Performance Modelling and Quantitative Analysis of Vehicular Edge Computing with Bursty Task Arrivals

Wang Miao*, Geyong Min*, Xu Zhang†, Zhiwei Zhao‡, Jia Hu*,
* College of Engineering, Mathematics and Physical Science, University of Exeter, UK
† School of Electronic Science and Engineering, Nanjing University, China
‡ School of Computer Science and Engineering, University of Electronic Science and Technology of China

Abstract—The quantitative performance analysis plays a critical role in assessing the capability of Vehicular Edge Computing (VEC) systems to meet the requirements of vehicular applications. However, developing accurate analytical models for VEC systems is extremely challenging due to the unique features of intelligent vehicular applications. Specifically, recent work revealed that the tasks generated by intelligent vehicular applications exhibit a high degree of burstiness, rendering the existing models that were designed based on the assumption of the non-bursty Poisson process unsuitable for VEC systems. To fill this gap, we developed an original analytical model to investigate the performance of VEC systems with bursty task arrivals. To facilitate vehicle cooperation, a new priority-based resource allocation scheme is exploited to schedule the tasks of vehicular applications, which are modelled by a Markov Modulated Poisson Process (MMPP). Next, a multi-state Markov chain is established to investigate the impact of load sharing strategy on the performance of VEC systems. Then, the end-to-end transmission latency is derived based on the proposed model. Comprehensive experiments are conducted to validate the accuracy of this analytical model under various system configurations. Furthermore, the developed model is used as a cost-effective tool to investigate the performance bottleneck of VEC systems.

Index Terms—Mobile Edge Computing, Vehicular Application, Analytical Modelling, Bursty Arrivals, Performance Analysis

1 Introduction

With the rapid development of the sensor technologies and advancements of autonomous driving, Internet-of-Vehicle (IoV) has been considered as a promising platform to improve road safety through exploiting the sensors (cameras, radar, sonar, and inertial measurement units) and cooperations of multiple vehicles to eliminate the potential mistakes that human drivers would routinely make [1]. These sensors provide rich data for learning-based algorithms to be trained, adjusted and optimised, obtaining better performance than human drivers in various vehicular applications, e.g., automated road hazard detection, accurate planning of complex driving manoeuvres, and the cooperation among vehicles. While, to obtain meaningful knowledge from a huge amount of raw data, sensor-rich data analysis requires substantial computing power and storage space [2]. Compared with the traditional Internet-of-Things (IoT) services, IoV applications are characterised by high reliability and agile response capability. As a result, it is intractable to exploit the remote cloud resource for conducting data analysis due to the long transmission latency and unstable network conditions. A more applicable approach is to migrate the computation-intensive tasks of IoV applications to network edge, generating a new paradigm named vehicular edge intelligence, which integrates the Vehicular Edge Computing (VEC) architecture and Artificial Intelligence (AI) capabilities. On one hand, by pushing cloud computing capabilities to network edge and vehicle terminals, VEC is emerging as a new compelling computing paradigm to underpin a variety of computation-intensive and delay-sensitive vehicular applications with low service latency and high computation capabilities [3] [4] [5] [6]. On the other hand, by deploying advanced AI algorithms in network edge and vehicle terminals, IoV applications obtain a number of benefits, including but not limited to the higher analysis capability, wider environment awareness, and more intelligent service provisioning. Through complementarily cross-fertilising these two technologies, vehicular edge intelligence is regarded as an indispensable technology in the evolution of lifting the performance of IoV systems to the next level.

Vehicular edge intelligence has attracted tremendous research interests in recent years [6] [7] [8]. However, to fully unleash its power for IoV applications, the underlying communication and computation infrastructure should satisfy the performance requirements of the upper-level AI algorithms. Compared with the centralised AI algorithms in the cloud environment, the performance of edge AI algorithms is highly restricted by the capability of the underlying infrastructure, which is characterised by distribution, heterogeneity, dynamicity and limited resources. For instance, federated learning is one of the most promising machine learning algorithms in IoV applications [9]. The simultaneous parameter uploading during model aggregation creates new challenges for underlying transmission systems [10], e.g., the strict transmission latency requirements and serious spectrum collision and congestion. In this regard, any abnormal transmission delay or packet loss in parameter transmission or processing would slow down

the algorithm convergence speed and result in unsatisfied accuracy in multi-round model updating. This may lead to serious consequences in highly dynamic IoV environments, such as road accidents and casualties, thus warranting a comprehensive performance investigation of the underlying edge computing system for intelligent IoV applications.

Performance analytical models have been widely used as a necessary stepping stone in system design and performance optimisation [11] [12]. It can capture the essential features of the system operation, gain significant insights, and offer a cost-effective and versatile tool to theoretically identify the performance bottleneck with different design alternatives and under various working conditions. Recently, several studies on the model analysis of intelligent edge computing have been recently reported in the current literature [13] [14] [15] [16]. Specifically, the works in [13] [14] investigated the service provisioning capabilities of edge computing systems for mobile devices with respect to service response time and outage probability. Furthermore, because the load sharing technology is of paramount importance for the edge computing system to handle a large number of task arrivals, the authors in [15] [16] proposed novel analytical models to investigate the performance improvement of load sharing for mobile edge computing systems. Although some progress has been made in the model analysis of edge computing systems, the existing analytical models can be hardly used to investigate the performance of intelligent edge computing with IoV applications. This is because, firstly, compared with the traditional IoT devices, the emerging vehicles are equipped with computers to process the data collected from various sensors and cameras. For example, multiple vehicles would exchange the collected data to expand the view of the road environment for safe driving [17]. Thus, the analytical model is required to be able to analyse the cooperation of multiple vehicles. Secondly, most of the existing analytical models were developed subject to the assumption that the tasks follow the non-bursty Poisson process, which is mainly used to model the text data on the Internet. While some recent work revealed that the tasks generated by AI-driven VEC systems exhibit a high degree of burstiness [18] [19] [20] [21] [22] [23] [24]. For instance, the local clients in federated learning submit their payloads during short periods in a synchronised manner [18], which forms accumulative traffic in a split second and brings high burstiness for resource consumption, calling for new analytical models to investigate its impact on the latency-sensitive IoV applications.

