing based traffic analysis using broad learning system. Firstly, edge computing can provide a distributed architecture, which can save the precious bandwidth resources and provide the safer service environment by offloading the analysis task to the network edge. Secondly, we use broad learning system to incrementally train the traffic data, which is more suitable for the edge computing because it has fast training speed and can support incremental learning. Then we implement the edge computing based traffic analysis system on the Raspberry Pi, which shows our model has the faster training speed compared with other neural network architecture.

Chapter 3

Multi-attribute Based Static/Offline Double Auction in VECN

3.1 Motivation

Different architectures of VEC have been proposed, including infrastructure-based VEC and vehicle-based VEC. Since some infrastructures, such as RSU, are not deployed in reality, vehicle-based VEC is regarded as more practical architecture. In this architecture, edge nodes refer to vehicles with remaining resources, especially slow-moving and parked vehicles. These vehicles have sufficient resources and motivations to provide services to vehicles with resource needs [34]. For the convenience of description, we refer to vehicles selling resources as vehicular edge nodes, and vehicles that need resources as client vehicles.

How to provide reasonable resource allocation is an important issue in VEC. As depicted in Fig. 3.1, on one hand, edge nodes need to consume their computing and storage resources when providing services. On the other hand, not all vehicles are willing to pay according to the wishes of edge nodes, which means different vehicles will pay different amounts of money for their tasks. Therefore, how to conduct fair trade between them is the key issue for providing reasonable resource allocation.

Auction is a popular way to provide fair resource allocation between buyers and sellers in the case of competition [8]. Since the interests of vehicles and edge nodes are usually inconsistent, it is better to design a double auction mechanism to consider the interests of all parties. Through double auction mechanism, the price charged from vehicles and payment for edge nodes could achieve a trade-off.

There are some works on edge-related auction mechanisms. In [25], the authors proposed a truthful auction mechanism in mobile cloud computing to achieve resource allocation

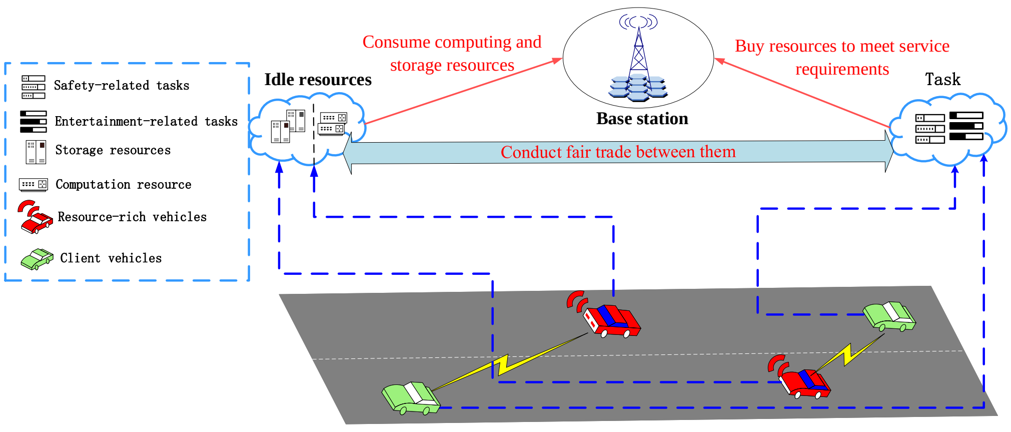

Safety-related tasks
Entertainment-related tasks
Storage resources
Computation resource
Resource-rich vehicles
Client vehicles
Idle resources
Consume computing and storage resources
Base station
Buy resources to meet service requirements
Task
Conduct fair trade between them

Fig. 3.1 The Reasonable Resource Allocation in VECN

between mobile devices and cloudlets. Sun *et al.* [57] considered the industrial Internet of things scenario in which the edge node is a resource-rich data center and extended the above truthful auction mechanism. However, few works study the resource auction issue in VEC scenario.

The existing edge-related auction mechanisms could not be directly used in VEC, which only consider the price information. Due to poor-quality wireless links, vehicular network has a large delay when transmitting large-scale data. One edge node is difficult to provide large-scale data processing services for vehicles who are far away. Therefore, location information should be considered when determining the matching in the VEC auction. Moreover, due to different types of tasks and different computing capabilities of edge nodes, vehicles have different preferences over edge nodes. For example, vehicles want to choose edge nodes with higher capabilities when processing safety-related tasks. Therefore, other non-price attributes, such as reputation and computing capability, also need to be considered in VEC auction.

In this Chapter, we design a multi-attribute based double auction mechanism in VEC scenario. The proposed auction mechanism not only considers the price but also considers non-price attributes when determining the winners, which could construct more reasonable matching between edge nodes and vehicles. In addition, our auction mechanism could satisfy the following economic properties: computational efficiency, individual rationality, budget balance, and truthfulness To verify our auction mechanism, we simulate the VEC scenario using VISSIM, an open source framework for running vehicle network simulation. Then we implement the proposed auction system by JAVA, and verify the effectiveness and efficiency of the double auction mechanism by the driving data extracted from VISSIM.

The main contributions are summarized as follows:

- We consider multi-attribute factors and propose an attribute-based matching model between edge nodes and vehicles. In our model, vehicles send task requests with bids and attribute requirements, and edge nodes process these tasks by providing their asks and attributes.
- We propose a multi-attribute based static/offline double auction mechanism which meets budget balance, truthfulness, computational efficiency, and individual rationality.
- We implement the proposed auction system by JAVA, and use the driving data extracted from VISSIM (vehicular network simulator) to verify the effectiveness and efficiency of the double auction mechanism. Experimental results show that our mechanism could meet the proposed properties.

The rest of this chapter is organized as follows. In section 3.2, we will give related work about resource allocation in vehicular edge computing. We briefly introduce the system model, auction model and the economic properties in section 3.3. Section 3.4 detailed presents the proposed multi-attribute based static/offline double auction mechanism. We discuss the experimental evaluation in section 3.5. We conclude the paper in section 3.6.

3.2 Related Work

3.2.1 Resource Allocation in Vehicular Edge Computing

VEC is an emerging paradigm in recent year, which introduces the idea of the edge computing paradigm into vehicular network to solve limitations (e.g., latency and transmission cost) in conventional vehicular network. Existing works on VEC architecture are mainly divided into two categories: infrastructure-based VEC which regards infrastructures close to vehicles as the edge nodes [39], and vehicle-based VEC which regards vehicles with the remaining resources as the edge nodes [40, 65]. Compared with infrastructure-based VEC which needs to deploy the additional infrastructure (e.g., Road Side Unit), vehicle-based VEC is easier to deploy. For example, Zhu *et al.* [76] proposed a VEC architecture which turns commercial fleets with predictable driving routes into edge nodes. Some applications in VEC have also been investigated, such as real-time traffic management [40] and edge-based vehicular crowdsensing [39].

As an emerging paradigm, existing works in VEC mainly focus on its architecture [65]. Few works investigated the resource allocation issue, which limits the development of VEC.

Feng *et al.* [15] designed a job scheduling method according to ant colony optimization. In [55], the authors proposed an adaptive resource scheduler for Edge Centers, which can maximize system efficiency. However, these works do not consider how to incentive vehicles and edge nodes to participate in resource sharing. The design of an efficient incentive mechanism in VEC scenario is still a great challenge.

3.2.2 Auction Mechanisms in Vehicular Network and Edge Computing

Nowadays, auction issues in traditional vehicular network have received the attentions from the academia. In [36], the authors proposed a VCG-based reverse auction scheme for cloud-based vehicular network, which can only meet the properties of truthfulness and individual rationality. Kumar *et al.* [29] studied the spectrum handoff issues in cognitive radio vehicular network, and proposed a game theoretic auction theory approach to select the optimal network for spectrum handoff. In [69, 74], the authors studied the energy trading of electric vehicles, and proposed the efficient auction mechanisms to incentive electric vehicles in the two-layer vehicle-to-grid (V2G) system. However, these works study the auction mechanisms in traditional cloud-based vehicular network and do not consider the auction issues in the vehicular edge computing scenario which contain many edge nodes with different interests.

Although there is little work to design auction mechanisms in VEC, some auction mechanisms in other edge computing scenarios have been proposed [70], such as mobile cloud computing [25, 26] and industrial Internet of things [57]. Sun *et al.* [57] considered industrial Internet of things scenario in which the edge node is a resource-rich data center and proposed a double auction scheme which can fit one-to-many scenario (one edge server can serve multiple devices). However, this auction mechanism cannot be applied to the VEC since vehicular edge nodes have fewer resources. In [4], the authors solved the resource auction problem at the edge/cloud levels. However, it cannot meet the property of truthfulness. Kiani *et al.* [28] introduced a hierarchical mobile edge computing which contains different types of cloudlets and proposed a resource allocation mechanism with two-time scale. However, this mechanism takes a long time, which is not suitable for rapid changes of the network topology in VEC. In [26], the authors proposed an incentive-compatible auction mechanism in mobile cloud computing. Then they extended it and proposed two auction mechanisms which can meet desirable properties according to different needs [25].

Note that the above works only consider the price factor when determining the winners, which could not be directly used in the VEC. As discussed in the introduction, due to the poor-quality wireless links which limit the range of data transmission among vehicles [78] and different types of tasks in the vehicular network which need to choose different edge nodes based on the task requirements, some non-price attributes, such as location, reputation and computing capability, also need to be considered in VEC auction. Therefore, in this paper, we consider these important non-price attributes to construct more reasonable matching between vehicles and edge nodes when designing the double auction mechanism for VEC scenario.

3.3 Problem Formulation

In this section, we will introduce the system model, auction model, and the design objective of the auction mechanism.

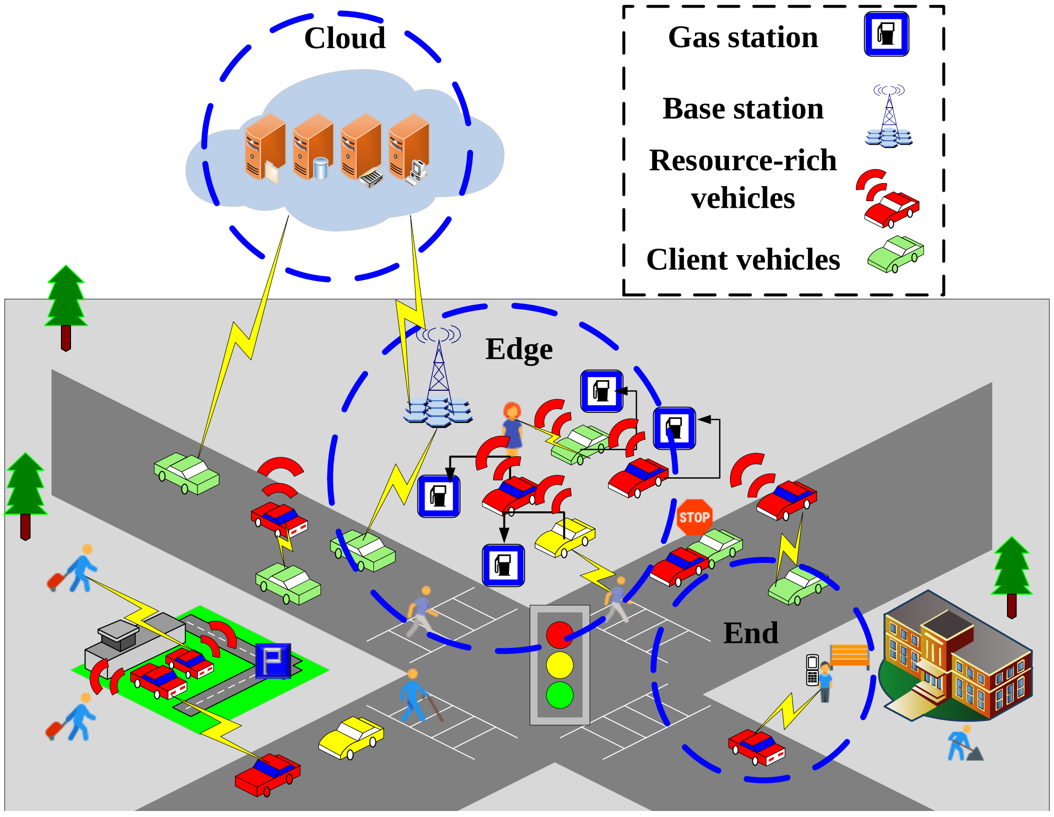


Fig. 3.2 The Architecture of VECN about Static Scenario

3.3.1 System Model

As depicted in Fig.3.2, the static scenario of VECN system includes a cloud-based platform, some base stations (BSs), and massive vehicles serving as vehicular edge nodes (sellers) or client vehicles (buyers). Due to the limitation of transmission distance, a vehicular edge node can only provide services for its neighboring client vehicles. Therefore, we divide the VEC system into some subsystems according to the coverage of a base station, and the auction is performed among vehicles covered by the same base station [76]. The base station, a trusted third party, can be used as an auctioneer to determine winners. In our model, the base station only needs to perform the auction process (such as matching and pricing), but it does not need to provide the resources for executing client vehicles' tasks. Note that we only consider that one edge node can only serve one vehicle at a time because the vehicular edge node has limited resources.

To construct more reasonable matching between vehicular edge nodes and client vehicles, we consider the non-price attributes in this paper. We choose three kinds of important attributes as the examples and explain the rationality of using them in the VEC auction mechanism. Note that it is easy to extend to other attributes. If we want to add one attribute, the auction system will notify client vehicles to submit their attribute requirements and edge nodes to submit their attribute values before the auction process begins.

(A.) Location: The poor-quality wireless links make the vehicular edge nodes can only serve the nearby vehicles [61, 44]. Therefore, the location should be considered when establishing the matching between them in the VEC auction scenario.

(B.) Reputation: It is difficult to ensure vehicles can honestly publish or perform tasks. For example, some sellers may forge calculation results or reduce the performing speed, which affects the buyer's service experience. Therefore, a reputation system is needed to ensure trust services.

(C.) Computing Power: Client vehicles have different requirements for the execution time of different tasks. For example, some safety-related tasks need to be processed in a very short time, which need more computing power. On the contrary, vehicles do not have such high requirements for entertainment-related tasks. Therefore, computing power also needs to be considered to provide flexible services for different needs of vehicles.

At the begin of each auction round, vehicles dynamically join a sub-auction system using different identities (vehicular edge node or client vehicle) and register with the auctioneer within the communication range using their personal information (ID, type of the vehicle, location, reputation and computing capabilities). At the same time, the information about all vehicular edge nodes will be distributed to each client vehicle under the same sub-auction system.

At the bid submission phase, each client vehicle computes its bids for all vehicular edge nodes, which are different values according to sellers' attributes (resources) and the buyer's preferences (the calculation method is shown in Section 3.3.2). At the same time, the client vehicles also submit minimum attribute requirements in order to construct reasonable

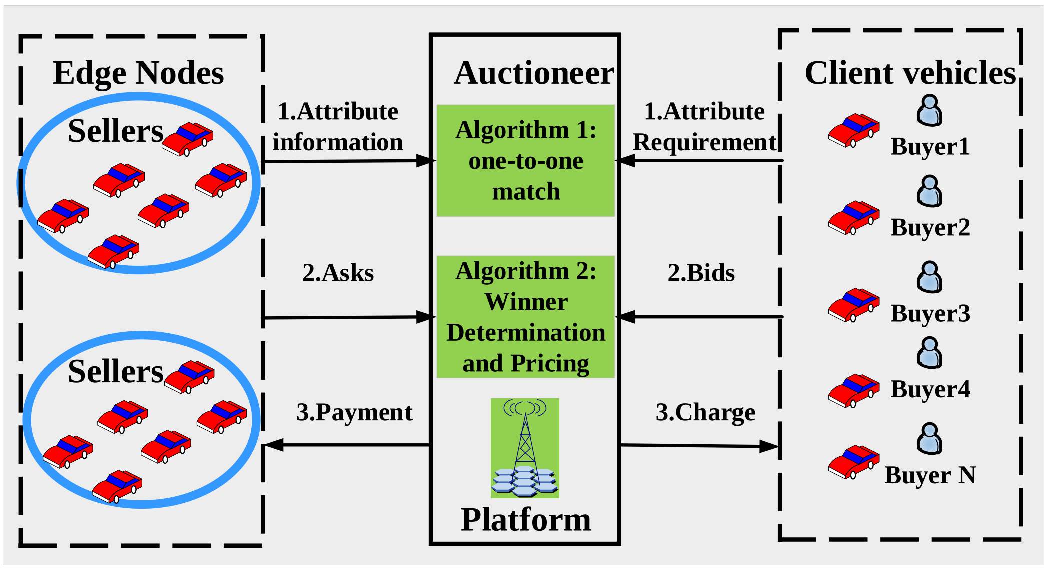


Fig. 3.3 The Multi-attribute based Double Auction Framework for VEC

matching. Similarly, the vehicular edge nodes submit their asks to the auctioneer. After receiving the bids and asks, the auctioneer executes the auction algorithm and determines the winners based on information provided by buyers and sellers. The concrete auction process is shown in Fig. 3.3.

3.3.2 Auction Model

Considering m vehicular edge nodes provide resources for n client vehicles, we model our problem as a single-round double auction:

- Let $\mathscr{B} = \{b_1, b_2, ..., b_n\}$ be the set of client vehicles (buyers), and $|\mathscr{B}| = n$. We denote minimum attribute requirements of i-th buyer as $q_i^b = \{(q_i^{11}, q_i^{12}), q_i^2, q_i^3, ..., q_i^k\}$, where q_i^{11} and q_i^{12} represent the location and acceptable distance with a vehicular edge node, while q_i^2, q_i^3, and q_i^k represent the acceptable reputation, computing power, and the requirement of k-th attribute.

- Similarly, $\mathscr{S} = \{s_1, s_2, ..., s_m\}$ is m vehicular edge nodes (sellers). $q_j^s = (q_j^1, q_j^2, q_j^3, ..., q_j^k)$ are the attribute values owned by thej-th vehicular edge node, which represent location, reputation, computing power, and the value of k-th attribute respectively.

- For each b_i, its bids for every seller s_j is $H_i = (h_i^1, h_i^2, ..., h_i^m)$. As different edge nodes have different attributes (resources), buyers are willing to pay different prices. Moreover, buyers have preferences over these attributes according to tasks, which can be defined as the attribute weights $\omega_i = (\omega_i^1, \omega_i^2, \omega_i^3, ..., \omega_i^k)$. For example, buyers processing safety-related tasks will give a larger weight to reputation. Based on attributes of edge nodes and attribute weights, one buyer could give different bids for sellers. We could compute the weights based on the attribute requirements q_i^b, and assign greater weight for the attribute which has more demand.

- For all sellers in $\mathscr{S}$, they give asks according to their resources, which is defined as $A = (A_1, A_2, ..., A_m)$. A seller requests the same ask for different buyers since it provides all resources for one client vehicle in one round.

According to resource attributes q_j^s of the edge node (obtained from the auctioneer) and the buyer's attribute weights $\omega_i = (\omega_i^1, \omega_i^2, \omega_i^3, ..., \omega_i^k)$, the buyer b_i could compute the valuation $V_i^j(\omega_i, q_j^s)$ for having services from the edge node s_j:

$$V_i^j(\omega_i, q_j^s) = \Phi_i + \omega_i^1 * (d_c - d(q_i^{11}, q_j^1)) + \sum_{a=2}^{k} \omega_i^a * q_j^a \tag{3.1}$$

where Φ_i is the fixed valuation. d_c is the diameter of the base station's coverage and $d(\cdot,\cdot)$ is the distance between the buyer and the seller. Note that these attribute values should be mapped to a unified non-dimensional interval firstly.

As a truthful auction mechanism, we will ensure that a buyer/seller has the maximum utility if it submits the bid h_i^j and ask A_j which are equal to the true valuation $V_i^j(\omega_i, q_j^s)$ and cost $C_j(q_j^s)$ for providing the resources (the proof is shown in Section 3.4.4). $C_j(q_j^s)$ is fixed, which will not change even if this seller s_j provides services for different buyers. Based on bids/asks, the auctioneer determines winning client vehicles $\mathscr{B}_w$ and vehicular edge nodes $\mathscr{S}_w$. Then it will determine the price P_i^b charged from b_i and the reward P_j^s for s_j. We define U_{ij}^b as the utility of the buyer b_i if this vehicle is matched with s_j and U_j^s as the utility of s_j:

$$U_{ij}^b = \begin{cases} V_i^j(\omega_i, q_j^s) - P_i^b;, & \text{if } b_i \in \mathscr{B}_w \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

$$U_j^s = \begin{cases} P_j^s - C_j(q_j^s), & \text{if } s_j \in \mathscr{S}_w \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

3.3.3 Economic Properties and Design Objective

A feasible and fair double auction mechanism usually meets the following basic properties:

- Individual Rationality: Individual rationality means the bid of a winning buyer should be greater than the charge ($h_i^j \geq P_i^b$) and the ask of a winning seller should be less than the payment ($A_j \leq P_j^s$).
- Computational Efficiency: The muti-attribute based double auction mechanism is computational efficiency if it has polynomial time complexity.
- Budget Balance: Total payments that the auctioneer pays to winning edge nodes should be less than total prices that the auctioneer charges from winning buyers.
- Truthfulness: As a truthful buyer/seller, it will honestly provide its bid/ask which is equal to its valuation/cost. However, it is reasonable to assume that the buyer/seller is selfish, and the buyer/seller has the enough motivation to increase its utility by submitting a bid/ask different from its true valuation/cost, which will affect the fairness of the auction. Therefore, our mechanism must ensure: the buyer/seller could obtain the maximum utility if they honestly provide a bid/ask which is equal to its valuation/cost. That means $\forall b_i \in \mathscr{B}, U_i^b$ is maximum when the bid $H_i = V_i$, and $\forall s_j \in \mathscr{S}, U_j^s$ is maximum when the ask $A_j = C_j$, where U_i^b is a vector representing the utilities of client vehicle i when it is matched by each edge node and V_i is the true valuations when client vehicle i is served by each edge node.

Moreover, system efficiency also needs to be considered for constructing an efficient auction mechanism. Since edge nodes have different resources and client vehicles have different resource requirements, it is better to increase resource utilization as much as possible on the basis of meeting buyers' needs. Therefore, we choose resource utilization (allocate more resources to the buyer who has higher requirements) as the measurement of system efficiency.

Although an auction mechanism that meets these five properties is a perfect auction mechanism, unfortunately, there is no double auction mechanism can satisfy these five properties at the same time [25, 66]. Therefore, we design a feasible auction mechanism that can strictly satisfy the first four properties, which can provide a fair and reasonable auction environment. At the same time, our mechanism can partly ensure system efficiency by allocating reasonable weights, as described in the following section.

Algorithm 3: One-to-One Assignment Algorithm

```
   Input: Weighted Bipartite Graph $T = (\mathscr{B} \cup \mathscr{S}, \mathscr{B} \leftrightarrow \mathscr{S})$
   Output: One-to-One Assignment $M$
1  Adding some virtual vertices and edges with a weight of $-1$ to $T$ to make it a
     balanced bipartite graph $T^{'}$.
2  Setting the initial labelling $l()$ for every vertex in $T^{'}$:
3  for each $b_i \in \mathscr{B}$ do
4      $l(b_i) = max_{s_j \in S}\{\lambda(b_i, s_j)\}$
5  end
6  for each $s_j \in \mathscr{S}$ do
7      $l(s_j) = 0$
8  end
9  Generating the equality graph $T_l^{'}$ which meets:
10 $T_l^{'} = \{(b_i, s_j) : l(b_i) + l(s_j) = \lambda(b_i, s_j)\}$
11 Selecting a random matching $M$ in $T_l^{'}$;
12 "Result" is a boolean variable; "$L \subset \mathscr{B}$, $R \subset \mathscr{S}$" are the sets of unsaturated and
     saturation points of "M" when the Hungarian algorithm terminates, respectively.
13 $Result$, $L$, $R \leftarrow$ Hungarian algorithm ($M$, $T_l^{'}$);
14 if $Result == True$ then
15     $M \leftarrow$ Remove the virtual vertices and edges in $M$;
16     return $M$;
17 end
18 else
19     $\beta_l = \min_{b_i \in L, s_j \in \mathscr{S}-R}\{l(b_i) + l(s_j) - \lambda(b_i, s_j)\}$;
20     updating the labelling:
21     $l^{'}(u) = \begin{cases} l(u) - \beta_l, & u \in L \\ l(u) + \beta_l, & u \in R \\ l(u), & others \end{cases}$
22     updating the equality graph $T_l^{'}$ based on the new labelling. Go to line 11.
23 end
```

3.4 Multi-attribute Based Static/Offline Double Auction Mechanism in VECN

The proposed multi-attribute based double auction mechanism (MADA) includes three main stages: Matching Stage, Assignment Stage, and Winner Determination and Pricing Stage. We will introduce these three stages in detail and demonstrate that the proposed auction mechanism could satisfy the basic properties in this section.

3.4.1 Attributes based Buyer-Seller Matching

The first step of the buyer-seller matching is to build the connection between client vehicles and vehicular edge nodes based on the attributes since not all edge nodes could meet client vehicles' requirements. Moreover, our design objective includes increasing resource utilization as much as possible on the basis of meeting buyers' needs. Therefore, we should set a reasonable weight for every connection to construct the optimal matching between them.

In this chapter, we model our problem as a weighted bipartite graph to represent the matching. The vertices are buyers and sellers, and the edges represent whether the sellers can provide services to the buyers. Then we set a reasonable weight for each edge to ensure assigning a larger weight to an edge who connects a buyer with greater requirements and a seller with more resources. Therefore, our design objective can be converted to find the maximum weighted matching in a weighted bipartite graph.

Constructing the Unweighted Bipartite Graph

We say b_i and s_j have a matching ($b_i \leftrightarrow s_j$) if the resources of s_j is greater than the requirements of b_i. Based on the matching between them, we can construct an unweighted bipartite graph $T = (\mathscr{B} \cup \mathscr{S}, \mathscr{B} \leftrightarrow \mathscr{S})$, where $\mathscr{B}$ and $\mathscr{S}$ are two sets of vertices representing n vehicles and m vehicular edge nodes, respectively. $\mathscr{B} \leftrightarrow \mathscr{S}$ represents the matching between them.

A buyer b_i has the minimum attribute requirements q_i^b, and s_j has the resources q_j^s. If there exists a matching ($b_i \leftrightarrow s_j$) between b_i and s_j, they should meet the non-price attribute constraints:

$$(d(q_i^{11}, q_j^1) \leq q_i^{12}) \cap (q_j^2 \geq q_i^2) \cap (q_j^3 \geq q_i^3) \cap \cdots \tag{3.4}$$

where q_i^{11} and q_i^{12} represent the location and acceptable distance with a vehicular edge node, while q_i^2 and q_i^3 represent the acceptable reputation and computing power. Similarly, q_j^1,

q_j^2, and q_j^3 are the attribute values owned by the j-th vehicular edge node, which represent location, reputation and computing power respectively. After satisfying the above constraints, we can establish the unweighted matching (coined as an edge) between b_i and s_j.

Setting the Weight for Each Edge

According to our design objective, we want to allocate more resources to the buyer who has higher demand for providing better services. Therefore, we define the weight as the product of the buyer's requirements and the seller's resources if there exists a matching between them, which could ensure assigning a larger weight to an edge who connects a buyer with greater demand and a seller with more resources.

$$\lambda(b_i, s_j) = (d_c - q_i^{12}) * (d_c - d(q_i^{11}, q_j^1)) + \sum_{a=2}^{k} q_i^a * q_j^a \tag{3.5}$$

Then we can obtain the weighted bipartite graph $T = (\mathscr{B} \cup \mathscr{S}, \mathscr{B} \leftrightarrow \mathscr{S})$ by setting the weight for every matching.

3.4.2 One-to-One Assignment by the Weighted Bipartite Graph

ph $T = (\mathscr{B} \cup \mathscr{S}, \mathscr{B} \leftrightarrow \mathscr{S})$. The concrete procedure is shown in Algorithm 3: Firstly, we add some virtual vertices and edges with the weights of "-1" (which means it is not a normal matching) to T to make it a balanced bipartite graph T'. Secondly, we set the initial labelling $l(\cdot)$ for every vertex, and generate the equality subgraph T_l' which meets the equation in line 10 of Algorithm 3:

$$T_l' = \{(b_i, s_j) : l(b_i) + l(s_j) = \lambda(b_i, s_j)\} \tag{3.6}$$

Thirdly, we execute the Hungarian algorithm (Line 13) to find the perfect matching (maximum weighted matching) in T_l'. If there is no perfect matching, we will relax the labelling (Line 18-21) for introducing new edges into T_l' and repeat this algorithm until we find the perfect matching M. After relaxing the labelling, the original feasible edge is still feasible, and the edge that was not feasible becomes a feasible edge now, which means we can definitely find the perfect matching after performing multiple relaxations. For each edge $(b_i \leftrightarrow s_j)$ in M, it means the task from vehicle b_i can be processed by the vehicular edge node s_j.

Based on the above steps, we could obtain the maximum weighted matching "M", which is the one-to-one assignment ($s_j = M(b_i)$) between edge nodes and client vehicles. In the next section, we will show how to design a reasonable winner determination and pricing scheme based on one-to-one assignment "M".

Algorithm 4: Winner Determination and Pricing

```
Input: The matching set $M$, the candidate client vehicles $\mathscr{B}_c$ and vehicular edge
       nodes $\mathscr{S}_c$ in $M$
Output: The winning buyers $\mathscr{B}_\omega$ and sellers $\mathscr{S}_\omega$; the charge $P^b$ and payment $P^s$
1  Sorting all buyers in $\mathscr{B}_c$ according to the bids by an descending order:
   $h_{i_1}^{M(i_1)} \geq h_{i_2}^{M(i_2)} \ldots$;
2  Sorting all sellers in $\mathscr{S}_c$ according to the asks by an ascending order: $A_{j_1} \leq A_{j_2} \ldots$;
3  Searching for the largest $g$ (Aligned Boundary) from the first element in sorted
     buyers/sellers: $h_{i_g}^{M(i_g)} \geq A_{j_g}$;
4  Searching for the largest $a$ from the (g+1)th buyer such that $h_{i_a}^{M(i_a)} \geq A_{j_g}$ and the
     largest $b$ from the (g+1)th seller such that $h_{i_g}^{M(i_g)} \geq A_{j_b}$;
5  $(\theta,\eta) \leftarrow (a-1,g-1)$ or $(g-1,b-1)$: Choosing one pair with more matchings
     from $(\mathscr{B}_{a-1},\mathscr{S}_{g-1})$ and $(\mathscr{B}_{g-1},\mathscr{S}_{b-1})$; // $\mathscr{B}_{a-1}$ means first "$a-1$" elements in
     sorted $\mathscr{B}_c$
6  The subscript of Boundary Pair: $(\theta,\eta) \leftarrow (\theta+1,\eta+1)$
7  // Pricing
8  $P^b \leftarrow h_{i_\theta}^{M(i_\theta)}, P^s \leftarrow A_{j_\eta}$;
9  $\mathscr{B}_\omega \leftarrow$ Choosing the first "$\theta - 1$" elements in sorted $\mathscr{B}_c$;
10 $\mathscr{S}_\omega \leftarrow$ Choosing the first "$\eta - 1$" elements in sorted $\mathscr{S}_c$;
11 $\mathscr{B}_\omega, \mathscr{S}_\omega \leftarrow$ Removing all buyers/sellers who do not have the matching in $\mathscr{S}_\omega$/$\mathscr{B}_\omega$;
12 return $(\mathscr{B}_\omega, \mathscr{S}_\omega, P^b, P^s)$
```

3.4.3 Winner Determination and Pricing

In the previous step, algorithm 3 outputs one-to-one assignment "M". Since part of assignments between buyers and sellers do not have a consistent price, these assignments do not represent final winners. Therefore, a reasonable winner determination and pricing algorithm is needed. McAfee double auction is a classical auction mechanism to price the homogeneous items. However, the simple pricing which does not consider the assignments between buyers and sellers will miss some winning buyers/sellers. Therefore, we refer to another truthful and computationally efficient pricing scheme [66].

The core idea of our pricing algorithm is shown as follows: We firstly sort the bids of buyers in descending order and asks of sellers in ascending order for finding the aligned boundary (Line 3). Then we fix the aligned boundary of one part (buyer or seller) and relax the boundary of another part for finding the extended boundary pairs, which have more candidate buyers and sellers. The boundary pair with more matchings will be regarded as the final price boundary. The concrete process is shown in Algorithm 4.

3.4.4 Theoretical Analysis

We will prove that our scheme satisfies the properties of individual rationality, computational efficiency, budget balance, and truthfulness.

Theorem 1. *MADA satisfies individual rationality.*

Proof. As shown in Line 1 and 2 of Algorithm 4, we sort the buyers and sellers by the descending and ascending order, and the winning buyers/sellers are the first "$\theta - 1/\eta - 1$" elements in sorted $\mathscr{B}_c/\mathscr{S}_c$. That means each winning buyer has the higher bid $h_i^{M(i)}$ than the charge $P^b = h_{i_\theta}^{M(i_\theta)}$, and each winning seller has the lower ask A_j than the payment $P^s = A_{j_\eta}$, which can ensure the individual rationality. □

Theorem 2. *MADA is budget-balance for the auctioneer.*

Proof. The muti-attribute based double auction mechanism meets the property of budget balance if the total rewards that the auctioneer pays to all winning edge nodes are not less than the total price the auctioneer charges from all winning client vehicles. For each b_i and the corresponding s_j, the utility that the auctioneer could obtain is $P_i^b - P_j^s = P^b - P^s \geq 0$. Therefore, the overall utilities of the auctioneer are greater than zero, which can meet the property of budget balance. □

Theorem 3. *The total time complexity of MADA is polynomial in the order of $O(\chi^3)$, where χ is the larger one of m and n.*

Proof. In the Matching stage, we need to set the weight between each buyer and seller. Therefore, the time complexity is $O(nm)$. In the Assignment stage, the KM algorithm always can be achieved in the time complexity of $O(\chi^3)$ [79], where χ is $\max\{m,n\}$. In the final stage, the sorting operations need a time complexity of $O(\chi \log(\chi))$. Then the searching operation needs a time complexity of $O(\psi)$, where ψ is less than $\min\{m,n\}$. Finally, the removing operations need to traverse each buyer/seller before the boundary pair, which needs a time complexity of $O(\psi^2)$. Therefore, the total time complexity of MADA is polynomial in the order of $O(\chi^3)$. □

Theorem 4. *our mechanism can ensure the truthfulness, which means the buyer/seller could not improve its utility by providing a bid/ask which is not equal to its real valuation.*

Proof. Firstly, we prove that buyers/sellers cannot submit fake attributes. Location can be obtained from GPS, which is easy to detect if they submit a fake location [22]. As for the reputation, it is evaluated by another entity (buyers/sellers). When a edge node completed a client vehicle' task, the client vehicle will evaluate the edge node in terms of execution time and accuracy of the results. Similarly, the edge node will evaluate the client vehicle in terms of payment time. Then these information (score) will be uploaded to a reputation server, and the distributed auctioneers will download and update this information in time. Therefore, they cannot modify it since reputation information cannot be controlled by themselves. Computing power is related to the type of vehicles, which will be provided when registering with the auction system. Since the type of vehicles is difficult to fake, it is easy to ensure drivers will submit the real computing power based on the registration information. Therefore, buyers/sellers cannot submit fake attributes. Then the matching and assignment algorithm will output a deterministic and bid/ask-independent assignment results, which means the changing of bids/asks will not affect assignment results.

Then we will prove that the changing of bids/asks will not affect the pricing stage. We use the proof of sellers' truthfulness as the example, and the truthfulness of buyers can be proved in similar ways. We define $\tilde{A}_{M(i)}$ as an ask different from the cost and $A_{M(i)}$ is equivalent to the cost $C_{M(i)}$. $\tilde{U}^s_{M(i)}$ and $U^s_{M(i)}$ represent their utilities respectively. We will discuss the following two cases separately:

1) $\tilde{A}_{M(i)} > A_{M(i)}$: There are four sub-cases.

- The seller $s_{M(i)}$ is the winner when submitting both $\tilde{A}_{M(i)}$ and $A_{M(i)}$. Without loss of generality, we assume the boundary pair is (b_{i_g}, s_{j_b}) when submitting $A_{M(i)}$ for auction. Note that another boundary pair (b_{i_a}, s_{j_g}) has the similar deduction processes. We define y, x as the position numbers of $s_{M(i)}$ and the corresponding buyer b_{i_x}, where

$A_{j_y} = A_{M(i_x)}$. Similarly, $\tilde{y}$ is the position number of $s_{M(i)}$ when submitting $\tilde{A}_{M(i)}$, where $\tilde{A}_{M(i)} = \tilde{A}_{j_{\tilde{y}}}$ and $\tilde{y} > y$. Therefore, we have $A_{j_y} < \tilde{A}_{j_{\tilde{y}}} < A_{j_b}$ or $A_{j_y} < A_{j_b} < \tilde{A}_{j_{\tilde{y}}}$.

For the first situation, we have: ① $y < \tilde{y} < g < b$; ② $g < y < \tilde{y} < b$; ③ $y < g < \tilde{y} < b$. For the sub-case ①, it will not affect the ask at/behind position g, which can obtain the same aligned boundary g and boundary pair. For the sub-case ②, since $h_{i_y} \le h_{i_{g+1}} < A_{j_{g+1}} \le A_{j_y} \le A_{j_{y+1}}$ in the original order (submitting $A_{M(i)}$) and $s_{j_{y+1}}$ in the original order will be moved to the position y when submitting $\tilde{A}_{M(i)}$, position g is still the aligned boundary ($h_{i_g} \ge A_{j_g}$, $h_{i_{g+1}} < A_{j_{g+1}}$). Therefore, we can obtain the same g and boundary pair since $h_{i_g} \ge A_{j_b} > \tilde{A}_{j_{\tilde{y}}}$. Sub-case ③ has the same deduction process as sub-case ②.