To fill this gap, we develop a new analytical model in this paper to quantitatively investigate the end-to-end performance of the intelligent edge computing system with IoV applications. The proposed analytical model is designed with the aim of capturing the unique features of VEC systems, including multiple vehicle cooperation, Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) communications, traffic burstiness, and load sharing strategy. To analyse the capability of the edge computing system hosting the reliable and agile IoV applications, the average end-to-end latency is derived based on the proposed analytical model. The main contributions are summarised as follows:

- To study the bursty feature of the traffic and tasks generated by intelligent vehicular applications, the Markov Modulated Poisson Process (MMPP) is firstly exploited for analysing the performance of mobile VEC and investigating its impacts on the end-to-end service provisioning.
- To capture the cooperation among multiple vehicles, a Priority Queue (PQ) based traffic scheduling method is developed to analyse the impact of vehicle cooperation on the latency performance. In addition, a multi-state Markov chain is established to investigate the performance of load sharing among edge servers.
- To validate the accuracy of the proposed analytical model, extensive simulation experiments are conducted, showing that the performance results achieved by the proposed analytical model match well with these obtained from the simulation experiments under various system configurations.
- To demonstrate its value for the mobile VEC system design, the proposed analytical model is used as a cost-effective tool to investigage the performance bottlenecks and optimisation strategies of task offloading and resource allocation in VEC systems.

The rest of this work is organised as follows, Section 2 presents the state-of-the-art of analytical model in edge intelligence for IoV applications. Section 3 abstracts an analytical model to capture the unique features of intelligent edge computing systems. Section 4 derives an analytical model to investigate the end-to-end system performance. Section 5 validates the accuracy of the proposed model and discusses its practical usage followed by Section 6, which concludes this study and provides the future research directions.

2 Related Work

Owning the benefits of IoV service provisioning, e.g., powerful computation, low latency and energy efficiency, VEC [3] [25] has been considered as a promising paradigm to satisfy the strict performance requirements of IoV applications. AI solutions play a pivotal role in the development of VEC systems. By enabling the VEC the capability of AI solution, also known as vehicle edge intelligence, IoV applications obtain a number of benefits, including but not limited to shortened system latency, reduced energy consumption, higher computation capability, and wider environmental awareness. Following this wisdom, a rich literature has been developed around edge intelligence for IoV applications [6] [7] [8]. For example, to facilitate the message delivery in vehicle ad-hoc networks, the authors in [6] proposed a new routing algorithm based on Reinforcement Learning (RL) architecture, the objective of which is to improve the message delivery performance in terms of the hop number and transmission delay. To enhance the resource utilisation efficiency in V2V communications, the authors in [7] built a distributed resource allocation mechanism based on Deep Reinforcement Learning (DRL), where the reward function, action space and state space were carefully designed to make the proposed solution applicable to both the unicast and multicast communication scenarios. To render the offloading decision capable of capturing the long-term system

status, the authors in [8] designed a DRL-based offloading strategy to jointly consider the resource requirement, network transmission quality, and vehicle mobility.

To release the potential of AI algorithms in VEC systems, the underlying computation and transmission infrastructure should provide sufficient performance to meet the strict Quality-of-Service (QoS) requirements of AI operations. In this regard, various analytical models have been proposed to investigate the service capability of the edge computing system with different system architectures. For instance, a stochastic analytical model was proposed in [13] to investigate the performance of offloading strategy for mobile applications. The results revealed that for delay-sensitive applications, a partial offloading strategy is preferable with stable application traffic. Following [13], the authors in [14] used the stochastic geometry to analyse service outage probability of heterogeneous mobile cloud computing systems, where the tasks are fully sent to radio access network for offloading decision making. In addition, to investigate the performance of cooperation strategies among edge servers, which could increase the service acceptance ratio for IoV applications, the authors in [15] proposed a Queueing theory based analytical model to investigate the performance of load sharing schemes in edge computing, with the aim of obtaining the metrics of packet blocking probability and average waiting time. Similarly, the authors in [16] developed a Markov multi-server queuing model to evaluate the performance of edge computing systems with limited computation capabilities. The minimum number of processors was derived based on the proposed model to satisfy certain QoS requirements.

Although some interesting research findings have been reported in the literature, due to the unique features of mobile VEC systems, the existing analytical models can be hardly used in the IoV scenario to evaluate the performance analysis of intelligent edge computing systems. For example, most of the analytical models reported are mainly developed to investigate the performance of edge computing with IoT applications, paying little attention to the unique features of IoV applications, e.g., vehicle cooperation and multiple device access. In addition, the existing models are developed based on unrealistic assumptions that the traffic follows the Poisson process, which is mainly used to model non-bursty text data. While a rich literature [19] [20] [21] demonstrates that the tasks and communication traffic generated by vehicular applications, e.g., safety, road traffic efficiency and infotainment, exhibit a high bursty feature. For example, the work in [19] discovered that high mobility of the vehicles results in more spatially and temporally coupled behaviour, making the real V2V information exchange to be bursty, and developed a coupled Markovian arrival process to capture the burstiness of vehicle traffic arrival. Similar to [19], the authors in [26] argued that due to the complicated surrounding environment, the Big Data collected from different vehicular application classes such as monitoring, protection and control, have the features in terms of burstiness, velocity, and freshness. Besides the bursty features of the vehicle traffic and applications reviewed above, the processing of AI algorithms is also characterised by high burstiness. For example, after investigating the real-world traces of the vehicle plate recognition, the work in [20] concluded that Machine Learning (ML) inference often has bursty task arrivals and analysis outputs, and highlighted that if burstiness is not carefully considered in the ML model design, the quality of the ML prediction may plummet. In addition, the authors in [21] proposed a machine learning based jamming detection in a vehicle network and discovered that the convergence speed of the ML-enabled jamming detection has a high degree of burstiness. Extensive simulation experiments were conducted to investigate the distribution of the burstiness length. How the burstiness of traffic and tasks affects the performance of the network transmission and server processing requires further research efforts.