For the second situation, we have: ① $y < g < b < \tilde{y}$; ② $g < y < b < \tilde{y}$. For the sub-case ①, we have $h_{i_g} \ge A_{j_b} \ge A_{j_{g+1}}$ and $h_{i_{g+1}} < A_{j_{g+1}} < A_{j_{g+2}}$ since the boundary pair is (b_{i_g}, s_{j_b}). When submitting $\tilde{A}_{j_{\tilde{y}}}$, $s_{j_{g+1}}$ and $s_{j_{g+2}}$ will be moved to position g and $g+1$. Therefore, position g is still the aligned boundary since $h_{i_g} \ge A_{j_{g+1}}$ and $h_{i_{g+1}} < A_{j_{g+2}}$. Since $h_{i_g} \ge A_{j_b}$ and $h_{i_g} < A_{j_{b+1}}$, the boundary of seller is located at position $b-1$ when we fix the boundary of buyer (Line 4 in Algorithm 4), which could not include the new ask $\tilde{A}_{j_{\tilde{y}}}$ into final winners. Therefore, this sub-case is impossible. We could derive the same result for the sub-case ②.

Therefore, we could obtain the same boundary pair when $\tilde{A}_{M(i)} > A_{M(i)}$. The seller will be paid the same price P^s and these two asks have the same utility: $\tilde{U}^s_{M(i)} = U^s_{M(i)} = P^s - C_{M(i)}$.

- $s_{M(i)}$ only wins when submitting $\tilde{A}_{M(i)}$; If the ask $\tilde{A}_{M(i)}$ wins and we submit a bid $A_{M(i)}$ which is less than $\tilde{A}_{M(i)}$, it must be a winning seller as discussed in the first sub-case. Therefore, this sub-case is impossible.

- $s_{M(i)}$ only wins when submitting $A_{M(i)}$; $U^s_{M(i)} \ge 0 = \tilde{U}^s_{M(i)}$ since the seller only wins when submitting $A_{M(i)}$.

- $s_{M(i)}$ is not the winner when submitting both $\tilde{A}_{M(i)}$ and $A_{M(i)}$; In this sub-case, $\tilde{U}^s_{M(i)} = U^s_{M(i)} = 0$.

2) $\tilde{A}_{M(i)} < A_{M(i)}$: there are also four sub-cases.

- $s_{M(i)}$ is the winner when submitting both $\tilde{A}_{M(i)}$ and $A_{M(i)}$. we also assume the boundary pair is (b_{i_g}, s_{j_b}) when submitting $A_{M(i)}$. Therefore, we have $\tilde{A}_{j_{\tilde{y}}} < A_{j_y} < A_{j_b}$. Then we have: ① $\tilde{y} < y < g < b$; ② $\tilde{y} < g < y < b$; ③ $g < \tilde{y} < y < b$. For the sub-case ①, it will not affect the ask at/behind position g, which can obtain the same g and boundary pair. For the sub-case ②, we have $h_{i_g} \geq A_{j_g} \geq A_{j_{g-1}}$ in the original order. When submitting $\tilde{A}_{M(i)}$, s_{j_g} will be moved to position $g+1$. If $h_{i_{g+1}} \geq A_{j_g}$, $g+1$ will be the new aligned boundary since $h_{i_{g+2}} < h_{i_{g+1}} < A_{j_{g+1}}$. Since $A_{j_g} \leq h_{i_{g+1}} < A_{j_{g+1}}$, the new boundary of seller will be located at position $g+1$ (s_{j_g} is moved to position $g+1$). Therefore, the new payment $\tilde{P}^s = A_{j_g} \leq A_{j_b} = P^s$. If $h_{i_{g+1}} < A_{j_g}$, the position g is still the aligned boundary and the boundary pair is still (b_{i_g}, s_{j_b}). For sub-case ③, it will not affect the ask before position g, which can obtain the same g and boundary pair. Overall, we could obtain a new payment $\tilde{P}^s \leq P^s$ or the same price $\tilde{P}^s = P^s$ when the boundary pair has not changed. Therefore, the new utility $\tilde{U}^s_{M(i)} = \tilde{P}^s - C_{M(i)} \leq U^s_{M(i)}$.

- $s_{M(i)}$ only wins when submitting $\tilde{A}_{M(i)}$. Since $s_{M(i)}$ only wins when submitting $\tilde{A}_{M(i)}$, we could obtain a new payment $\tilde{P}^s = A_{\tilde{j}_\eta}$, which is less than $A_{M(i)} = C_{M(i)}$ since $s_{M(i)}$ loses by submitting $A_{M(i)}$. Therefore, we have $\tilde{U}^s_{M(i)} = \tilde{P}^s - C_{M(i)} \leq 0 = U^s_{M(i)}$.

- $s_{M(i)}$ only wins when submitting $A_{M(i)}$. This sub-case is similar to the second sub-case when $\tilde{A}_{M(i)} > A_{M(i)}$.

- $s_{M(i)}$ is not the winner when submitting both $\tilde{A}_{M(i)}$ and $A_{M(i)}$. In this sub-case, $\tilde{U}^s_{M(i)} = U^s_{M(i)} = 0$.

Considering the above situations, we always have the $U^s_{M(i)} \geq \tilde{U}^s_{M(i)}$, which can prove that sellers cannot improve its utility by providing a fake ask. The truthfulness of buyers can be proved in similar ways. □

3.5 Evaluation

3.5.1 Experiment Setup

We simulate the VEC using VISSIM, a classic open source framework for vehicular network simulation. It can analyze the operation of urban traffic and public transportation under various traffic conditions. With VISSIM, we can construct different scales of vehicular network, and obtain the driving data in real time, such as location, speed, vehicle type, power, and so on. We initialize the Luxembourg map and randomly load multiple vehicles to implement the scenario of the vehicular network. After the simulator has been running for a while, we select an urban-intersection with a radius of 500m in this map and extract the driving data (location, speed, and vehicle type/power) of vehicles driving in this urban-intersection at this moment.

Then we divide these vehicles into two types (client vehicles and vehicular edge nodes) based on the driving speed and vehicle type/power. It is reasonable since edge nodes always refer to vehicles with the remaining resources, especially slow-moving and parked vehicles in VEC architecture. We consider three attributes (location, reputation, and computing power) in our auction system. Since reputation could not be obtained from VISSIM, we randomly generate them in our experiment. In practice, the auctioneer has enough ability to collect the reputation using another server in real time [75].

In our experiments, we select 150 client vehicles and 150 vehicular edge nodes from the candidate vehicles, and record their attributes. The bids and asks are computed based on these attributes. We also vary the number of buyers or sellers for evaluating the performance of the auction mechanism under different number of buyers/sellers. We conduct multiple experiments on the window PC with 64-bit intel-core i5-6200U CPU at 2.3 GHz and 8 GB memory, and average the experimental results. The experimental setting and parameters are shown in Table 3.1

Table 3.1 Experimental Setting and Parameters

Parameters	Value
Vehicular network Simulator	VISSIM
The duration time of simulator	30 mins
The radius of simulation area	500m
Number of vehicles in simulation area	1000
Running by	Windows PC
Auction system implementation	JAVA
Number of client vehicles (n)	150
Number of vehicular edge nodes (m)	150
The range of bid/ask	(50, 100]
The value of attributes	(50, 100]
The range of weights	(0, 1]

3.5.2 Individual Rationality/Budget Balance

Individual rationality means the winning buyers will not pay the charges more than their bids, and the winning sellers will not obtain the payment less than their asks. Therefore, we run the auction mechanism between 50 client vehicles and 50 vehicular edge nodes, and output the final charges and payments, as shown in Fig. 3.4. The abscissa represents the i-th winning buyer-seller pair, and the ordinate represents the price. We can see that the bid of the winning buyer is always higher than the final charge, and the ask is always less than the final payment, which can prove the individual rationality.

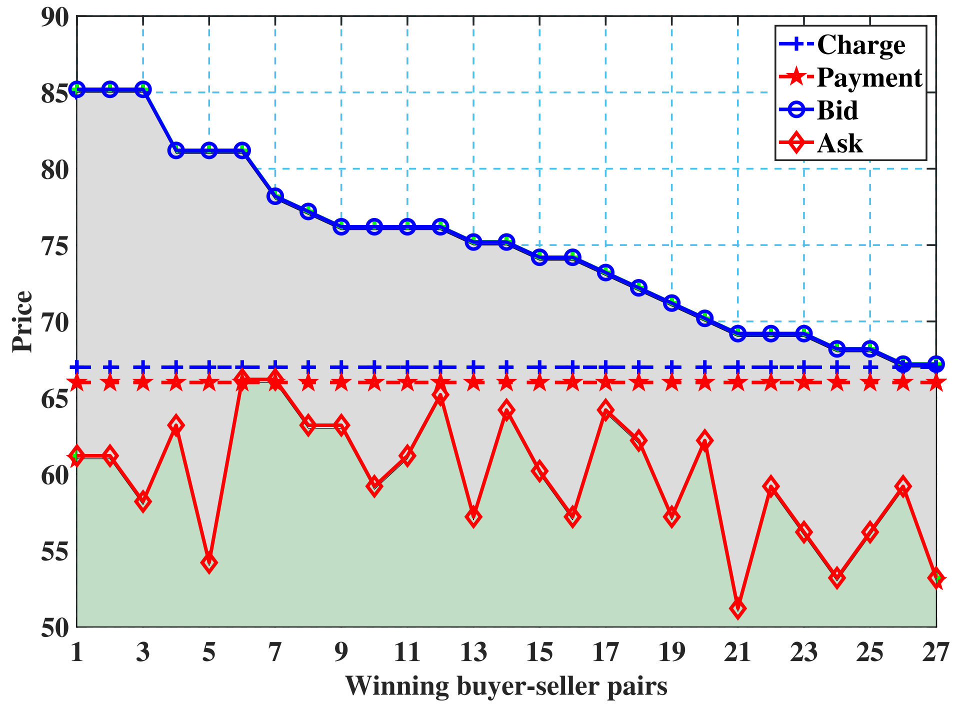


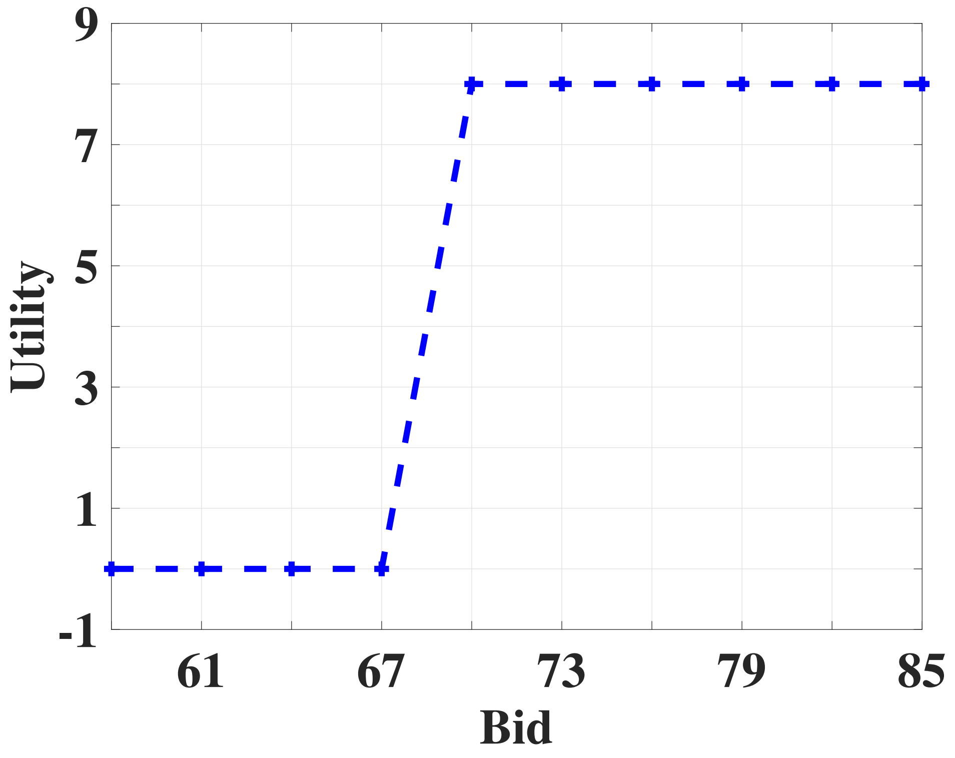
Fig. 3.4 The Individual Rationality and Budget Balance

As for the budget balance, we can see that the line representing the charge is higher than the line representing the payment. Therefore, the auctioneer will obtain additional income from the auction, which can prove the budget balance.

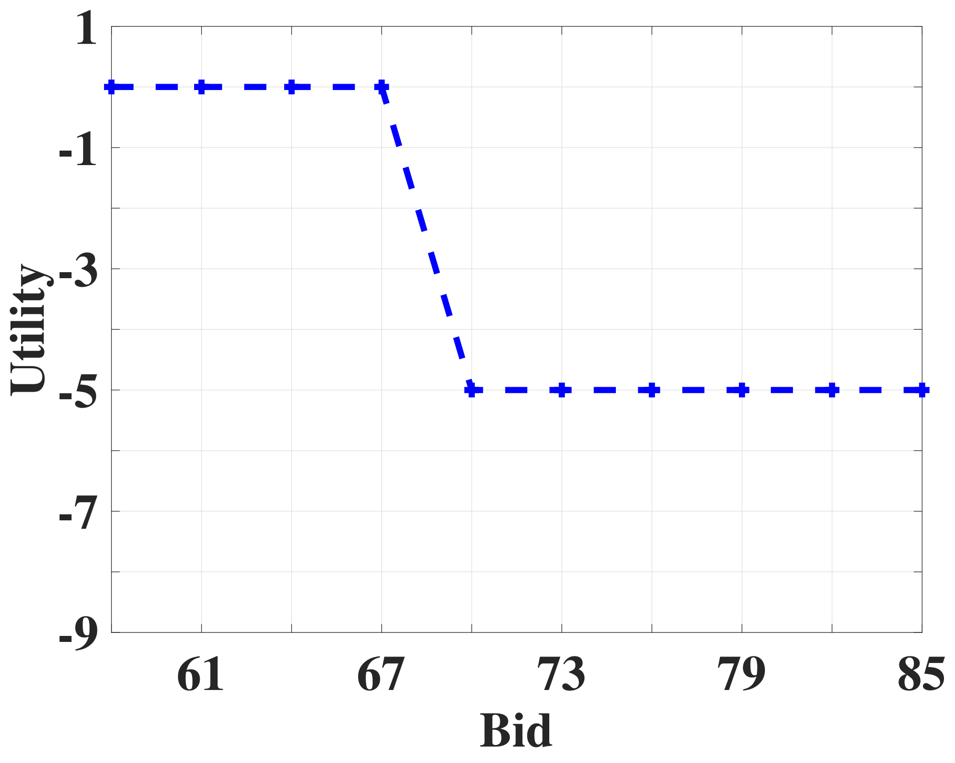
3.5.3 Truthfulness

We verify the truthfulness of the proposed multi-attribute double auction mechanism by the following experiments, as an auxiliary way to theoretical analysis (Section 3.4.4). We randomly select a buyer/seller from the final winning set and a buyer/seller who is not in the winning set. Then we change the bid/ask of this buyer/seller, containing the value greater than and less than the true valuation/cost. To provide a consistent environment for comparison, we keep all parameters unchanged in addition to the bid/ask of this buyer/seller.

Fig. 3.5 shows utilities when buyers provide different bids. Fig. 3.5a is the result when b_i is the final winning buyer. We can see that b_i has the maximum utility when it bids by the true valuation (76), and other bids will not bring more utilities. Fig. 3.5b shows the utilities when b_i is not the final winning buyer. The maximum utility of b_i is zero since they are not

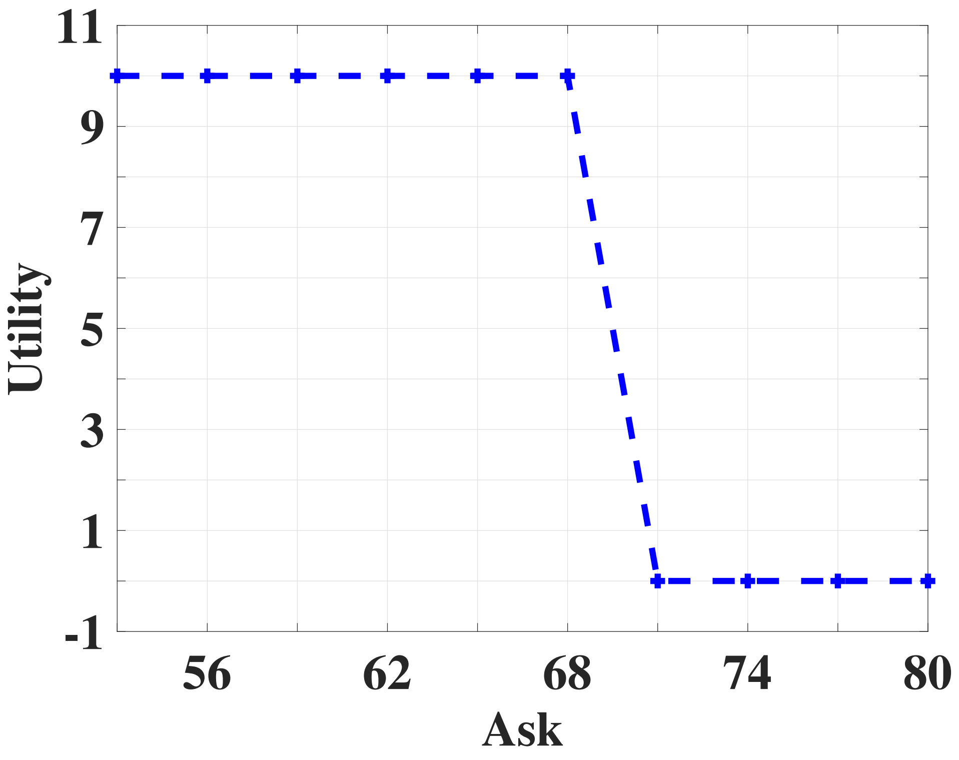


(a) Buyer $b_i \in B_w$

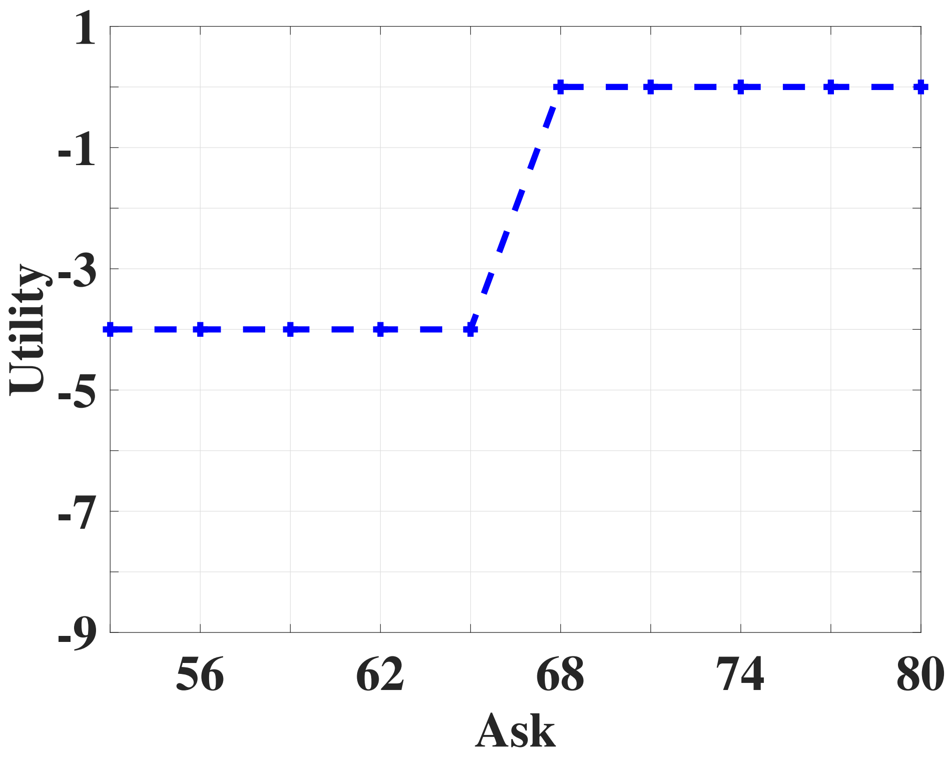


(b) Buyer $b_i \notin B_w$

Fig. 3.5 Truthfulness of buyers



(a) Seller $s_j \in S_w$



(b) Seller $s_j \notin S_w$

Fig. 3.6 Truthfulness of sellers

the final winner if it truthfully submits the bid (64). Although it can be the final winner when it changes the bid, it also cannot obtain the utility which is greater than zero.

Fig. 3.6 shows utilities when sellers provide different asks. Fig. 3.6a and Fig. 3.6b are two cases of sellers. From Fig. 3.6a and Fig. 3.6b we can see that these sellers have the maximum utilities when they offer the real cost (59, 71). In summary, we can verify the truthfulness of our mechanism.

3.5.4 Computational Complexity

We evaluate the impact of the number of buyers/sellers on running time by choosing a different number of buyers/sellers. In this experiment, we fix buyers/sellers at 100 and change the number of sellers/buyers from 50 to 150, respectively. Fig. 3.7 shows the results of the running time under a different number of buyers/sellers. The purple bar shows the running time when we fix the sellers, and the other is the results when we fix the buyers. We can see that both of them have a polynomial computation time, which can show the stability of our system under different scale of data.

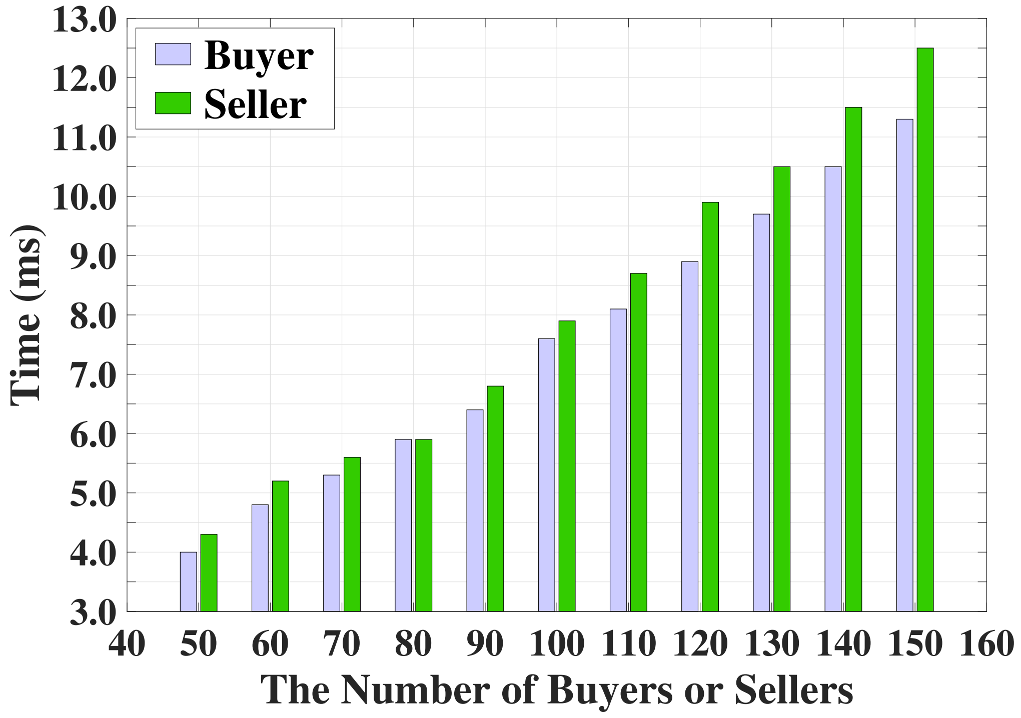
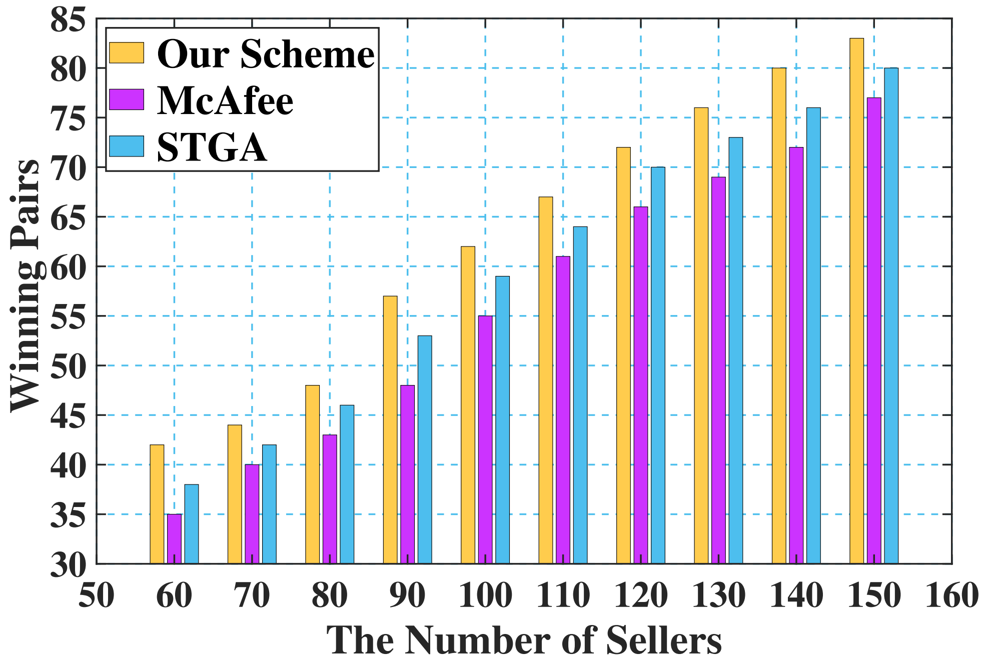
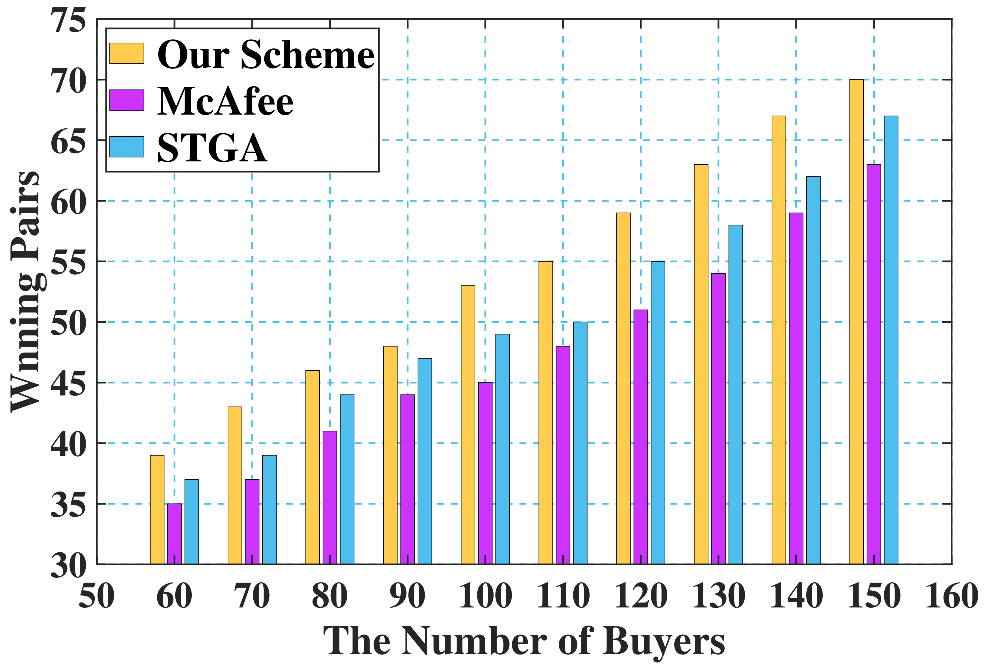


Fig. 3.7 Computational Complexity



(a) Comparison under Different Number of Sellers



(b) Comparison under Different Number of Buyers

Fig. 3.8 Performance under Different Number of Buyers/Sellers

3.5.5 Performance Comparison with Other Auction Mechanisms

We compare the proposed mechanism with McAfee auction and another edge-related double auction mechanism (STGA). To provide a consistent comparison environment, we use same dataset and parameters when running these mechanisms. We use the number of winning pairs as the performance metric to conduct the comparative experiment, as described in Section 3.4.3. we fix the number of buyers/sellers at 100 and vary the number of sellers/buyers from 60 to 150, respectively. The compare results are shown in Fig. 3.8a and Fig. 3.8b, where the yellow bar represents the winning pairs of our system, the blue bar represents the STGA, and the pink bar represents the McAfee auction. We can see that our scheme always has more winning pairs than other auction mechanisms, which can show the superiority of the proposed system's performance.

3.6 Conclusion

In this chapter, we design a multi-attribute based double auction mechanism in VEC scenario. The proposed auction mechanism not only considers the price but also considers non-price attributes when determining the winners. In addition, our auction mechanism could satisfy the following economic properties: computational efficiency, individual rationality, budget balance, and truthfulness. To verify our auction mechanism, we simulate the VEC scenario using VISSIM (a framework for running vehicle network simulation), and extract the driving data (location, speed, and vehicle type/power) of vehicles for the auction. Experimental results show the effectiveness and efficiency of our auction mechanism.

Chapter 4

Dynamic/Online Resource Allocation in Vehicular Edge Computing Network

4.1 Motivation

Among the proposed VECN architectures, vehicle-based VECN is a more practical architecture since it does not require the deployment of additional infrastructure (such as Road Side Unit), which treats vehicles with remaining resources as the edge nodes. Usually, not all vehicles have tasks that need to be done in a period of time, which means they have idle computing/storage resources. To decrease the waste of resources and obtain additional income, they (coined as vehicular edge nodes) are willing to perform tasks for other vehicles (coined as client vehicles).

During the task offloading process, edge nodes should provide their communication, computing, and storage resources to vehicles for task processing [73]. They always want to obtain rewards from vehicles for their costs, which is different for every edge node with different resources. From the perspective of the vehicles, they are willing to give different payments for their tasks even if they have the same task. Therefore, we should consider a resource allocation mechanism for the vehicular edge computing network which can meet the requirements of edge nodes and vehicles [8].

Since edge nodes and vehicles have their own interests, a double auction scheme which could consider interests of both parties is needed in vehicular edge computing network scenario. During the double auction process, vehicles submit their bids and edge nodes give their asks to trusted auctioneer. Then this auctioneer executes winner determination and pricing algorithms to determine the final winners and price. Since edge nodes and vehicles do not know the bids/asks of other vehicles and edge nodes, the double auction can give a fair trading platform.

However, we introduced a multi-attribute based static/offline double auction mechanism in the last chapter. Considering the dynamic flexibility in VECN, buyers and sellers can join or leave the auction mechanism at any time according to their own wishes, which will not be controlled by the auctioneer. Therefore, in the edge scenario, the auctioneer needs to make decisions in a more dynamic environment, which could be achieved by the online auction mechanism.

Moreover, these schemes only use price for constructing the matching between buyers and sellers. However, in VECN scenario, communication quality can be poor because of mobility of vehicles, which means long-distance communication will have a large delay. Even if there exists a matching between a vehicle and an edge node according to existing edge-related auction mechanisms, the winning edge node still cannot provide services for the winning vehicle if there is a long distance between them (e.g., cannot complete the transmission of massive data between vehicles and edge nodes in time). Therefore, location is an important factor to construct reasonable matching in VECN auction scenario. In addition to location, other non-price attributes are also important in VECN auction scenario (as discussed in Section 4.3).

In this Chapter, we design an online resource auction scheme for an edge with non-price attributes, which is in line with a dynamic situation that vehicular edge nodes and client vehicles can at any time join or leave the auction system. Moreover, in addition to the bids/asks, we also consider the non-price attributes when constructing the matching between buyers and sellers. To the best of our knowledge, this is the first work to consider the online auction mechanism for VECN scenario. We simulate VECN using a vehicular network simulator for verifying our work. Then we evaluate our work from the perspective of running time, charges/rewards, and utilities of users. Experimental result shows that our scheme could meet properties of computational efficiency, individual rationality, budget-balance, and truthfulness.

The main contributions of this paper are summarized as follows:

- We propose an online double auction mechanism in the VECN scenario, which is in line with the dynamic situation that users can join or leave the auction system at any time.

- We propose a one-to-one matching algorithm based on the non-price attributes, which is more suitable for the VECN scenario.

- We simulate VECN using a vehicular network simulator for verifying our work. Experimental result shows that our scheme could meet desired properties of the general online double auction.

The rest of this paper is organized as follows.In section 4.2, we will give the related work about auction mechanisms in VECN. Section 4.3 shows the importance of non-price attributes in VECN auction scenario. The auction model is shown in section 4.4. Section 4.5 presents the proposed online double auction mechanism and theoretical analysis. The experimental results are shown in section 4.6. Finally, we conclude the paper in section 4.7.

4.2 Related Work

4.2.1 Applications in VECN

Nowadays, massive applications in the vehicular network will generate a lot of data in a short time. For example, in some driver assistance systems, vehicles need to process massive data generated from vehicle sensors (camera, radar, and so on), which is difficult to do alone by resource-limited vehicles. Vehicular Edge Computing Network (VECN) is proposed to achieve real-time data processing. Many works have studied this new paradigm and proposed some architectures in vehicular network scenario. In [32], the authors discussed how to integrate the mobile edge computing into the cellular vehicular network for providing flexible vehicle-related services. To cope with the large content volume in the vehicular network which is time-varying, location-dependent, and delay-constrained, some works proposed the edge computing-based caching [68] and edge computing-based content dissemination framework for providing high-quality services [20]. Applications in VECN have also been investigated, such as crowdsourcing-based information confusion applications and location-based subscribing applications.

In addition to the architecture and applications, how to allocate resources is an important issue to make the VECN deploy in practice [56]. Zhang *et al.* [72] presented an efficient predictive relegation scheme, in which tasks are offloaded to edge nodes by predictive relay transmissions. In [13], the authors jointly considered the load balancing and offloading, and studied the resource allocation issues by proposing a optimal edge node selection and offloading algorithm. However, few works study the auction mechanisms in this new paradigm. During the task offloading process, vehicular edge nodes should provide their communication, computing, and storage resources to vehicles for task processing [46]. Therefore, they always want to obtain the payments for their costs. From the perspective of the vehicles, they are willing to give different payments for their tasks. Therefore, a double auction scheme is needed in vehicular edge computing network scenario.

4.2.2 Auction Mechanisms in VECN

Few works consider the auction mechanisms in vehicular edge computing network scenario, which is one of the factors affecting the development of VECN architecture. Fair trading can attract more vehicles to provide services for other vehicles. In [36], a reverse auction scheme was studied to encourage vehicles to share resources. However, they only consider the interests of one party, which cannot provide a fair auction environment for all parties. In addition to the auction mechanisms in vehicular network scenario, there are some edge-related auction mechanisms towards other scenarios in the past few years. Wang *et al.* [63] designed a multi-round auction mechanism between edge nodes and mobile devices, which needs more time to complete the auction process. In [24, 23], the authors considered deploying edge computing into mobile blockchain, and proposed an auction mechanism providing a fair trading environment. However, this mechanism could not meet the basic property of budget balance, which is not in line with the actual situation. In [57], the authors studied the resource allocation in industrial IoT, and proposed two double auction schemes. They assumed each edge node can serve multiple mobile devices, which is not suitable for the VECN scenario.

However, almost no works study the online auction mechanisms in edge computing based scenarios. Current edge-related auction mechanisms do not consider the dynamics of buyers/sellers and assume that the auctioneer knows all information about sellers/buyers before the auction begins, which is not suitable to be applied to the VECN scenario since vehicular edge nodes and client vehicles can join/leave the auction system at any time. Therefore, we propose a multi-attribute based online resource auction mechanism for VECN scenario.

4.3 Attribute Constraints in VECN Online Auction

As discussed in the previous section, some physical limitations, such as poor wireless links and diverse task types, exist in VECN, which causes the result that not all vehicular edge nodes could provide services for client vehicles. Some attribute constraints should be considered when constructing the matching between vehicular edge nodes and client vehicles.

In this section, we use the following attributes as the examples to show why we should consider the attribute constraints when designing a reasonable online auction mechanism in VECN scenario.

- Location: If a client vehicle wants to outsource an urgent task (safety-related task) and we assign a vehicular edge node which is far from the client vehicle, this allocation scheme will waste a lot of resources and time on the data transmission [45], which is obviously not suitable for performing urgent tasks in VECN. Therefore, location is an important factor to construct reasonable matching in VECN.
- Reputation: Due to the open and fast-changing environments of vehicular network, some untrusted edge nodes may disrupt the task, such as reporting an incorrect result to client vehicles. Therefore, vehicles should choose the edge nodes with higher reputation when performing some safety-related tasks.
- Computing power: Vehicles should choose edge nodes with higher computing power when performing some urgent tasks, which means these tasks have the minimum demand constraints toward computing power.

Since other important and relevant attributes also need to be considered, we use the general notation to represent attribute requirements and values for covering all attributes. For the convenience of description, we use $q_j = (q_j^1, q_j^2, q_j^3, ..., q_j^k)$ to represent vehicular edge node's values of location, reputation, computing power and k-th attribute. Similarly, $p_i = \{(p_i^{11}, p_i^{12}), p_i^2, p_i^3, ..., p_i^k\}$ represents client vehicle's attribute requirements. p_i^{11} and p_i^{12} are location and acceptable distance. p_i^2, p_i^3, and p_i^k are acceptable reputation, computing power, and k-th attribute respectively.