To address these challenges, a new analytical model is proposed in this study to analyse the performance of mobile VEC systems under the burstiness traffic. In this study, we exploit the MMPP to model the burstiness of traffic arrivals in VEC systems. MMPP model is a stochastic Poisson process whose arrival rates vary according to a continuous time Markov chain. Due to its superior capability of capturing the burstiness for system traffics, MMPP has been used in a wide spectrum of fields, e.g., Satellite Data Relay Networks (SDRN) to model the on-off feature of SDRN traffic arrival [22], cellular network to model the burstiness of voice traffic [23], cluster-based computer system to model the non-stationary and bursty features of traffic arrival [24] and so on. As the traffic generated by the AI-enabled VEC systems exhibits a high degree of burstiness due to the spatial and temporal interaction feature and the complex vehicle environment, MMPP is exploited in this study to model the traffic arrivals of VEC systems. In addition, due to the features of dynamic traffic arrivals, unstable wireless transmission, and varying resource availability, most of the existing offloading strategies target to optimise the QoS metrics from the perspective of statistical average, such as the average offloading time [5] [8] [14], average virtual machine migration costs [27], average queueing time [15] [16], average power consumption [13], average throughput [4] [26], and so on. In line with the existing work on task offloading, this paper aims to develop a new analytical model capable of quantitatively studying the average performance metrics of VEC systems with bursty task arrivals.

Furthermore, before diving into the details of the model derivation, we give a brief description about the distinctions among several relevant concepts, including Edge Computing (EC), mobile EC, multi-access EC, and VEC. Specifically, EC is a general term for a cloud-based IT service environment located at the edge of a network. Mobile EC and multi-access EC are two typical architecture concepts of EC, defined by the European Telecommunications Standards Institute (ETSI). The main difference between mobile EC and multi-access EC lies in that the latter refers to providing cloud-computing capability at the edge of any networks, while the former solely focuses on the edge of the mobile networks. Meanwhile, distinguished from mobile EC and multi-access EC, VEC refers to pushing the computation resources from cloud to both the network edge and vehicle terminals to support mission-critical vehicular applications, where the features of vehicular scenario, e.g., high mobility, dynamic V2V/V2I channels, and multi-vehicle cooperation, are considered in the design of VEC systems to satisfy the

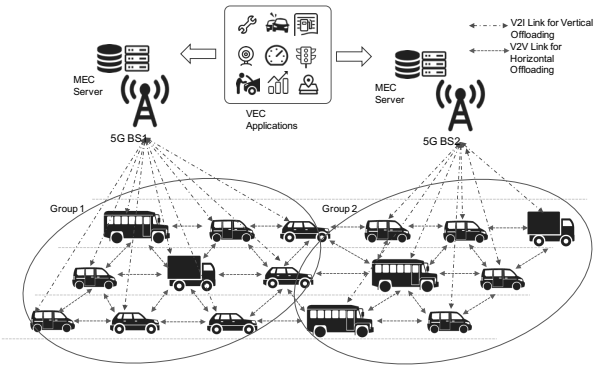

Fig. 1: VEC system architecture

strict QoS requirements of vehicular applications.

3 VEC Working Mechanism and Model Abstraction

This section firstly presents the working mechanism of mobile VEC with a focus on how edge computing is used to support and meet the strict performance requirements of IoV applications. Then we abstract an analytical model to capture the methodologies and unique features of the mobile VEC systems. Finally, a subsection is presented to describe the fundamental knowledge of the analytical model, which will be used to derive the end-to-end performance in Section 4.

3.1 Working mechanism of VEC

This subsection presents the working mechanism of the VEC ecosystem. As shown in Fig. 1, the vehicle in the VEC ecosystem is equipped with various communication and sensing technologies to realise various critical functions for autonomous driving, including localisation, environment perception, path planning, vehicle manoeuvres and so on. The vehicles are connected to VEC servers through cellular networks and further linked to the cloud through core and Internet infrastructure, referred to as V2I communication. In addition, the vehicles can communicate with other nearby vehicles through Dedicated Short-Range Communication (DSRC), referred to as V2V communications. In this regard, this paper exploits both the broadcast mode and unicast mode of DSRC to establish communications with the neighbour vehicles. This is because the IoV system is characterised by highly dynamic environment, e.g., network topology and channel condition, which requires the vehicle to frequently disseminate its status information, e.g., computation resource availability, to neighbour vehicles for assisting offloading decision making. In this case, in light of the high resource efficiency, the broadcast mode, e.g., Basic Safety Messages (BSM), is utilised to establish the status sharing among multiple vehicles. Meanwhile, when an offloading decision is made based on the cooperation and negotiation among multiple vehicles, a unicast approach, e.g., IP-based, will be used to send computation tasks to specific neighbour vehicles. By exploiting V2V and V2I, vehicles obtain a better understanding of the dynamic surrounding environment to make more accurate and safer manoeuvres compared with human counterparts. Furthermore, multiple VEC servers are clustered into a domain in the VEC ecosystem to serve the vehicles under the coverage of 5G Base Station (BS). Normally, one VEC domain is connected to one BS. While under the high mobility scenario, e.g., dual carriageway or motorway, one VEC domain could simultaneously connect to multiple 5G BSs to enlarge the VEC service coverage. This design could effectively handle the issues of service quality degradation caused by frequent service migrations among VEC clusters.

In the VEC architecture, a huge amount of data is generated by various onboard sensors and devices, which needs to be processed and analysed normally in real-time to guarantee the functionality and safety of IoV applications. To satisfy the strict QoS performance of task accomplishment, the vehicle firstly needs to make the task offloading decision to determine which tasks can be executed in the local server, and which tasks can be uploaded to the edge computing servers or nearby vehicles for processing. The aim of the task offloading is to jointly consider multiple factors, e.g., the resource availabilities, channel transmission condition, and energy consumption, to optimally transmit the tasks to meet the application requirements. After finishing the task offloading, the vehicle uploads the tasks to MEC servers or nearby vehicles through V2I and V2V wireless communication channel. Once received the tasks, AI algorithms deployed in the MEC server or nearby vehicle analyse, process, mining and execute the task. The analytical results would be sent back to the vehicle for further operations. The next subsection focuses on abstracting the working mechanism of VEC into a system model for further mathematical analysis.