4.4 Auction Model and Problem Formulation

We will give the auction model and economic properties of the proposed scheme.

4.4.1 Auction Model

We adopt a general VECN system, which contains a cloud server, some base stations for data transmission, and a large amount of vehicles, the dynamic scenario of VECN as shown in Fig. 4.1. In our system, vehicles that have idle computing and storage resources could serve as vehicular edge nodes (sellers) to provide services for client vehicles (buyers). Note that a vehicle can serve as the buyer and seller at the same time if it cannot deal with its task which

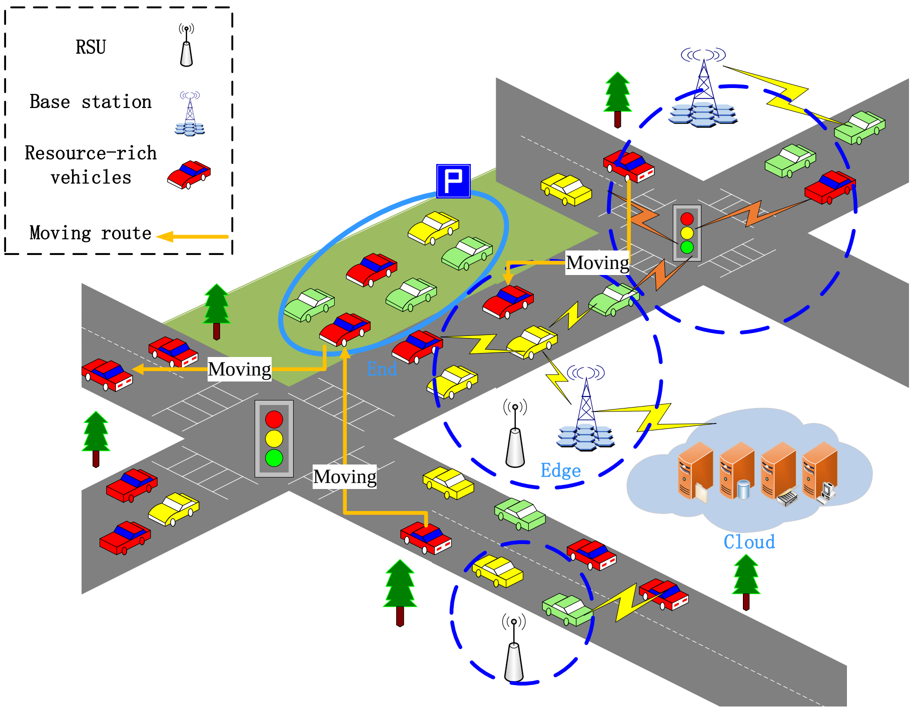


Fig. 4.1 The Architecture of VECN about Dynamic Scenario

needs lots of computing resources but its remaining resources can meets the requirements of other vehicles. The role of a vehicle could change according to their requirements (buying resources or selling resources) after the vehicle finished the last auction process. Then we will introduce the auction participants and the auction process.

Client Vehicles (Buyers)

Let $b_i \in \mathscr{B}$ represent a client vehicle who has a task that needs to be performed. In the dynamic auction scenario, a vehicle b_i could enter/leave the market at any time. Therefore, we use a_i^b/d_i^b to denote arrival/departure time of vehicle b_i. Since a vehicle could not wait forever, we assume the waiting time is bounded by a constant K, which meets $d_i^b - a_i^b \leq K$. When the vehicle has a task and enters the auction market, it will submit the bid h_i for this task and corresponding attribute requirements p_i.

Vehicular Edge Nodes (Sellers)

Let $s_j \in \mathscr{S}$ represent a vehicular edge node who wants to sell its resources. Similar to client vehicles, we use a_j^s/d_j^s to denote arrival/departure time of vehicular edge node s_j

($d^s_j - a^s_j \leq K$). The vehicular edge node s_j will report its ask l_j for providing their resources and the attribute values q_j.

Auction Process

Different from other networks with fast transfer speed, the poor wireless link of the vehicular network could not support the long-distance transmission [59]. Therefore, similar to the resource assignment scheme proposed in [76] where the base station collects vehicles' tasks and assigns them to vehicular edge nodes under its coverage, we also regard a base station as the auctioneer. Edge nodes and vehicles will tell base station when they enter/leave the auction system, and trading process will happen between vehicular edge nodes and client vehicles under the same base station, which can construct a distributed auction platform. Since vehicular edge nodes are mainly slow-moving and parked vehicles, they are relatively static compared to client vehicles. Therefore, vehicular edge nodes and client vehicles have enough time to achieve the trade under a base station.

Similar to other online auction mechanism [64], we also divide the auction process into continuous time slots $T = \{1, 2, ..., t, ...\}$. The vehicular edge nodes/client vehicles will choose to sell their resources/publish their tasks at different slots, which means the arrival and departure time of buyers and sellers belongs to different time slots ($a^b_i/a^s_j, d^b_i/d^s_j \in T$). And the waiting time ($[a^b_i, d^b_i]/[a^s_j, d^s_j]$) of buyers and sellers will last for one or multiple time slots.

At each slot t, vehicles will choose to buy resources or sell resources (buyers or sellers). A buyer b_i will send its bid h_i, attribute requirements p_i, and arrival/departure time a^b_i/d^b_i. Similarly, a seller s_j will send its ask l_j, attribute values q_j, and arrival/departure time a^s_j/d^s_j. At each slot t, the auctioneer will collect the information of active users arriving in the previous slot ($a^b_i/a^s_j < t < d^b_i/d^s_j$) and newly arrived users ($a^b_i/a^s_j = t$). Then it performs the auction scheme to select winning buyers $\mathscr{B}^t_w$, winning sellers $\mathscr{S}^t_w$, and final price (charges $P^b(t)$ from buyers and rewards $P^s(t)$ for sellers). Some important notations are shown in Table 4.1. We define v_i as the true valuation of client vehicle b_i if the task is completed and c_j as the trust cost of vehicular edge node for providing its resources. Therefore, the utility U^b_i of the buyer b_i and the utility U^s_j of the seller s_j are denoted as follows:

$$U^b_i = \begin{cases} v_i - p^b_i(t), & \text{if } b_i \in \mathscr{B}^t_w \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

$$U^s_j = \begin{cases} p^s_j(t) - c_j, & \text{if } s_j \in \mathscr{S}^t_w \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

Table 4.1 Important Notations

Symbol	Description
b_i	buyer (a client vehicle who wants to process some tasks)
s_j	seller (a vehicular edge node)
p_i	attribute requirements of the *i-th* client vehicle
q_j	attribute values owned by the *j-th* vehicular edge node
$d(\cdot,\cdot)$	distance between the buyer and the seller
a_i^b	arrival time of the client vehicle b_i
d_i^b	departure time of the client vehicle b_i
a_j^s	arrival time of the vehicular edge node s_j
d_j^s	departure time of the vehicular edge node s_j
K	boundary of waiting time
T	total time slots
h_i	bid of the client vehicle b_i
l_j	ask of the edge node s_j
$\mathscr{B}_w^t$	set of the winning buyers at slot t
$\mathscr{S}_w^t$	set of the winning sellers at slot t
$P^b(t)$	charges from the winning buyers
$P^s(t)$	rewards to the winning sellers
v_i	true valuation of client vehicle b_i
c_j	true cost of vehicular edge node s_j
$p_i^b(t)$	price charged from the client vehicle b_i
$p_j^s(t)$	reward to the edge node s_j
U_i^b	utility of the client vehicle b_i
U_j^s	utility of the edge node s_j

4.4.2 Economic Properties

Definition 1. ***(Computational Efficiency)*** *Computational efficiency means the proposed online auction mechanism has a polynomial time complexity.*

Definition 2. ***(Individual Rationality)*** *The multi-attribute based online double auction mechanism satisfies individual rationality when each buyer's bid is greater than the charge and each seller's ask is less than the reward:*

$$\begin{aligned} h_i &\geq p_i^b(t),\ \forall\, b_i \in \mathscr{B}_w^t \\ l_j &\leq p_j^s(t),\ \forall\, s_j \in \mathscr{S}_w^t \end{aligned} \tag{4.3}$$

Definition 3. ***(Budget Balance)*** *Our auction mechanism meets the property of budget balance when total charges from winning buyers are greater than total rewards to winning sellers:*

$$\sum_{t=1}^{\infty} \sum_{b_i \in \mathscr{B}_w^t} p_i^b(t) \geq \sum_{t=1}^{\infty} \sum_{s_j \in \mathscr{S}_w^t} p_j^s(t) \tag{4.4}$$

Definition 4. ***(Truthfulness)*** *If the buyer and seller are trusted, they will submit the real bid/ask information to the auctioneer. However, some users are selfish or even malicious, and these users will give the fake information to obtain more benefits. Our online auction scheme ensures they could obtain the maximum utilities if they give the real information to the auctioneer.*

Online double auction mechanisms that meet the above properties could provide a feasible and fair auction environment for vehicular edge nodes and client vehicles. In the next section, we will propose an multi-attribute based online double auction mechanism which can strictly satisfy the properties of individual rationality, computational efficiency, budget balance, and truthfulness.

4.5 Online Double Auction Mechanism with Non-price Attributes

We will introduce the proposed online double auction mechanism AucM(χ, ψ), which includes the one-to-one matching $\chi(b_i, s_j)$ and the pricing scheme $\psi(p_i^b, p_j^s)$ under the online VECN auction scenario. The proposed auction mechanism is based on a general online auction framework [9, 71], which firstly removes the lower-price users based on the admission price at the beginning of each slot and applies the McAfee auction to the remaining active users.

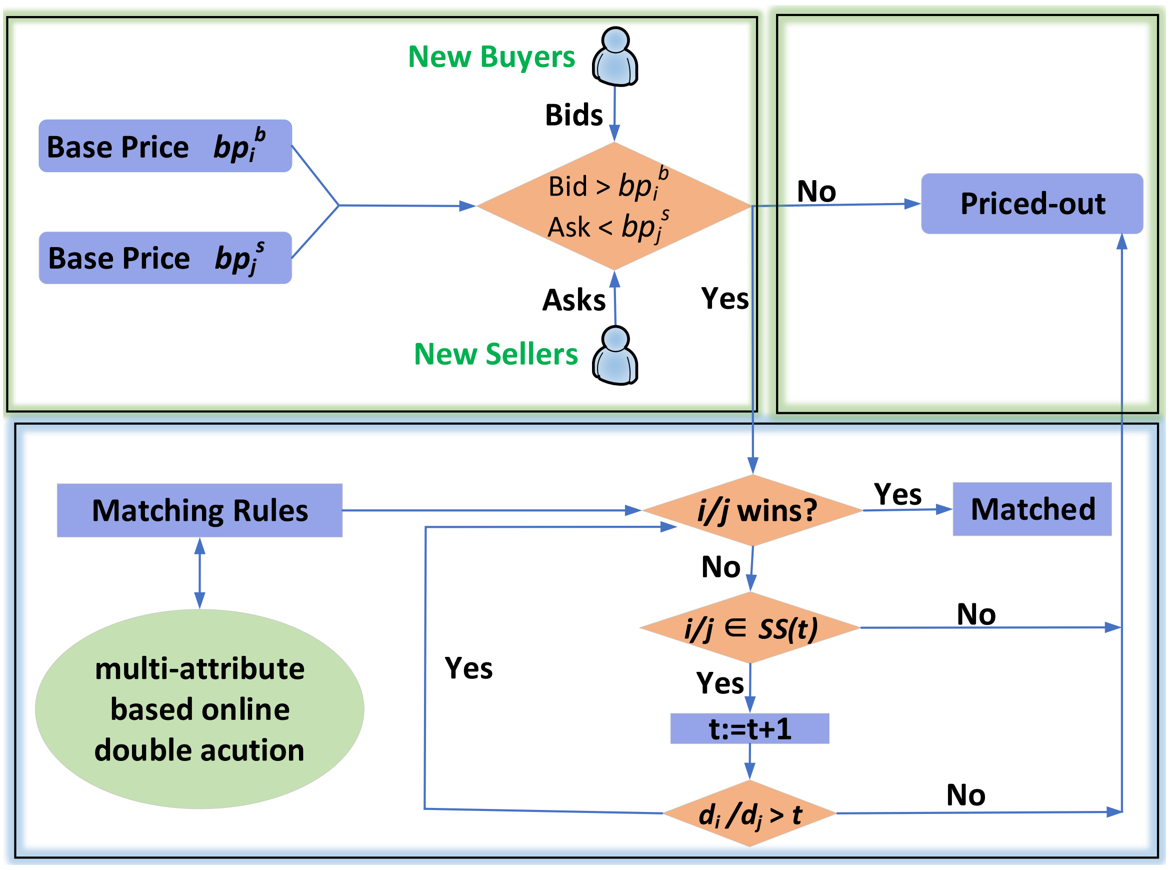


Fig. 4.2 The Framework of Online Double Auction for VECN

However, the existing framework that does not consider reasonable matching (for example, not all edge nodes could provide services for one client vehicle) cannot be directly used in VECN scenario. Therefore, our mechanism further extends the existing framework by designing a new winner selection rule and adding a new matching rule based on vehicles' attributes, which can obtain reasonable matching and more winning buyer-seller pairs. As shown in Fig. 4.2, the proposed auction mechanism includes three main stages: Active Users Selection Stage, Candidate Winners Selection Stage, as well as Matching and Pricing Stage. Firstly, we will detailed present the proposed online auction scheme. Then we will give the theoretical analysis to demonstrate that the online auction mechanism could satisfy the proposed economic properties.

4.5.1 Active Buyer/Seller Selection

As a dynamic auction scenario, client vehicles and edge nodes will join the auction system at different time slots randomly. Therefore, at time slot t, auctioneer should firstly decide the active buyers and sellers which will participate in this round of auction, including buyers/sellers who arrived in the previous slot ($a_i^b/a_j^s < t < d_i^b/d_j^s$) and are still in the survivor set $SS(t-1)$, as well as newly arrived users ($a_i^b/a_j^s = t$) which are not ruled out by the auction system. If there are not enough active buyers/sellers and no trade occurs in the previous time slot $t-1$, these active buyers/sellers will be put into the survivor set $SS(t-1)$. Otherwise, $SS(t-1)$ is an empty set, as shown in Algorithm 5 (Line 14-16).

For the newly arrived users, we should set a base price bp to rule out some users before the auction begins since some buyers (sellers) may give the lower (higher) prices to disrupt the auction market, which means a buyer (seller) whose bid (ask) is lower (higher) than its base price bp will be excluded to maintain a stable market (coined as failed set). Then the remaining users (including users in $SS(t-1)$) will be regarded as the active users in time slot t for auction. After the auction is completed in this time slot, users can be in one of the three sets: winning set, survivor set, or failed set. We denote $HS(t^{'}), t^{'} < t$ as the active users in the time slot $t^{'}$. Assuming a buyer b_i whose active time is (a_i^b, d_i^b), its base price bp_i^b is calculated by adding this user to the auction in time slot $t^{'}$ ($t^{'} \in \{d_i^b - K, a_i^b - 1\}$).

- If $d_i^b - a_i^b = K$, $t^{'}$ will not exist since $t^{'} \in \{d_i^b - K, a_i^b - 1\}$. Therefore, we set $bp_i^b = 0$.

- If $d_i^b - a_i^b < K$:

$$bp_i^b = \begin{cases} +\infty, & b_i \text{ loses } \& \ b_i \notin SS(t^{'}) \text{ for any } t^{'} \\ 0, & b_i \in SS(t^{'}) \text{ for any } t^{'} \\ \max_{\{t^{'},\ b_i \notin SS(t^{'})\}} p_i^b(t^{'}), & \text{otherwise} \end{cases} \tag{4.5}$$

For a buyer s_j whose auction time is (a_j^s, d_j^s), its base price bp_j^s is defined as follows:

- If $d_j^s - a_j^s = K$, $t^{'}$ will not exist since $t^{'} \in \{d_j^s - K, a_j^s - 1\}$. In this case, we set $bp_j^s = +\infty$.
- If $d_j^s - a_j^s < K$:

$$bp_j^s = \begin{cases} 0, & s_j \text{ loses } \& \ s_j \notin SS(t^{'}) \text{ for any } t^{'} \\ +\infty, & s_j \in SS(t^{'}) \text{ for any } t^{'} \\ \min_{\{t^{'},\ s_j \notin SS(t^{'})\}} p_j^s(t^{'}), & \text{otherwise} \end{cases} \tag{4.6}$$

where $p_i^b(t^{'})/p_j^s(t^{'})$ is the charge/reward when joining the auction in $t^{'}$ using the following pricing algorithm (Section 4.5.3).

4.5.2 Candidate Winners Selection

Based on the previous step, we can rule out some users according to the base price bp, and the remaining users (including users in $SS(t-1)$) will be regarded as the active users in time slot t for auction, which can be divided into active buyers $\mathscr{B}(t)$ and active sellers $\mathscr{S}(t)$. The existing online double auction framework [9, 71] uses McAfee auction to select the candidate winning buyers and sellers from these active users. McAfee auction firstly sorts the bids of buyers by the descending order and the asks of sellers by the ascending order. Then it will find the large position where buyer's bid is greater than seller's ask at the same position, and regard the buyers/sellers before this position as the final winners, which does not consider the one-to-one matching of these winners.

Moreover, this scheme may lose some reasonable winners since it does not have a cross-comparison between the buyers and sellers. For example, we assume there exists seven buyers $\{b_1, b_2, b_3, b_4, b_5, b_6, b_7\}$ with the bids $\{15, 14, 12, 9, 9, 6, 4\}$ and seven sellers $\{s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$ with the asks $\{2, 5, 7, 8, 11, 12, 13\}$. Based on the McAfee algorithm, we could obtain the candidate winning buyers $\{b_1, b_2, b_3, b_4\}$ and candidate wining sellers

Algorithm 5: Online Auction Algorithm

```
Input: Newly arrived Buyers $\mathscr{B}^a(t)$ and Sellers $\mathscr{S}^a(t)$ in $t$; Survivor Set $SS(t-1)$;
       Time slots $T$;
Output: The winning buyers $\mathscr{B}_\omega$ and sellers $\mathscr{S}_\omega$; one-to-one matching $\mathscr{M}(t)$; the
        charges $P^b$ and rewards $P^s$
1  $t \leftarrow 0$
2  while $t \leq T$ do
3      Active Buyers $\mathscr{B}(t) \leftarrow SS_{\mathscr{B}}(t-1)$;
4      Active Sellers $\mathscr{S}(t) \leftarrow SS_{\mathscr{S}}(t-1)$;
5      for each buyer $b_i \in \mathscr{B}^a(t)$ do
6          Calculating base price $bp_i^b$ based on Equation 4.5;
7          If $h_i \geq bp_i^b$, add $b_i$ into $\mathscr{B}(t)$;
8      end
9      for each seller $s_j \in \mathscr{S}^a(t)$ do
10         Calculating base price $bp_j^s$ based on Equation 4.6;
11         If $l_i \leq bp_j^s$, add $s_j$ into $\mathscr{S}(t)$;
12     end
13     $SS_{\mathscr{B}}(t) \leftarrow \emptyset$; $SS_{\mathscr{S}}(t) \leftarrow \emptyset$;
14     if $\min\{|\mathscr{B}(t)|, |\mathscr{S}(t)|\} < 2$ then
15         $SS_{\mathscr{B}}(t) \leftarrow \mathscr{B}(t)$; $SS_{\mathscr{S}}(t) \leftarrow \mathscr{S}(t)$;
16         Continue;
17     end
18     else
19         Sorting asks of sellers in $\mathscr{S}(t)$ by an ascending order: $l_{j_1} \leq l_{j_2}...$;
20         Similarly, Sorting bids of buyers in $\mathscr{B}(t)$ by an descending order: $h_{i_1} \geq h_{i_2}...$;
21         Finding the largest position $g$ (Aligned boundary) which meets: $h_{i_g} \geq l_{j_g}$;
22         Finding the largest $\alpha$: $h_{i_\alpha} \geq l_{j_g}$ and the largest $\beta$: $h_{i_g} \geq l_{j_\beta}$;
23         // Final boundary pair
24         $(\theta, \eta) \leftarrow (\alpha, g)$ or $(g, \beta)$: Selecting one boundary with more users in
            $(\mathscr{B}_\alpha(t), \mathscr{S}_g(t))$ and $(\mathscr{B}_g(t), \mathscr{S}_\beta(t))$; // $\mathscr{B}_\alpha(t)$ is the first $\alpha$ elements in
            sorted $\mathscr{B}(t)$
25         $p_b^{candi} \leftarrow h_{i_\theta}$, $p_s^{candi} \leftarrow l_{j_\eta}$;
26         $\mathscr{B}_c(t) \leftarrow \mathscr{B}_{\theta-1}(t)$, $\mathscr{S}_c(t) \leftarrow \mathscr{S}_{\eta-1}(t)$;
27         Constructing the bipartite graph: $G = (\mathscr{B}_c(t) \cup \mathscr{S}_c(t), \mathscr{B}_c(t) \leftrightarrow \mathscr{S}_c(t))$;
28         $\mathscr{M}(t), \mathscr{B}_w^t, \mathscr{S}_w^t \leftarrow F(G)$; // $F(\cdot)$ is different according to the purposes
29         $p_i^b(t) = \max\{p_b^{candi}, bp_i^b\}, \forall\, b_i \in \mathscr{B}_w^t$;
30         $p_j^s(t) = \min\{p_s^{candi}, bp_j^s\}, \forall\, s_j \in \mathscr{S}_w^t$;
31         Output $(\mathscr{B}_w^t, \mathscr{S}_w^t, \mathscr{M}(t), P^b, P^s)$;
32         $t \leftarrow t+1$;
33     end
34 end
```

$\{s_1, s_2, s_3, s_4\}$. However, since the buyer b_5 has the bid 9 which is equal to the bid of buyer b_4 and greater than all asks of candidate winning sellers, it also can be a candidate winning buyer. Since we consider the attributes in our model, if the buyer b_5 has fewer attribute requirements than other buyers, it may replace another buyer as the final winner when constructing the one-to-one matching based on attributes (as shown in the next section).

Therefore, a reasonable candidate winners selection algorithm is needed for increasing the number of candidate winning buyers and sellers, as shown in Algorithm 5: First of all, we sort bids/asks by the descending/ascending order, and find the largest position where the buyer's bid is greater than the seller's ask, which is coined as the aligned boundary. Different from McAfee double auction which defines the aligned boundary as the final boundary of candidate winners, we will relax the aligned boundary for finding other reasonable candidate winners (Line 22). We can obtain two boundary pairs with the previous step, and regard the boundary pair which has more users as the final boundary. All users in front of the boundary pair will be regarded as the candidate winning vehicular edge nodes $\mathscr{S}_c(t)$ and client vehicles $\mathscr{B}_c(t)$. The bid and ask of boundary pair will be regarded as the candidate charge p_b^{candi} and reward p_s^{candi}.

4.5.3 Multi-attribute based Matching and Pricing

In the previous section, we could obtain the candidate winning vehicular edge nodes $\mathscr{S}_c(t)$ and client vehicles $\mathscr{B}_c(t)$ based on the candidate winners selection algorithm. However, these users are not the final winners since they may not meet the attribute constraints between vehicular edge nodes and client vehicles. For example, if the attribute requirements of a client vehicle are greater than the attribute values of all vehicular edge nodes in candidate winners, it will be deleted from the winning set, which means no vehicular edge node could serve this vehicle.

Therefore, we should design a reasonable multi-attribute based matching and pricing algorithm for vehicular edge nodes and client vehicles to determine the final winners and charges/rewards, which is the biggest difference between our mechanism and the existing framework since current framework assumes that all sellers could provide services for every buyer and does not consider the matching. The concrete matching scheme is shown as follows.

Finding All matching between Candidate Winning Buyers and Sellers

According to the dynamic/online double auction model, a client vehicle has the attribute requirements $p_i = \{(p_i^{11}, p_i^{12}), p_i^2, p_i^3, ..., p_i^k\}$ and a vehicular edge node has its own attribute values $q_j = (q_j^1, q_j^2, q_j^3, ..., q_j^k)$. A vehicular edge node could provide the services for a client vehicle if and only if the vehicular edge node's attribute values are greater than attribute requirements of client vehicle:

$$(d(p_i^{11}, q_j^1) \leq p_i^{12}) \cap (q_j^2 \geq p_i^2) \cap (q_j^3 \geq p_i^3) ... \cap (q_j^k \geq p_i^k) \tag{4.7}$$

where $d(p_i^{11}, q_j^1)$ is the distance between the client vehicle and the vehicular edge node.

We model the matching issue as the construction of a bipartite graph $G = (\mathscr{B}_c(t) \cup \mathscr{S}_c(t), \mathscr{B}_c(t) \leftrightarrow \mathscr{S}_c(t))$. The vertices are vehicular edge nodes $\mathscr{S}_c(t)$ and client vehicles $\mathscr{B}_c(t)$, and the edges $(\mathscr{B}_c(t) \leftrightarrow \mathscr{S}_c(t))$ represent the matching between the vehicular edge nodes and client vehicles. We could construct an edge $(b_i \leftrightarrow s_j)$ between the client vehicle b_i and the vehicular edge node s_j if they meet the attribute constraints in Equation 4.7 (coined as a matching).

Constructing the One-to-one Matching

The bipartite graph $G = (\mathscr{B}_c(t) \cup \mathscr{S}_c(t), \mathscr{B}_c(t) \leftrightarrow \mathscr{S}_c(t))$ shows the matching between the candidate winning vehicular edge nodes and client vehicles, where one vehicular edge node (client vehicle) could have multiple edges with client vehicles (vehicular edge nodes) if they meet the attribute constraints in Equation 4.7. However, in our model, we consider that one vehicular edge node can only provide services for one client vehicle since the vehicular edge node has limited resources compared with cloud server.

Therefore, we need to find the one-to-one matching $\mathscr{M}(t)$ from bipartite graph $G = (\mathscr{B}_c(t) \cup \mathscr{S}_c(t), \mathscr{B}_c(t) \leftrightarrow \mathscr{S}_c(t))$. We use $F(\cdot)$ to represent the algorithm to look for the unique matching, which can be different according to the objective. For example, if we want to find the buyer-seller pairs as much as possible, we could use the maximum matching algorithm (such as Hungary Algorithm). If we want to increase the resource utilization, we firstly allocate corresponding weights for all matching and find the unique matching using the maximum weighted matching algorithm.

Based on the algorithm $F(\cdot)$, we could obtain the one-to-one matching $\mathscr{M}(t)$, which includes the final winning vehicular edge nodes $\mathscr{S}_w^t$ and client vehicles $\mathscr{B}_w^t$. Then we could determine the final charges from winning client vehicles $\mathscr{B}_w^t$ and rewards for final winning vehicular edge nodes $\mathscr{S}_w^t$:

$$p_i^b(t) = \max\{p_b^{candi}, bp_i^b\}, b_i \in \mathscr{B}_w^t \tag{4.8}$$

$$p_j^s(t) = \min\{p_s^{candi}, bp_j^s\}, s_j \in \mathscr{S}_w^t \tag{4.9}$$

where p_b^{candi} and p_s^{candi} are the candidate payment and reward calculated in Section 4.5.2, respectively. In theoretical analysis part, we will prove that this pricing scheme could meet the truthfulness.

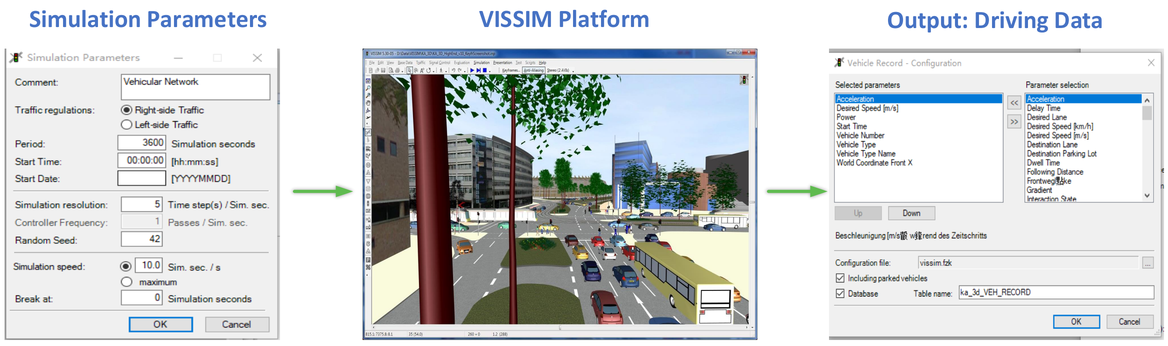
4.5.4 Theoretical Analysis

We demonstrate the multi-attribute based online auction mechanism satisfies the basic properties defined in Section 4.4.2 by the theoretical analysis.

Theorem 5. *The total time complexity is $O(\chi(\Gamma + \chi \log(\chi)))$, where Γ represents the time complexity of the one-to-one matching algorithm $F(\cdot)$ and $\chi = \max\{m_2 + n_2, \chi_2\}$.*

Proof. We assume there are m_1 active client vehicles and n_1 vehicular edge nodes at time slot t. In the candidate winners selection phase, the sorting operations need a time complexity of $O(m_1 \log(m_1) + n_1 \log(n_1))$, and the searching operations need a time complexity of $O(\chi_1)$, where χ_1 is less than $\min\{m_1, n_1\}$. For the bipartite graph construction and one-to-one matching algorithm $F(\cdot)$, the time complexity is dynamic according to the concrete algorithm. In this paper, we use a general notation Γ to represent its time complexity. We assume there are m_2 newly arrived client vehicles and n_2 vehicular edge nodes at time slot t. For each newly arrived user, we need to obtain the base price which is calculated by placing this user in previous time slots before t. Therefore, the time complexity of this step is $O((m_2 + n_2)(\Gamma + \chi_2 \log(\chi_2) + \chi_2))$, where χ_2 is the largest number of active users in the time slots before t. Therefore, the total time complexity is $O(\chi(\Gamma + \chi \log(\chi)))$, where $\chi = \max\{m_2 + n_2, \chi_2\}$.

$\square$



Comment: Vehicular Network
Traffic regulations: [✓ Right-side Traffic] Left-side Traffic
Period: 3600 Simulation seconds
Start Time: 00:00:00 [hh:mm:ss]
Start Date: [YYYYMMDD]
Simulation resolution: 5 Time step(s) / Sim. sec.
Controller Frequency: 1 Passes / Sim. sec.
Random Seed: 42
Simulation speed: [✓ 10.0 Sim. sec. / s] maximum
Break at: 0 Simulation seconds

Selected parameters: Acceleration, Desired Speed [m/s], Power, Start Time, Vehicle Number, Vehicle Type, Vehicle Type Name, World Coordinate Front X

Parameter selection: Acceleration, Delay Time, Desired Lane, Desired Speed [km/h], Desired Speed [m/s], Destination Lane, Destination Parking Lot, Dwell Time, Following Distance, Frontwegl[illegible]ke, Gradient, Interaction State

Beschleunigung [m/s[illegible] w[illegible]rend des Zeitschritts

Configuration file: vissim.fzk
[✓] Including parked vehicles
[✓] Database Table name: ka_3d_VEH_RECORD

Fig. 4.3 The Simulation of Vehicular Network Using VISSIM

Theorem 6. *The multi-attribute based online double auction mechanism satisfies individual rationality.*

Proof. The online double auction mechanism with non-price attributes satisfies individual rationality when buyer's bid is greater than the charge and each seller's ask is less than the reward. If a buyer b_i wins in the online auction, it will be charged by $p_i^b(t) = \max\{p_b^{candi}, bp_i^b\}$. According to Equation 4.5 and Algorithm 5, the bid h_i is greater than bp_i^b if it is admitted and the bid h_i is greater than p_b^{candi} since p_b^{candi} is the bid of the buyer after the last candidate winning buyer in the sorted buyers. Therefore, each buyer's bid h_i is greater than the charge $p_i^b(t)$. If a seller s_j wins in the online auction, it will be rewarded by $p_j^s(t) = \min\{p_s^{candi}, bp_j^s\}$. According to Equation 4.6 and Algorithm 5, the ask l_j is less than bp_j^s if it is admitted and the ask l_j is less than p_s^{candi} since p_s^{candi} is the ask of the seller after the last candidate winning seller in the sorted sellers. Therefore, each seller's ask l_j is less than the reward $p_j^s(t)$.

□

Theorem 7. *The multi-attribute based online auction mechanism is budget-balance.*

Proof. Our auction mechanism meets the property of budget-balance when payments from winning buyers are greater than rewards to the winning sellers. Namely, $\sum_{t=1}^{\infty}\sum_{b_i \in \mathscr{B}_w^t} p_i^b(t) - \sum_{t=1}^{\infty}\sum_{s_j \in \mathscr{S}_w^t} p_j^s(t) \geq 0$. At time slot t, $p_i^b(t) = \max\{p_b^{candi}, bp_i^b\} \geq p_b^{candi} \geq p_s^{candi} \geq p_j^s(t) = \min\{p_s^{candi}, bp_j^s\}$ for each winning pair (b_i, s_j). In other time slots, we could obtain the same results. Therefore, we could say that our scheme is budget-balance.

□

Theorem 8. *The proposed online double auction mechanism could meet the property of truthfulness for buyers and sellers, which means users could not improve their utility by providing fake information.*

Proof. We demonstrate the truthfulness of our work by showing that it meets common constraints for truthful online auction mechanism proposed in [18]. The proposed auction mechanism is based on a general online auction framework [9, 71], which applies the McAfee auction to active users. Different from the existing framework which does not consider the reasonable matching between buyers and sellers, our mechanism further extends the existing framework by designing a new winner selection rule and adding a matching rule based on vehicles' attributes. Since the existing framework has proved the truthfulness by the truthful constraints proposed in [18], we only need to prove that our modifications do not affect the truthfulness.

At the candidate winners selection stage, we design a new winner selection rule to replace the McAfee auction for covering more reasonable buyers/sellers which will be used in the matching stage. Our modification is to relax the aligned boundary used in the McAfee auction to look for a new boundary pair that covers more reasonable buyers/sellers, which also guarantees that all candidate winning buyers' bids are greater than candidate winning sellers' asks. Since we also define the price of the last buyer-seller pair which ensures the bid is greater than the ask as a candidate price and use the same pricing rule as the existing framework, we also could ensure the truthful constraints, as proved in [9, 71]. Therefore, our modification towards the candidate winners selection will not reduce the truthfulness.

Then we add a new stage, buyer-seller matching stage, to construct the reasonable one-to-one matching, which is only based on the non-price attributes. Therefore, the fake bid/ask and arrival/departure time will not affect this stage. For the non-price attributes which affect the matching stage, we could ensure their authenticity by the following schemes. Our matching rule is based on the following attributes: location, reputation, and computing power. Fake location information could be easily detected by many different countermeasures, such as crowdsourcing, statistical properties, and signal strength [7]. Reputation is a piece of public information scored by another part, and buyers/sellers cannot modify their own information. Therefore, they cannot modify the reputation. Computing ability is based on vehicles' type, which is provided when the vehicles register with the government agency. Therefore, this information is difficult to fake. Therefore, they could not submit fake non-price attributes used in the matching stage, and this stage will not affect the truthfulness. □

4.6 Evaluation

We evaluate the performance of the multi-attribute based online double auction mechanism and analyze the economic properties by the experiments.

4.6.1 Vehicular Network Simulation

To simulate the vehicular edge computing scenario, we use a classic vehicular network simulator, VISSIM. By the vehicular network simulator, we can load different roads and vehicles for constructing different scales of vehicular networks [42]. At the same time, the generated data of vehicles, including location, driving speed, vehicle type, and power, also can be obtained in real-time.

We use the commonly used city, Luxembourg, for vehicular network simulation. We choose an urban-intersection with the radius of $500m$ as the auction area, and store the generated driving data, as shown in Fig. 4.3. The driving data collection process will last for 60 minutes, and we divide the entire process into equal time slot $t = 3$ mins.