3.2 Model abstraction

In order to quantitatively analyse the performance of VEC architecture, a new model is abstracted in this subsection to capture the unique features of mobile VEC compared with the traditional MEC architecture, including vehicle cooperation, bursty and heterogeneous features of task generation, priority-based service provisioning, and multiple server model and load sharing, as shown in Fig. 2.

Vehicle cooperation. Due to the high requirement for safety guarantee in IoV applications, V2V is introduced to provide redundancy for autonomous driving workloads, alleviate stringent performance and energy constraints on the vehicle devices and provide cooperation among vehicles for key IoV applications, e.g., see-through. To capture the communication redundancy, the model to be abstracted should consider both the V2I and V2V communications, where the tasks are offloaded to both the VEC servers and nearby vehicles for execution. Herein, multiple nearby vehicles are considered in the model abstraction, where N_v denote the total number of neighbour vehicles participating in the task offloading operation.

Bursty and heterogeneous features of tasks generated by AI algorithms. IoV systems are extremely complex; they tightly integrate many technologies, such as sensing, localisation, perception, decision making, as well as the smooth interactions with edge computing for high-definition map generation and data storage. The tasks generated by these complex applications and onboard sensors are highly heterogenous

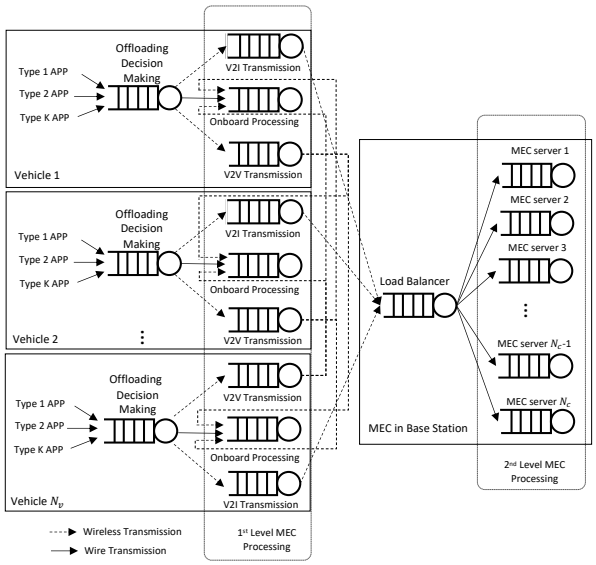

Fig. 2: System model abstraction for mobile VEC scenario

and bursty. For performance analysis, these two features should be captured and analysed.

Priority-based service provisioning. Compared with the traditional IoT devices, the vehicle in IoV systems has a certain amount of computation and storage resources. The tasks received come from two sources, generated by local applications and sensors and from nearby vehicles. The traditional First-In-First-Out (FIFO) traffic scheduling would result in high performance degradation for the local applications with a high volume of tasks from the nearby vehicles. In this situation, priority-based service provisioning should be deployed to facilitate a higher priority for the local tasks in service provisioning.

Multiple server model and load sharing. Different from cloud computing, one key principle in the MEC system is that the computation node should not be overloaded by the offloaded computation tasks to avoid the failure of task computation. A promising solution for this issue is to deploy multiple servers in an edge domain to reduce the outage of task computation and analysis. The model to be abstracted should consider the scenario of multiple server deployment and the load sharing among servers to improve the task acceptance ratio for IoV applications.

3.3 Model description

This study investigates a multiple-level VEC architecture consisting of N_v vehicles and N_c edge servers as shown in Fig. 2. According to the existing work in [28] [29], we assume the times spent in the modules of the abstracted model, including vehicle Offloading Decision Making (ODM), Vehicle Servers (VS), V2I Transmission (V2IT), V2V Transmission (V2VT), VEC Load Balancer (CLB), and VEC Servers (CS), follow the exponential distribution with the service rates of μ_{dM}, μ_{vS}, μ_{vI}, μ_{vV}, μ_{cB}, and μ_{cS}, respectively. Two kinds of service strategies are exploited in the abstracted model to process the incoming tasks: PQ and FIFO. To guarantee enough computation resources for local task computation, the onboard processing module uses the PQ strategy to process the computation requests, where the requests generated by the local devices enter a high-priority queue and the ones from the nearby vehicles go to the low priority queue. If multiple vehicles are cooperating to provide a single IoV service, e.g., objective detection, we assume the vehicle maintains a single buffer for the tasks sent to other vehicles as there is no processing priority for the tasks from the same IoV service. In this case, there will be only one low priority queue in the model abstraction. While if there are multiple services deployed at IoV systems, each vehicle will need to maintain multiple buffers for the tasks scheduled to the neighbour vehicles. In this regard, to make the proposed analytical model adaptable to multi-service deployment scenarios, there will be multiple low priority queues in the model abstraction, each of which will be used to store the tasks for a specific IoV service. When the higher priority queue becomes empty, the local server would provide the service for the tasks in the lower priority queue. Except for the onboard processing module, the other modules in the abstracted model use the FIFO

TABLE 1: THE NOTATION DEFINATION

Notations	Definitions
$\mu_{dM}, \mu_{vS}, \mu_{vI}, \mu_{vV}, \mu_{cB}, \mu_{cS}$	Service rates of ODM, VS, V2IT V2VT, CLB and CS components
$Q_{k,j}$	Infinitesimal generator of the jth application of the kth vehicle
$\Lambda_{k,j}$	Traffic rate matrix of the jth application of the kth vehicle
$\lambda_{1k,j}, \lambda_{2k,j}$	Traffic arrival rates when the Markov process is in state "1" and "2"
$\varphi_{1k,j}, \varphi_{2k,j}$	Transmission rates from state "1" to "2" and from "2" to "1"
B	Wireless channel bandwidth
L	Packet length
C_{vV}, C_{vI}	V2V and V2I channel throughputs
Lat_n, Lat_c, Lat_v	Average latency that tasks will be experienced at neighbour vehicle, VEC servers and local servers
ξ, η	Probability that a task will be processed locally or sent to VEC servers
ϑ_i	Probability that a task will be sent to the ith neighbour vehicle
$a_{s,k}$	Average arrival rate
$I_{s,k}(t_1)$	Degree of the task burstiness
$I_{s,k}$	Limited degree of the task burstiness
$\mu^3_{s,k}(t_1)$	The third centralised moment of the task burstiness
$h, h^{(2)}$	The first and second moments of the exponential service process
ρ	Server utilisation
Lat_{wa}	Virtual waiting time
g	Steady-state vector of the $\text{MMPP}^{\text{in}}_{\text{D,k}}$/M/1 queuing system
μ^e_{vL}, μ^e_{vH}	Effective service rates of decomposed SSSQ systems
$\pi_{vL,k}, \bar{\lambda_{vL,k}}$	Steady-state vector and the average task rate for $SSSQ_l$
$l_{vL,k}$	Queue length distribution
$\bar{\lambda_{vL,k}}$	The first moment of queue length distribution
ϵ	Threshold of iteration search process
Pb_c	Packet loss probability
K_v, K_c	Buffer sizes of vehicle and MEC servers
$\Re_{(f,l)}$	The case that $MMPP^{in}_{cS,j}$ is in the state of l and there are f tasks
$P_{f,l}$	Steady state probability that the queueing system is in the state $\Re_{(f,l)}$
$\mathbf{e}$	$2N_s$ dimension unit vector

strategy to process the arriving tasks to guarantee fairness among tasks execution. Within the VEC platform, the load sharing strategy shows superior performance against the non-sharing strategy. Therefore, the load sharing strategy in the CS module is to reallocate the incoming tasks into less overloaded servers. It is worth noting that the main focus of this study is on building a comprehensive analytical model for full task offloading strategies in VEC systems. In addition, the buffer sizes of the decision making module in vehicle and load balancer module in VEC are assumed to be infinite. While due to the limitation of computation resource available, there is a probability that the server is full and the new coming task cannot be served, resulting in serious QoS degradation in intelligent edge service provisioning. Thus, the buffer sizes of vehicle and VEC servers are set to be finite in the abstracted model, represented as K_v and K_c, respectively.

As discussed in the previous subsection, the tasks generated by IoV applications show a high degree of burstiness. Let N_k denote the number of the vehicular applications running in the kth vehicle. Then, for the jth application in the kth vehicle, the task arrival is modelled as an independent two-state MMPP, represented as $\text{MMPP}_{\text{tG,k,j}}$. Compared with the non-bursty Poisson process, MMPP is widely used to analyse the burstiness and correlation of Internet traffic [30]. The $\text{MMPP}_{\text{tG,k,j}}$ is characterised by an infinitesimal generator $Q_{k,j}$ and a rate matrix $\Lambda_{k,j}$ denoted as follows,

$$Q_{k,j} = \begin{bmatrix} -\varphi_{1_{k,j}} & \varphi_{1_{k,j}} \\ -\varphi_{2_{k,j}} & -\varphi_{2_{k,j}} \end{bmatrix} \quad \text{and} \quad \Lambda_{k,j} = \begin{bmatrix} \lambda_{1_{k,j}} & 0 \\ 0 & \lambda_{2_{k,j}} \end{bmatrix} \quad (1)$$

where $\lambda_{1_{k,j}}$ and $\lambda_{2_{k,j}}$ denote the arrival rates when Markov process is in state "1" and "2". The component $\varphi_{1_{k,j}}$ represents the transmission rate when Markov process transmits from state "1" to "2" and $\varphi_{2_{k,j}}$ is the rate from "2" to "1".

The service rates of V2V and V2I wireless communication channels are defined by $1/T_{vV}$ and $1/T_{vI}$. According to Shannon Theory, the throughput of a wireless communication system related to two factors, the Signal Interference Noise Ratio (SINR) and the transmission bandwidth B, obtained by,

$$C = B * log\,(1 + SINR) \quad (2)$$

where 5G BS exploits the pilot signal to estimate the channel condition to obtain SINR and configure B to satisfy the throughput requirement at each resource allocation frame. Let L represent the length of the task to be calculated and analysed. With a given transmission throughput, the service time of V2V and V2I link to transmit a task can be obtained by $T_{vV} = L/C_{vV}$ and $T_{vI} = L/C_{vI}$, where C_{vV} and C_{vI} stand for the transmission throughputs of V2V and V2I links, respectively.

4 Analytical Model Derivation

In this section, we provide the derivation methodology of the proposed analytical model. As shown in Fig. 2, the mean end-to-end latency, $\overline{Lat_{e2e}}$, can be expressed as follows,

$$\overline{Lat_{e2e}} = Lat_d + \sum_{i=1}^{N_v} \vartheta_i Lat_{n,i} + \eta Lat_c + \xi Lat_v \quad (3)$$

where Lat_d denotes the average waiting time for a task in the ODM module. After task offloading, the tasks will be sent to three destinations, neighbour vehicles, VEC servers and the local server, for data analysis and task computation. The probabilities that tasks will be assigned to and the latency that the tasks will be experienced in these three destinations are $\{\vartheta, \eta, \xi\}$ and $\{Lat_{n,i}, Lat_c, Lat_v\}$, respectively, where $\eta + \xi + \vartheta = 1$. Herein, we exploit a channel-aware scheduling strategy to calculate the probability that tasks are assigned to the ith vehicle for execution. Specifically, the probability, ϑ_i, is calculated by $\vartheta_i = \vartheta \frac{1}{T_{vV,i}} / \sum_{j=0}^{N_v} \frac{1}{T_{vV,j}}$, where $T_{vV,i}$ is the service time of V2V channel between the local vehicle and the ith neighbour vehicle.

It should be noticed that the method to calculate the probabilities is beyond the scope of this work. There have been a lot of research results appeared in the literature to investigate how to assign the tasks into three destinations for computation. With the aim of designing a general analytical model, not limited to a certain offloading strategy, we exploit a parametrised method to use the variables of ϑ_i, ξ and η to present the results of offloading decision making in the analytical model. For instance, the highly dynamic mobility of vehicles would affect the number of vehicles involved in the process of task offloading and the amount of tasks to be scheduled among vehicles. By instantiating offloading variables, ϑ_i, ξ and η and N_v, the parameterised approach makes the designed analytical model capable of adapting to different vehicle mobility scenarios and investigating the performance of offloading algorithms with diverse system configurations. By customising offloading variables according to their working scenario and conducting performance evaluation, the parameterised approach enables the designed analytical model to adapt to different vehicle mobility scenarios, investigating the performance of various offloading algorithms, and potentially applicable to diverse and hybrid communication protocols, such as 4G, 5G and IEEE 802.11p.