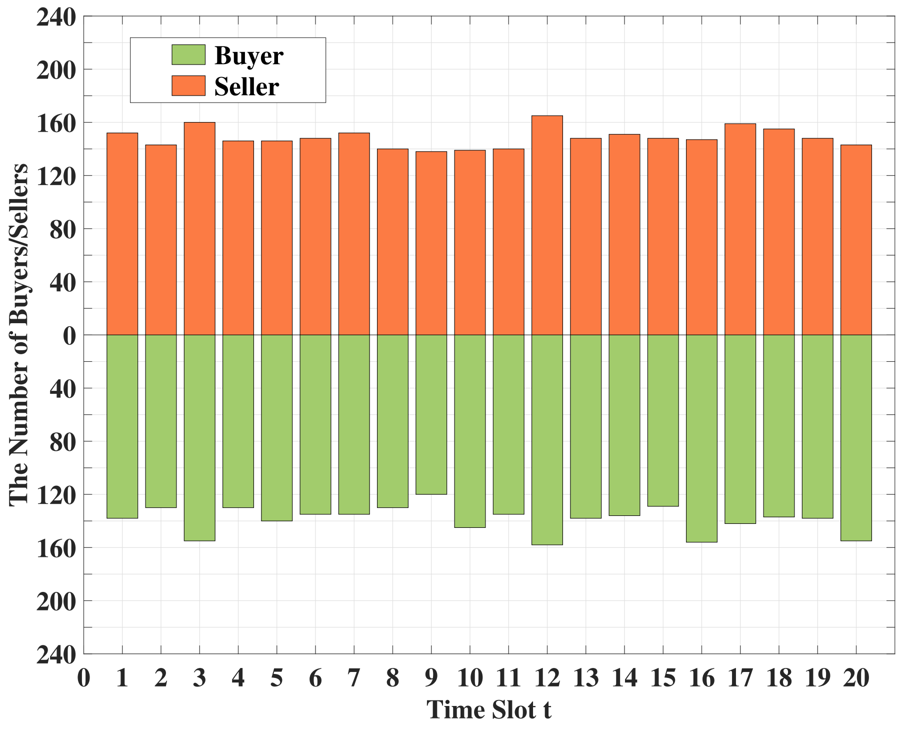


Fig. 4.4 The Number of Vehicles Arriving in Each Time Slot

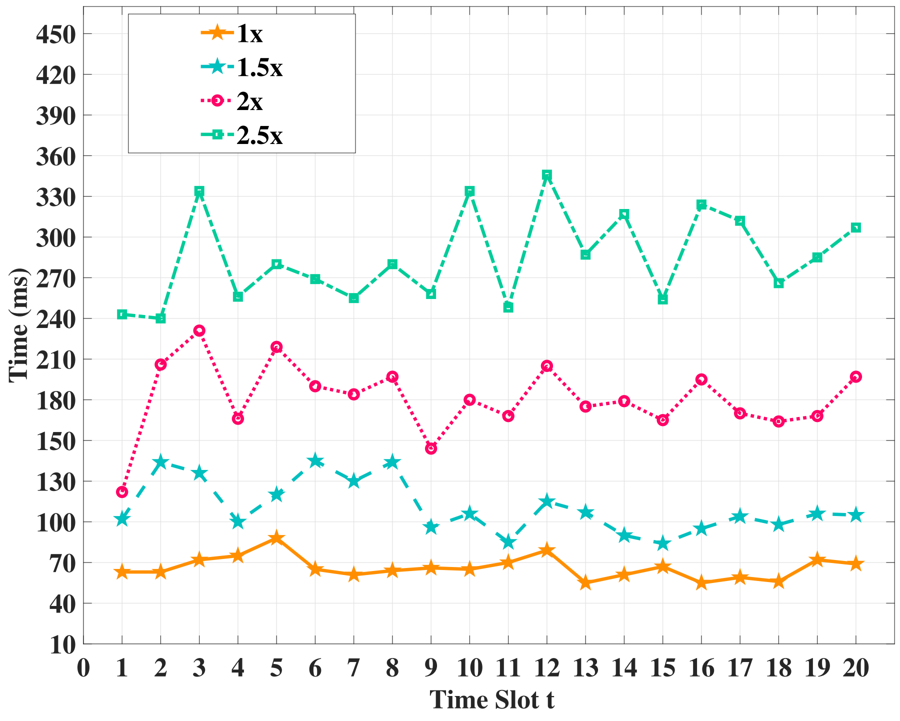


Fig. 4.5 Running Time with Different Number of Vehicles

4.6.2 Data Processing

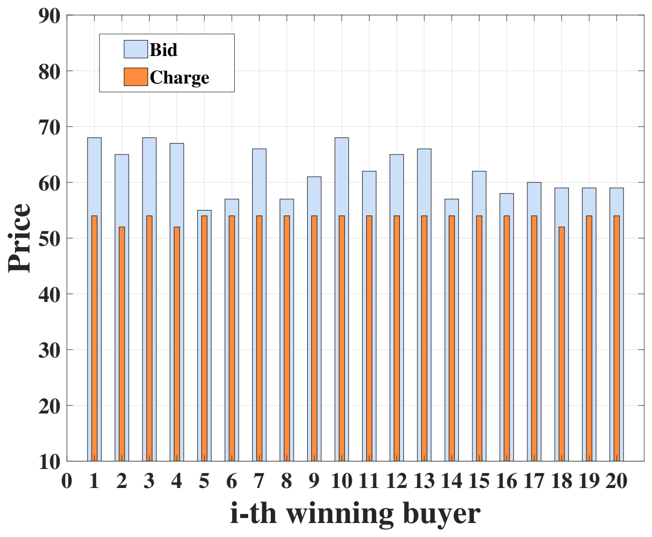
For the convenience of experiment, we split vehicles into two categories (vehicular edge nodes/client vehicles) based on speed and power, which means vehicles with the lower speed and higher vehicle power will be regarded as the vehicular edge nodes since vehicular edge nodes always refer to slow-moving and parked vehicles with the remaining resources in our model. Note that a vehicular edge node can also be a buyer when it has some tasks. In real-world implementations, the auctioneer does not need to execute this operation since a vehicle can claim to be a seller or a buyer according to its own needs. Then qualification review will be conducted on sellers to determine whether they can be sellers.

For each vehicular edge node, we consider three attributes, as discussed in Section 4.3. Location and computing power could be obtained from the collected driving data. Since reputation is the evaluation from other nodes, we could not obtain it from the vehicular network simulator. Therefore, we randomly generate the reputation of each vehicle in our experiment. Similarly, we also randomly generate three attribute requirements for each client vehicle. Fig. 4.4 shows the number of vehicular edge nodes and client vehicles arriving in each time slot.

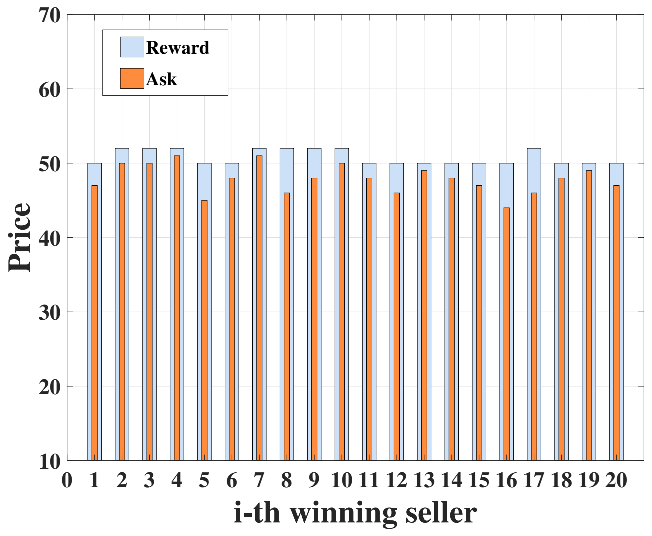
In our experiments, The bids of client vehicles and the asks of vehicular edge nodes are randomly generated in the range of $(30, 100]$ based on the following rules: client vehicles who want to obtain the better services (aka higher attribute requirements) will give a higher bid, and vehicular edge nodes who have more resources will assign a higher ask. We adopt the maximum matching algorithm (Hungary Algorithm) as an instantiation method of the one-to-one matching function $F(\cdot)$ unless otherwise stated, as introduced in Section 4.5.3. Performance metrics of the experiments are shown as follows.

4.6.3 Running Time at Each Time Slot

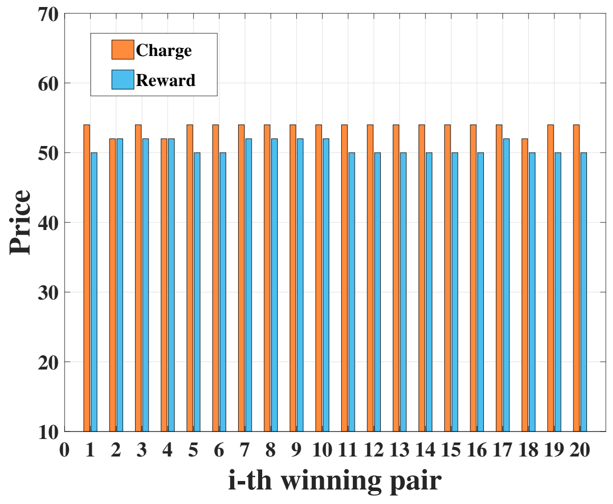
In addition to the theory analysis, we also use the experiments to prove that the proposed mechanism could satisfy the basic properties of online auction. The first one is the computational complexity. We deploy our scheme on a windows PC for testing its running time. Fig. 4.9 is the running time with different number of vehicles in each time slot. '2x" means the numbers of vehicles are twice as many as the original number of vehicles. We conduct many experiments to evaluate the performance of our scheme. From this figure, we can see that the online auction mechanism can be completed in milliseconds even if in the presence of a large number of users, which is far smaller than the length of time slot. Therefore, our scheme will not affect the performance of base stations.



(a) Bids VS Charges



(b) Asks VS Rewards



(c) Charges VS Rewards

Fig. 4.6 The Individual Rationality and Budget Balance

4.6.4 Final Price of Buyers, Sellers, and Auctioneer

The second property is the individual rationality, which means each buyer's bid is greater than the charge and each seller's ask is less than the reward, as defined in Section 4.4.2. To prove the individual rationality of the proposed online auction mechanism, we depict the bid (ask) and the final price (charge and reward) of each winning buyer (seller) at a time slot, as shown in Fig. 4.6a and Fig. 4.6b. Note that users in other time slots have the same results, and we do not show them due to space limitations.

As shown in Fig. 4.6a and Fig. 4.6b, the bid is greater than the charge, and the ask is less than the reward. Therefore, the proposed scheme meets the individual rationality. Fig. 4.6c shows the charge and the reward of each winning buyer-seller pair. Since our algorithm sets the same candidate charge/reward for the winning buyers/sellers, the charges (rewards) of some winning buyers (sellers) are the same in this figure. In this figure, the charge from winning buyers is always greater than the rewards to winning sellers. Therefore, the proposed scheme could meet budget-balance.

4.6.5 The Utility of Buyer/Seller under Different Bids/Asks

As discussed in Section 4.4.2, some selfish users will submit a fake bid/ask to improve their utilities. In this section, we will compute the utilities of some buyers/sellers under different bids/asks to prove that our scheme could meet the property of truthfulness. In order to cover all situations, we select the following four buyers and sellers at the same time slot: client vehicle $b_i \in \mathscr{B}_w(t)$, client vehicle $b_i^{'} \notin \mathscr{B}_w(t)$, vehicular edge node $s_j \in \mathscr{S}_w(t)$, and vehicular edge node $s_j^{'} \notin \mathscr{S}_w(t)$. Note that users in other time slots have the same results, and we do not show them due to space limitations. For these four users, we change their bids/asks and keep all parameters unchanged for computing the utilities under different bids/asks.

Fig. 4.7 is the results of these four buyers/sellers. In this figure, the abscissa represents the different bids/asks, and the ordinate represents the utilities under different bids/asks. The four lines represent the client vehicle b_i (true valuation: 57), client vehicle $b_i^{'}$ (51), vehicular edge node s_j (true cost: 48), and vehicular edge node $s_j^{'}$ (57) respectively. From this figure, we can see that users could obtain maximum utilities when they provide true information, no matter they are in the winning sets or not.

4.6.6 The Influence of Different Lengths of Time Slot

Then we evaluate the effects of different lengths of time slot. We change the length of time slot to evaluate the matching rate of buyers/sellers under different lengths of time slot, which

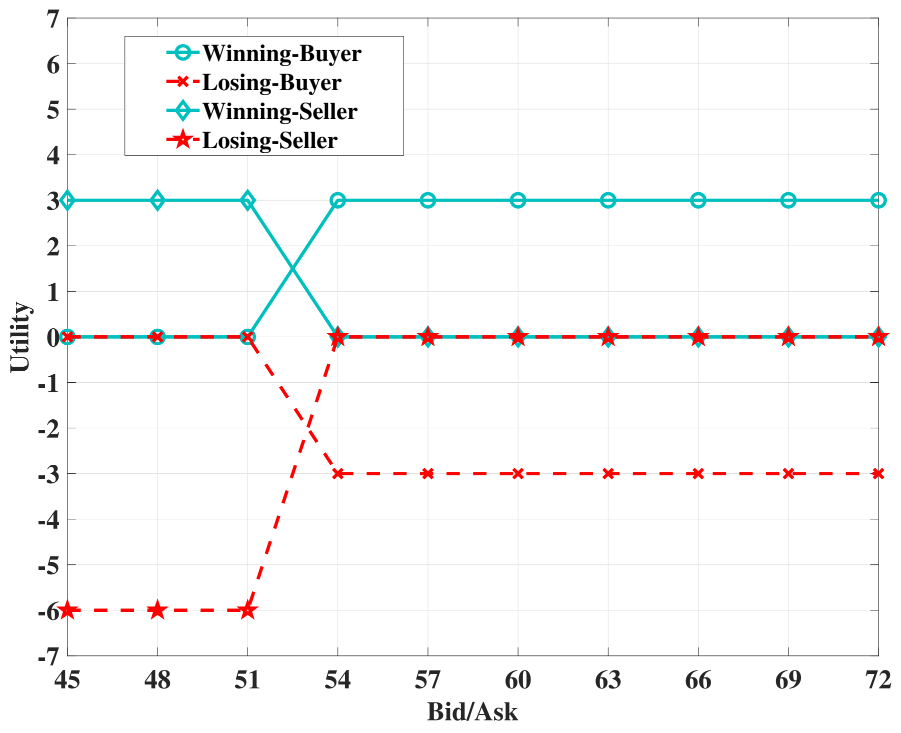


Fig. 4.7 The Utilities of Buyers and Sellers

is shown in Fig. 4.8. The matching rate is the number of matches divided by the number of buyers/sellers. From this figure, we can see that the length of time slot has little effect on matching rate. Even if a buyer/seller will be classified into different time slots when the length of time slot changes, it can still be matched by a seller/buyer if its bid/ask is reasonable. Therefore, the auction system can choose a reasonable length of time slot according to its needs.

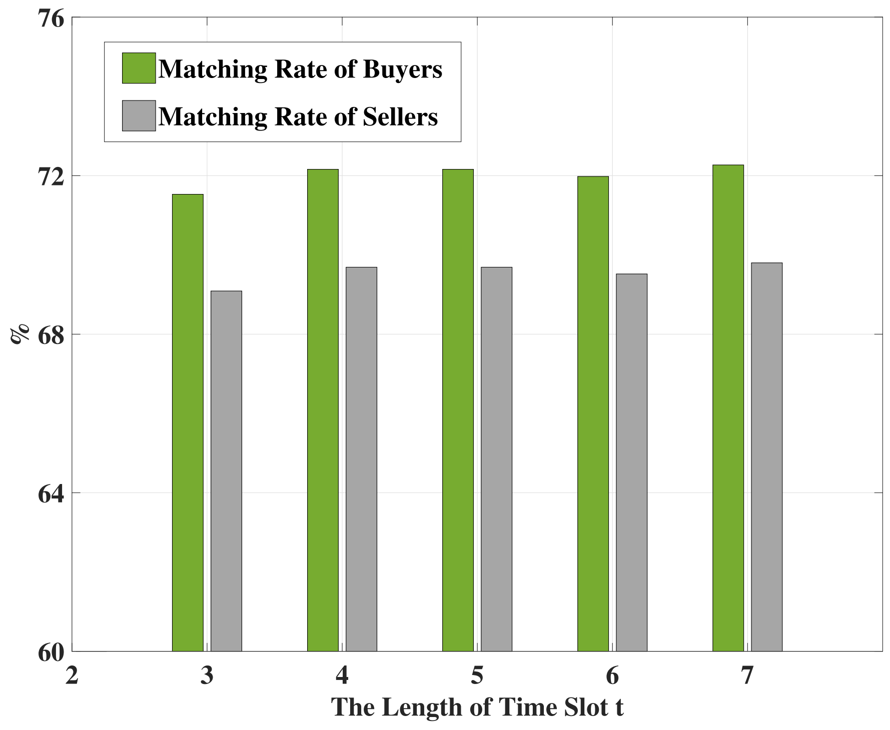


Fig. 4.8 Matching Rate under Different Lengths of Time Slot

4.6.7 The Influence of Different Matching Algorithms

As shown in Section 4.5.3, our mechanism provides a flexible one-to-one matching algorithm $F(\cdot)$ according to the different objectives. We evaluate the influence of different matching algorithms by the metric of running time. We apply different matching algorithms on the same buyers/sellers and keep other steps consistent to provide a fair comparison environment. For the maximum weighted matching algorithm, we assign a higher weight to an edge which links the client vehicle with higher requirements and the edge node with higher attribute values for increasing the resource utilization. Comparative result is shown in Fig. 4.9. It shows running times of all time slots under different matching algorithms. We can see that the maximum weighted matching algorithm always needs more time than the maximum matching algorithm since it needs to consider the weight of each edge when constructing the one-to-one matching. However, the difference is small and both of the running times are polynomial time.

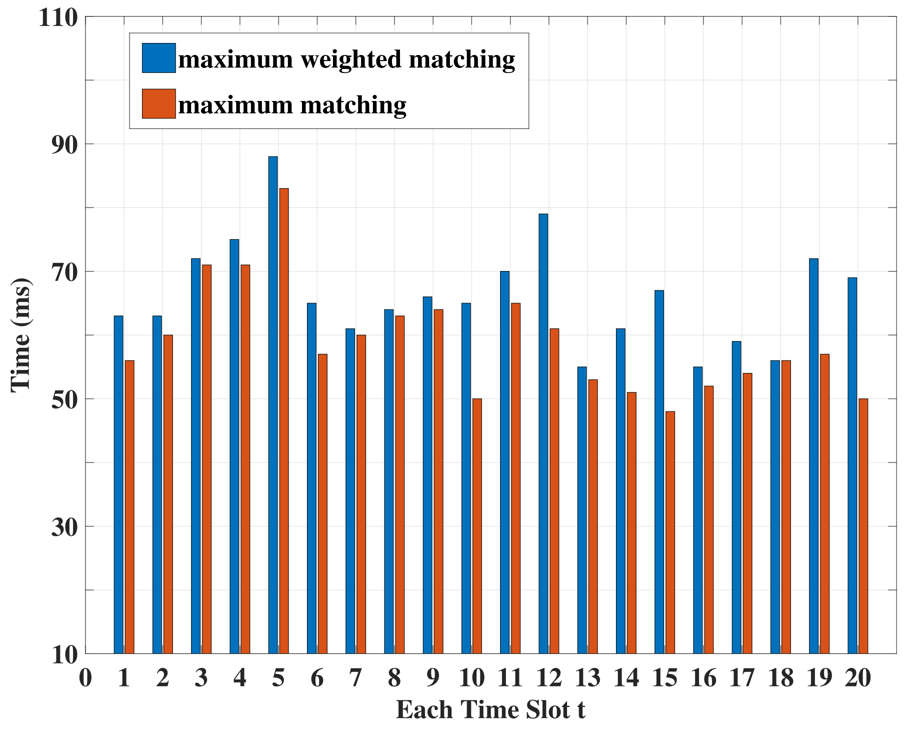


Fig. 4.9 Running Time under Different Matching Methods

4.7 Conclusion

In this chapter, we design an online auction scheme for VECN scenario, which could satisfy the dynamics of users in the auction system. Different from the static/offline auction which assumes that the auctioneer knows all information about sellers/buyers before the auction begins, we consider the situation that vehicular edge nodes and vehicles can join/leave the auction system at any time according to their wishes. Moreover, in addition to the bids/asks, we also consider the non-price attributes when constructing the matching between buyers and sellers. We simulate VECN using a vehicular network simulator for verifying our work. We evaluate our work from the perspective of running time, charges/rewards, and utilities of users. Experimental results show that the online auction mechanism could meet the proposed economic properties. In the future work, we will research auction mechanisms in other scenarios, such as Real-Time Bidding of Ad network.