4.1 Latency in offloading decision making module

As shown in Fig. 2, the tasks received by ODM module come from multiple applications, each of which independently generates the requests for task computation and can be modelled as a two state MMPP process, denoted as $\text{MMPP}_{\text{tG,k,j}}$ and characterised by an infinitesimal generator $Q_{tG,k,j}$ and a rate matrix $\Lambda_{tG,k,j}$ Then the task arrival at ODM module would be the superposition of multiple independent MMPP process. According to [31], the superposition of multiple independent MMPP processes results in a new MMPP process. Let $\text{MMPP}^{\text{in}}_{\text{D,k}}$ denote the total task arrival at ODM module characterised by an infinitesimal generator $Q^{in}_{D,k}$ and a rate matrix $\Lambda^{in}_{D,k'}$ which can be calculated as follows,

$$\begin{aligned} Q^{in}_{D,k} &= Q^{in}_{tG,k,1} \oplus Q^{in}_{tG,k,1} \oplus \cdots \oplus Q^{in}_{tG,k,N_k-1} \oplus Q^{in}_{tG,k,N_k} \\ \Lambda^{in}_{D,k} &= \Lambda^{in}_{tG,k,1} \oplus \Lambda^{in}_{tG,k,2} \oplus \cdots \oplus \Lambda^{in}_{tG,k,N_k-1} \oplus \Lambda^{in}_{tG,k,N_k} \end{aligned} \quad (4)$$

where $\oplus$ represents the operator of Kronecker sum. N_k denotes the number of the applications in the kth vehicle. Compared with the two state $\text{MMPP}_{\text{tG,k,j}}$, the state space of $\text{MMPP}^{\text{in}}_{\text{D,k}}$ would be much larger after the state superposition, which makes the mode derivation intractable. To address this issue, the approximation method in the work

in [32] is used to design a two-state MMPP process, denoted as $\widehat{MMPP}^{in}_{D,k}$, to approximate the multi-state $\text{MMPP}^{\text{in}}_{\text{D,k}}$ process. This kind of approximation operation brings the benefit of making a complex network system to be tractable for performance analysis with satisfied approximation accuracy and has been appeared in the performance analysis of high-performance computing architecture and software defined network architecture. During the approximation, the aim is to compute an infinitesimal generator $\widetilde{Q}^{in}_{D,k}$ and a rate matrix $\widetilde{\Lambda}^{in}_{D,k}$ from the four characteristics of multi-state $\text{MMPP}^{\text{in}}_{\text{D,k}}$ process, which are the average arrival rate $a_{s,k}$, the degree of the task burstiness, $I_{s,k}(t_1)$, the limited degree of the task burstiness, $I_{s,k}$, and the third centralised moment of the task burstiness, $\mu^3_{s,k}(t_1)$. The work in [33] presents the methodology of how to compute the individual four parameters, $a_{k,j}(t_1), I_{k,j}, I_{k,j}(t_1), \mu^3_{k,j}(t_1)$ from $Q_{tG,k,j}$ and $\Lambda_{tG,k,j}$ of the jth MMPP process. Then the merged four parameters, $a_{s,k}$, $I_{s,k}(t_1)$, $I_{s,k}$ and $\mu^3_{s,k}(t_1)$, can be calculated as follows,

$$\begin{aligned} a_{s,k} &= \sum_{j=1}^{N_k} a_{k,j}(t_1) \\ I_{s,k}(t_1) &= \sum_{j=1}^{N_k} I_{k,j}(t_1)\frac{a_{k,j}(t_1)}{a_{s,k}} \\ I_{s,k} &= \sum_{j=1}^{N_k} I_{s,k}(t_1)\frac{a_{k,j}(t_1)}{a_{s,k}} \\ \mu^3_{s,k}(t_1) &= \sum_{j=1}^{N_k} \mu^3_{k,j}(t_1) \end{aligned} \tag{5}$$

After obtaining the parameters of $a_{s,k}$, $I_{s,k}(t_1)$, $I_{s,k}$ and $\mu^3_{s,k}(t_1)$, the transfer algorithm proposed in [34] calculates the infinitesimal generator $\widetilde{Q}^{in}_{D,k}$ and the rate matrix $\widetilde{\Lambda}^{in}_{D,k}$ of $\widehat{MMPP}^{in}_{D,k}$. With an exponential distribution service process, the task processing can be modelled as a $\text{MMPP}^{\text{in}}_{\text{D,k}}$/M/1 queuing system. The average waiting time in the $\text{MMPP}^{\text{in}}_{\text{D,k}}$/M/1 queuing system can be calculated by,

$$Lat_d = \frac{1}{2(1-\rho)\rho}\left\{Lat_{wa} - (1-\rho)\lambda_t h^{(2)}\right\} \tag{6}$$

where $h^{(2)}$ is the second moment of the exponential service process, calculated by $h^{(2)} = [E(s_d(t))]^2 - E\left(s_d(t)^2\right)$, ρ is the service utilisation of the ODM server, obtained by $\rho = \frac{\lambda_t}{\mu_d}$, and Lat_{wa} is the virtual waiting time for a virtual task arrived at the ODM server, calculated by,

$$Lat_{wa} = \left[2\rho + \lambda_t h^{(2)} - 2h\left((1-\rho)g + h\pi\Lambda\right)(Q + e\pi)^{-1}\right]\lambda \tag{7}$$

where π is the steady-state vector of the Markov chain, given by $\pi = \frac{1}{\varphi_{d1,k}+\varphi_{d2,k}}\left(\varphi_{d2,k}, \varphi_{d1,k}\right)$. λ_t is the average arrival rate. h and g are the first moment of the exponential service process ($h = [E(s_d(t))]$) and the steady-state vector of the $\text{MMPP}^{\text{in}}_{\text{D,k}}$/M/1 queuing system.

The output of the $\widehat{MMPP}^{in}_{D,k}/M/1$ queue would be spited into three parts, the tasks executed in vehicle servers (denoted as $\text{MMPP}^{\text{out}}_{\text{vS}}$), the tasks sent to VEC servers ($\text{MMPP}^{\text{out}}_{\text{cS}}$) and tasks offloaded to neighbour vehicles ($\text{MMPP}^{\text{out}}_{\text{nS}}$). Section. 4.2 will discuss how to split a MMPP process and obtain the key parameters of the new MMPP process including the infinitesimal generator and the rate matrix.

4.2 Latency for task execution in nearby vehicle servers

From Fig. 2, the tasks delivered through V2V link will be popped in the low-priority queue of the server of nearby vehicle. Thus, the average latency, Lat_n, can be computed by,

$$Lat_n = Lat_{vV} + Lat_{nL} \tag{8}$$

where Lat_{vV} and Lat_{nL} denote the transmission latencies in V2V wireless channel, and processing latency in nearby vehicle server, respectively.

As there are N_v neighbour vehicles, the input of the ith V2V channel, denoted as $\text{MMPP}^{\text{in}}_{\text{vV,k,i}}$, is the split of the output of the ODM module, $\text{MMPP}^{\text{out}}_{\text{nS,k}}$. Given the probability that a task will be sent to the ith neighbour vehicle for execution as a, the rate matrix $\Lambda^{in}_{vV,k,i}$ and an infinitesimal generator $Q^{in}_{vV,k,i}$ can be obtained by $\widetilde{\Lambda}^{in}_{vV,k,i} = \vartheta_i\widetilde{\Lambda}^{out}_{nS,k}$ and $\widetilde{Q}^{in}_{vV,k,i} = \widetilde{Q}^{out}_{nS,k}$. Given an average serve time for the ith V2V channel, $\mu_{vV,i} = 1/T_{vV,i}$, the transmission process can be modelled as a $\text{MMPP}^{\text{in}}_{\text{vV,k}}$/M/1 queueing system. Then the transmission latency could be easily obtained by inserting the infinitesimal generator, $\widetilde{Q}^{in}_{vV,k,i}$, and the rate matrix, $\widetilde{\Lambda}^{in}_{vV,k,i}$, in Eq. (6). After obtaining the transmission latency, this subsection will derive the processing latency for the tasks in the low priority queue of the nearby vehicle.

In the multi-level PQ system with N_k queues, the tasks in the lower priority queue receive the service only when the higher priority queues become empty. Therefore, processing latency for the new arrival task in the low priority queue is equal to the sum of the serving time and the waiting time that this task waited in the queue until all tasks of both the higher priority queues and low priority queue are served. Directly building the Markov chain to derive the steady-state probability distribution and calculate the average processing latency is quite challenging. To address this issue, instead of analysing the behaviour of a multi-level PQ-based queuing system, the queuing decomposition technology is exploited in this subsection to split the multi-level PQ queue system into multiple relatively simple Single-Server-Single-Queue (SSSQ) systems.

The key point for decomposing a multi-level PQ-based queueing system into multiple SSSQ systems is to calculate the effective service rate, μ^e_v for individual SSSQ server. Let μ^e_{vH} denote the effective service rate for the highest priority queue. Because the highest priority queue has the absolute priority in the task computation, it can readily obtain that the effective service rate, μ^e_{vH}, is equal to the rate of the local server, μ_{vS}, presented as $\mu^e_{vH} = \mu_{vS}$. Therefore, the difficulty of decomposing the multi-level PQ into multi-level SSSQ systems is how to calculate the service rate for the low priority SSSQ system, μ^e_{vL}. To address this issue, inspired by our preliminary work in [35], which focuses on the queueing decomposition for a pre-emptive priority queue system, we develop a new queueing decomposition approach, named Multi-Level Empty Buffer Approximation (MLEBA). The key idea of the MLEBA is to first derive

the average number of tasks in the lower priority queue (L_{vL}), and then search a proper service rate, μ^e_{vL} to satisfy the relationship between the service rate and the average number of tasks in the system. In the MLEBA, the number of tasks in the low priority queues is calculated by deducting the average number of tasks in the highest priority (L_{vH}) from the total number of tasks in the multi-level PQ system (L_{vT}), expressed as,

$$L_{vL,k} = L_{vT,k} - L_{vH,k} \tag{9}$$

Let $MMPP^{in}_{vH,k}$ and $MMPP^{in}_{vT,k}$ denote the task arrival at the highest priority queue and the whole multi-level PQ system. The tasks arriving at the highest priority queue come from the output of ODM module, so $MMPP^{in}_{vH,k}$ is equal to $MMPP^{out}_{vS,k}$. The tasks arriving at the multi-level PQ system come from two sources, the local ODM module and the V2V transmission from multiple neighbour vehicles, therefore, $MMPP^{in}_{vT}$ is the superposition of $MMPP^{out}_{vS,k}$ and $MMPP^{out}_{vV,j}$, where parameter j denotes the index of the neighbour vehicle. According to [34], the average queue length is given as,

$$\begin{aligned} L_{vL,k} =& l^{(1)}_{vL,k} + \left[\frac{1}{\lambda_{\bar{vL},k}}\pi_{vL,k}\Lambda_{vL,k} - l_{vL,k}\right] \\ & \left(e\pi_{vL,k} + Q_{vL,k}\right)^{-1}\Lambda_{vL,k}e \end{aligned} \tag{10}$$

where $\pi_{vL,k}$ and $\lambda_{\bar{vL},k}$ are the steady-state vector and the average task rate for $SSSQ_l$, calculated by, $\lambda_{\bar{vL},k} = \frac{\lambda_{vL1,k}\varphi_{vL2,k}+\lambda_{vL2,k}\varphi_{vL1,k}}{\varphi_{vL2,k}+\varphi_{vL1,k}}$. $l_{vL,k}$ denotes the queue length distribution and $l^{(1)}_{vL,k}$ is the first moment of $l_{vL,k}$, which are calculated by applying "z-transform" to the transmission probability matrix of $\mathrm{MMPP}^{\mathrm{in}}_{\mathrm{vL,k}}$/M/1 [34]. The search process is conducted through recursively comparing the queue length for the low priority queue from Eq. (9), $L_{vL,k}$, and the result from Eq. (10), $\widetilde{L_{vL,k}}$, until the comparison meets the recursive stop condition, $\left|L_{vL,k} - \widetilde{L_{vL,k}}\right| < \epsilon$. When the iteration process is finished, we can obtain the overall service rate for the low priority queues.