Chapter 5

Conclusions and Future Work

This chapter summarizes the main research results of this dissertation, and prospects for future research. The author wishes to provide some inspiration for the relevant researchers of vehicular network. This dissertation is divided into four parts.

- The first part is composed of Chapter 1 : It briefly introduces the background and current status of the research about IoV in this dissertation. This part summarizes the background of VECN and the application scenarios we mainly focus on. Through the analysis of typical reference, the author reveals the disadvantage of the existing research and then explains the motivation of this dissertation, analyzes the current research challenges, research content, and significance.

- The second part is Chapter 2 of this dissertation: A broad learning based lightweight traffic analysis system was proposed. Firstly, edge computing can provide a distributed architecture, which can save the precious bandwidth resources and provide the safer service environment by offloading the analysis task to the network edge. Secondly, we use broad learning system to incrementally train the traffic data, which is more suitable for the edge computing because it has fast training speed and can support incremental learning. Then we implement the edge computing based traffic analysis system on the Raspberry Pi, which shows our model has the faster training speed compared with other neural network architecture.

- The third part is Chapter 3 of this dissertation: We design a multi-attribute based static/offline double auction mechanism in VECN scenario. The proposed auction mechanism not only considers the price but also considers non-price attributes when determining the winners. In addition, our auction mechanism could satisfy the following economic properties: computational efficiency, individual rationality, budget balance, and truthfulness. To verify our auction mechanism, we simulate the VECN scenario using VISSIM (a framework for running vehicle network simulation), and extract the driving data (location, speed, and vehicle type/power) of vehicles for the auction. Experimental results show the effectiveness and efficiency of our auction mechanism.

- The last part is Chapter 4 of this dissertation: We design a multi-attribute based online auction mechanism in VECN scenario, which could satisfy the dynamics of users in the auction system. Different from the static/offline auction which assumes that the auctioneer knows all information about sellers/buyers before the auction begins, we consider the situation that vehicular edge nodes and vehicles can join/leave the auction system at any time according to their wishes. Moreover, in addition to the bids/asks, we also consider the non-price attributes when constructing the matching between buyers and sellers. We simulate VECN using a vehicular network simulator for verifying our work. We evaluate our work from the perspective of running time, charges/rewards, and utilities of users. Experimental results show that the online auction mechanism could meet the proposed economic properties. In the future work, we will research auction mechanisms in other scenarios, such as real-time bidding of Ad network.

References

[1] (2019). Qualcomm technologies, inc. https://www.qualcomm.com/news/releases.

[2] (2019). A quarter billion connected vehicles will enable new in-vehicle services and automated driving capabilities. https://www.gartner.com/en/newsroom/press-releases/.

[3] Anderson, B. and McGrew, D. (2017). Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1723–1732. ACM.

[4] Bahreini, T., Badri, H., and Grosu, D. (2018). An envy-free auction mechanism for resource allocation in edge computing systems. In *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 313–322.

[5] Balsubramani, A., Dasgupta, S., and Freund, Y. (2013). The fast convergence of incremental pca. In *Advances in Neural Information Processing Systems*, pages 3174–3182.

[6] Bar-Magen, J. (2013). Fog computing: introduction to a new cloud evolution. In *Escrituras silenciadas: paisaje como historiografía*, pages 111–126. Servicio de Publicaciones.

[7] Bisio, I., Garibotto, C., Lavagetto, F., Sciarrone, A., and Zappatore, S. (2019). Blind detection: Advanced techniques for wifi-based drone surveillance. *IEEE Transactions on Vehicular Technology*, 68(1):938–946.

[8] Borjigin, W., Ota, K., and Dong, M. (2018). In broker we trust: A double-auction approach for resource allocation in nfv markets. *IEEE Transactions on Network and Service Management*, 15(4):1322–1333.

[9] Bredin, J., Parkes, D. C., and Duong, Q. (2007). Chain: A dynamic double auction framework for matching patient agents. *Journal of Artificial Intelligence Research*, 30:133–179.

[10] Cascarano, N., Ciminiera, L., and Risso, F. (2011). Optimizing deep packet inspection for high-speed traffic analysis. *Journal of Network and Systems Management*, 19(1):7–31.

[11] Chen, C. and Liu, Z. (2018). Broad learning system: An effective and efficient incremental learning system without the need for deep architecture. *IEEE Transactions on Neural Networks Learning Systems*, 29(1):10–24.

[12] Dagher, I. (2010). Incremental pca-lda algorithm. In *2010 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, pages 97–101.

[13] Dai, Y., Xu, D., Maharjan, S., and Zhang, Y. (2019). Joint load balancing and offloading in vehicular edge computing and networks. *IEEE Internet of Things Journal*, 6(3):4377–4387.

[14] Feghhi, S. and Leith, D. J. (2016). A web traffic analysis attack using only timing information. *IEEE Transactions on Information Forensics and Security*, 11(8):1747–1759.

[15] Feng, J., Liu, Z., Wu, C., and Ji, Y. (2017). Ave: Autonomous vehicular edge computing framework with aco-based scheduling. *IEEE Transactions on Vehicular Technology*, 66(12):10660–10675.

[16] Gepperth, A. and Hammer, B. (2016). Incremental learning algorithms and applications. In *European symposium on artificial neural networks (esann)*.

[17] Group, O. C. A. W. et al. (2016). Openfog architecture overview. *White Paper OPFWP001*, 216:35.

[18] Hajiaghayi, M. T. (2005). Online auctions with re-usable goods. In *Proceedings of the 6th ACM Conference on Electronic Commerce*, EC ’05, pages 165–174, New York, NY, USA. ACM.

[19] Han, S., Xu, S., Meng, W., and Li, C. (2018). Dense-device-enabled cooperative networks for efficient and secure transmission. *IEEE Network*, 32(2):100–106.

[20] Hui, Y., Su, Z., Luan, T. H., and Cai, J. (2019). Content in motion: An edge computing based relay scheme for content dissemination in urban vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, 20(8):3115–3128.

[21] Jalali, F., Hinton, K., Ayre, R., Alpcan, T., and Tucker, R. S. (2016). Fog computing may help to save energy in cloud computing. *IEEE Journal on Selected Areas in Communications*, 34(5):1728–1739.

[22] Jansen, K., Schäfer, M., Moser, D., Lenders, V., Pöpper, C., and Schmitt, J. (2018). Crowd-gps-sec: Leveraging crowdsourcing to detect and localize gps spoofing attacks. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 1018–1031. IEEE.

[23] Jiao, Y., Wang, P., Niyato, D., and Suankaewmanee, K. (2019). Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks. *IEEE Transactions on Parallel and Distributed Systems*, 30(9):1975–1989.

[24] Jiao, Y., Wang, P., Niyato, D., and Xiong, Z. (2018). Social welfare maximization auction in edge computing resource allocation for mobile blockchain. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–6.

[25] Jin, A., Song, W., Wang, P., Niyato, D., and Ju, P. (2016). Auction mechanisms toward efficient resource sharing for cloudlets in mobile cloud computing. *IEEE Transactions on Services Computing*, 9(6):895–909.

[26] Jin, A., Song, W., and Zhuang, W. (2018). Auction-based resource allocation for sharing cloudlets in mobile cloud computing. *IEEE Transactions on Emerging Topics in Computing*, 6(1):45–57.

[27] Joo, C. and Eryilmaz, A. (2018). Wireless scheduling for information freshness and synchrony: Drift-based design and heavy-traffic analysis. *IEEE/ACM Trans. Netw.*, 26(6):2556–2568.

[28] Kiani, A. and Ansari, N. (2017). Toward hierarchical mobile edge computing: An auction-based profit maximization approach. *IEEE Internet of Things Journal*, 4(6):2082–2091.

[29] Kumar, K., Prakash, A., and Tripathi, R. (2017). A spectrum handoff scheme for optimal network selection in cognitive radio vehicular networks: A game theoretic auction theory approach. *Physical Communication*, 24:19 – 33.

[30] Li, H., Ota, K., and Dong, M. (2018). Learning iot in edge: Deep learning for the internet of things with edge computing. *IEEE Network*, 32(1):96–101.

[31] Li, H., Ota, K., and Dong, M. (2019). Deep reinforcement scheduling for mobile crowdsensing in fog computing. *ACM Trans. Internet Technol.*, 19(2):21:1–21:18.

[32] Li, L., Li, Y., and Hou, R. (2017). A novel mobile edge computing-based architecture for future cellular vehicular networks. In *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6.

[33] Li, X., Qi, F., Xu, D., and Qiu, X.-s. (2011). An internet traffic classification method based on semi-supervised support vector machine. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–5. IEEE.

[34] Liao, H., Zhou, Z., Mumtaz, S., and Rodriguez, J. (2019). Robust task offloading for iot fog computing under information asymmetry and information uncertainty. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–6.

[35] Liu, J., Wan, J., Zeng, B., Wang, Q., Song, H., and Qiu, M. (2017). A scalable and quick-response software defined vehicular network assisted by mobile edge computing. *IEEE Communications Magazine*, 55(7):94–100.

[36] Liwang, M., Dai, S., Gao, Z., Tang, Y., and Dai, H. (2019). A truthful reverse-auction mechanism for computation offloading in cloud-enabled vehicular network. *IEEE Internet of Things Journal*, 6(3):4214–4227.

[37] Mao, Y., You, C., Zhang, J., Huang, K., and Letaief, K. B. (2017). A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys Tutorials*, 19(4):2322–2358.

[38] Naboulsi, D., Fiore, M., Ribot, S., and Stanica, R. (2016). Large-scale mobile traffic analysis: a survey. *IEEE Communications Surveys & Tutorials*, 18(1):124–161.

[39] Ni, J., Zhang, A., Lin, X., and Shen, X. S. (2017). Security, privacy, and fairness in fog-based vehicular crowdsensing. *IEEE Communications Magazine*, 55(6):146–152.

[40] Ning, Z., Huang, J., and Wang, X. (2019). Vehicular fog computing: Enabling real-time traffic management for smart cities. *IEEE Wireless Communications*, 26(1):87–93.

[41] Patel, P., Intizar Ali, M., and Sheth, A. (2017). On using the intelligent edge for iot analytics. *IEEE Intelligent Systems*, 32(5):64–69.

[42] Peng, H. and Shen, X. S. (2020). Deep reinforcement learning based resource management for multi-access edge computing in vehicular networks. *IEEE Transactions on Network Science and Engineering*, pages 1–1.

[43] Peng, M., Xie, X., Hu, Q., Zhang, J., and Poor, H. V. (2015). Contract-based interference coordination in heterogeneous cloud radio access networks. *IEEE Journal on Selected Areas in Communications*, 33(6):1140–1153.

[44] Qin, J., Zhu, H., Zhu, Y., Lu, L., Xue, G., and Li, M. (2016). Post: Exploiting dynamic sociality for mobile advertising in vehicular networks. *IEEE Transactions on Parallel and Distributed Systems*, 27(6):1770–1782.

[45] Rodrigues, T. G., Suto, K., Nishiyama, H., and Kato, N. (2017). Hybrid method for minimizing service delay in edge cloud computing through vm migration and transmission power control. *IEEE Transactions on Computers*, 66(5):810–819.

[46] Rodrigues, T. G., Suto, K., Nishiyama, H., Kato, N., and Temma, K. (2018). Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration. *IEEE Transactions on Computers*, 67(9):1287–1300.

[47] Rodrigues, T. K., Suto, K., Nishiyama, H., Liu, J., and Kato, N. (2019). Machine learning meets computation and communication control in evolving edge and cloud: Challenges and future perspective. *IEEE Communications Surveys Tutorials*, pages 1–1.

[48] Rost, P., Bernardos, C. J., Domenico, A. D., Girolamo, M. D., Lalam, M., Maeder, A., Sabella, D., and Wübben, D. (2014). Cloud technologies for flexible 5g radio access networks. *IEEE Communications Magazine*, 52(5):68–76.

[49] RP-181429, G. (2018). New sid: Study on nr-v2x. In *3GPP TSG RAN Meeting*.

[50] Sankari, S., Varalakshmi, P., and Divya, B. (2015). Network traffic analysis of cloud data centre. In *International Conference on Computing and Communications Technologies*, pages 408–413.

[51] Satyanarayanan, M., Lewis, G., Morris, E., Simanta, S., Boleng, J., and Ha, K. (2013). The role of cloudlets in hostile environments. *IEEE Pervasive Computing*, 12(4):40–49.

[52] Shaukat, U., Ahmed, E., Anwar, Z., and Xia, F. (2016). Cloudlet deployment in local wireless area networks, motivation, taxonomies, and open research challenges. *J. Netw. Comput. Appl*, 62(10.1016).

[53] Shi, W., Cao, J., Zhang, Q., Li, Y., and Xu, L. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646.

[54] Shi, W. and Dustdar, S. (2016). The promise of edge computing. *Computer*, 49(5):78–81.

[55] Shojafar, M., Cordeschi, N., and Baccarelli, E. (2019). Energy-efficient adaptive resource management for real-time vehicular cloud services. *IEEE Transactions on Cloud Computing*, 7(1):196–209.

[56] Sorkhoh, I., Ebrahimi, D., Atallah, R., and Assi, C. (2019). Workload scheduling in vehicular networks with edge cloud capabilities. *IEEE Transactions on Vehicular Technology*, 68(9):8472–8486.

[57] Sun, W., Liu, J., Yue, Y., and Zhang, H. (2018). Double auction-based resource allocation for mobile edge computing in industrial internet of things. *IEEE Transactions on Industrial Informatics*, 14(10):4692–4701.

[58] Taleb, T., Dutta, S., Ksentini, A., Iqbal, M., and Flinck, H. (2017). Mobile edge computing potential in making cities smarter. *IEEE Communications Magazine*, 55(3):38–43.

[59] Tang, F., Fadlullah, Z. M., Kato, N., Ono, F., and Miura, R. (2018). Ac-poca: Anticoordination game based partially overlapping channels assignment in combined uav and d2d-based networks. *IEEE Transactions on Vehicular Technology*, 67(2):1672–1683.

[60] Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., and Ghogho, M. (2016). Deep learning approach for network intrusion detection in software defined networking. In *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, pages 258–263.

[61] Tao, M., Wei, W., and Huang, S. (2019). Location-based trustworthy services recommendation in cooperative-communication-enabled internet of vehicles. *Journal of Network and Computer Applications*, 126:1 – 11.

[62] Taylor, V. F., Spolaor, R., Conti, M., and Martinovic, I. (2018). Robust smartphone app identification via encrypted network traffic analysis. *IEEE Transactions on Information Forensics and Security*, 13(1):63–78.

[63] Wang, Q., Guo, S., Wang, Y., and Yang, Y. (2019). Incentive mechanism for edge cloud profit maximization in mobile edge computing. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–6.

[64] Wei, Y., Zhu, Y., Zhu, H., Zhang, Q., and Xue, G. (2015). Truthful online double auctions for dynamic mobile crowdsourcing. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 2074–2082.

[65] Xiao, Y. and Chao Zhu (2017). Vehicular fog computing: Vision and challenges. In *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 6–9.

[66] Yang, D., Fang, X., and Xue, G. (2011). Truthful auction for cooperative communications. In *Proceedings of the Twelfth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, page 9. ACM.

[67] Yi, S., Li, C., and Li, Q. (2015). A survey of fog computing: concepts, applications and issues. In *Proceedings of the 2015 workshop on mobile big data*, pages 37–42.

[68] Yuan, Q., Zhou, H., Li, J., Liu, Z., Yang, F., and Shen, X. S. (2018). Toward efficient content delivery for automated driving services: An edge computing solution. *IEEE Network*, 32(1):80–86.

[69] Zeng, M., Leng, S., Maharjan, S., Gjessing, S., and He, J. (2015). An incentivized auction-based group-selling approach for demand response management in v2g systems. *IEEE Transactions on Industrial Informatics*, 11(6):1554–1563.

[70] Zhang, D., Tan, L., Ren, J., Awad, M. K., Zhang, S., Zhang, Y., and Wan, P. (2019). Near-optimal and truthful online auction for computation offloading in green edge-computing systems. *IEEE Transactions on Mobile Computing*, pages 1–1.

[71] Zhang, H., Liu, B., Susanto, H., Xue, G., and Sun, T. (2016). Incentive mechanism for proximity-based mobile crowd service systems. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9.

[72] Zhang, K., Mao, Y., Leng, S., He, Y., and ZHANG, Y. (2017). Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading. *IEEE Vehicular Technology Magazine*, 12(2):36–44.

[73] Zhao, L., Wang, J., Liu, J., and Kato, N. (2019). Optimal edge resource allocation in iot-based smart cities. *IEEE Network*, 33(2):30–35.

[74] Zhong, W., Xie, K., Liu, Y., Yang, C., and Xie, S. (2017). Efficient auction mechanisms for two-layer vehicle-to-grid energy trading in smart grid. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6.

[75] Zhou, L., Du, S., Zhu, H., Chen, C., Ota, K., and Dong, M. (2019). Location privacy in usage-based automotive insurance: Attacks and countermeasures. *IEEE Transactions on Information Forensics and Security*, 14(1):196–211.

[76] Zhu, C., Pastor, G., Xiao, Y., and Ylajaaski, A. (2018). Vehicular fog computing for video crowdsourcing: Applications, feasibility, and challenges. *IEEE Communications Magazine*, 56(10):58–63.

[77] Zhu, H., Li, M., Fu, L., Xue, G., Zhu, Y., and Ni, L. M. (2011). Impact of traffic influxes: Revealing exponential intercontact time in urban vanets. *IEEE Transactions on Parallel and Distributed Systems*, 22(8):1258–1266.

[78] Zhu, H., Li, M., Zhu, Y., and Ni, L. M. (2009). Hero: Online real-time vehicle tracking. *IEEE Transactions on Parallel and Distributed Systems*, 20(5):740–752.

[79] Zhu, H., Zhou, M., and Alkins, R. (2012). Group role assignment via a kuhn munkres algorithm-based solution. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 42(3):739–750.

Publications

Journals

1. Xiting Peng, Kaoru Ota, Mianxiong Dong, "Multi-attribute based Double Auction Towards Resource Allocation in Vehicular Fog Computing," IEEE Internet of Things Journal (IOTJ),vol. 7, no. 4, pp. 3094-3103, April 2020.

2. Xiting Peng, Kaoru Ota, Mianxiong Dong, "A Broad Learning-Driven Network Traffic Analysis System Based on Fog Computing Paradigm," China Communications, Vol. 17, Issue. 2, pp. 1-13, March 2020.

Proceeding of International Conference

1. Xiting Peng, Kaoru Ota, Mianxiong Dong, "Edge Computing based Traffic Analysis System Using Broad Learning,"(invited paper) EAI International Conference on Artificial Intelligence for Communications and Networks (AICON 2019), Harbin, China, May 25-26, 2019. (Best Paper Award)

Under Review

1. Xiting Peng, Kaoru Ota, Mianxiong Dong, "Online Resource Auction for Edge-assistant Vehicular Network with Non-price Attributes," IEEE Transactions on Vehicular Technology (TVT), Minor Revision.