To this end, the original multi-level PQ system is decomposed into an SSSQ system with the service rate of μ^e_{vH}, and a new PQ system with $N_k - 1$ queues and the service rate of μ^e_{vL}. After repeatedly applying the proposed MLEBA approach to the new PQ system, we could obtain the equivalent service rates for multiple SSSQ systems and the average latency for a specific SSSQ system, Lat_{nL}, could be calculated from inserting the model parameters of $MMPP^{in}_{vL,k}/M/1$ in Eq. (6).

4.3 Latency for task execution in VEC servers

This subsection calculates the latency experienced by the tasks allocated to the VEC servers, which pass through the modules of V2I, CLB, and VECS. So, the latency, Lat_v is given by,

$$Lat_c = Lat_{vI} + Lat_{cB} + Lat_{cS} \tag{11}$$

where Lat_{vI}, Lat_{cB}, and Lat_{cS} are the latencies in the modules of V2I, CLB and CS. The input of V2I module comes from the output of ODM module. So the infinitesimal generator $Q^{in}_{vI,k}$ and the rate matrix $\Lambda^{in}_{vI,k}$ of $MMPP^{in}_{vI}$ are equal to those of $MMPP^{out}_{vC}$. Given the service rate of V2I module, μ_{vI}, calculated by Eq. (2), the V2I latency, Lat_{vI} could be from Eqs. (6-7).

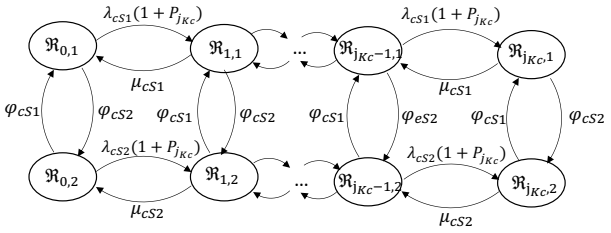

Fig. 3: Markov chain for calculating the block probability in the jth server

As the VEC platform serves multiple vehicles within its coverage, the outputs of V2I modules would be merged in the VLB module. Thus, the input of VLB, $MMPP^{in}_{cB}$, is the superposition of multiple two-state MMPP processes, $MMPP^{out}_{vI,k}$. Then, the infinitesimal generator $Q^{in}_{cB,k}$ and the rate matrix $\Lambda^{in}_{cB,k}$ are calculated from Eq. (4). As described in Subsection. 4.1, the space number of $MMPP^{in}_{cB,k}$ with the superposition of K_v two-state $MMPP^{out}_{vI,k}$, is $K_v + 1$, which is no more than a two state MMPP process and poses significant challenges for obtaining the average end-to-end latency. To address this issue, we use a new two state MMPP process $\widetilde{MMPP^{in}_{cB}}$ to approximate the $K + 1$ state MMPP process $MMPP^{in}_{cB}$. This process is to calculate a new infinitesimal generator $\widetilde{Q^{in}_{cB}}$ and the rate matrix $\widetilde{\Lambda^{in}_{cB}}$ from K_v $MMPP^{in}_{cB,k}$. The details of the approximation process have been discussed in Subsection. 4.1.

The CLB module is in charge of allocating and balancing the tasks among multiple servers. Thus, the arriving tasks would be equally splitted and assigned to the server for task computation. Let $MMPP^{in}_{cS,l}$ represent the tasks received by the lth server. As the queue space is set to be finite in the abstracted model, there is a probability, denoted as Pb_c, that the newly arrived task is rejected by a busy server because the buffer is full. To fully utilise the server resources and improve the service QoS, the rejected task will be assigned to another MEC server to improve the acceptance rate. The task arrival for the lth server come from two sources, the uploaded tasks $MMPP^{in_u}_{cS,l}$ and the ones that have been rejected by servers respectively. For the lth server, let $MMPP^{rej}_{cS,l}$ denote the tasks that can't enter the queue, which is a fraction, Pb_c, of tasks arriving, denoted as $MMPP^{in}_{cS,l}$. Then the total task arrival at the $MMPP^{in}_{cS,l}$ server, is the superposition of one $MMPP^{in_u}_{cS,l}$ process, and multiple $MMPP^{rej}_{cS,l}$ processes.

Let K_c denote the maximal number of tasks that can enter the server system. Then the service with a bursty $MMPP^{in}_{cS,j}$ arrival is modelled as a $MMPP^{in}_{cS,j}/M/1/K_c$ system. It is characterised by a two state Markov chain as shown in Fig. 3. State $\mathfrak{R}_{(f,l)}$, where $f \in [0,1,...,K_c]$ and $l \in [0,1]$, represents the case that $MMPP^{in}_{cS,j}$ is in the state of l and there are f tasks in the $MMPP^{in}_{cS,j}/M/1/N_s$ system. The transmission from the state $\mathfrak{R}_{(f,l)}$ to $\mathfrak{R}_{(f+1,l)}$ means there is a new task enters the system with the arrival rate λ_{cSl}. The transmission from the state $\mathfrak{R}_{(f,l)}$ to $\mathfrak{R}_{(f-1,l)}$ means a task finishes its service with the service rate λ_{cSl}. Transmission

from state $\mathfrak{R}_{(f,1)}$ to $\mathfrak{R}_{(f,0)}$ or from state $\mathfrak{R}_{(f,0)}$ to $\mathfrak{R}_{(f,1)}$, means that the $MMPP^{in}_{cS,j}$ changes from the state 1 to 0 with the transmission rate φ_{cS1}, or 0 to 1 with φ_{cS2}.

From Fig. 3, we can build the transmission rate matrix, G. Let $P_{f,l}$ denote the steady state probability that the queueing system is in the state $\mathfrak{R}_{(f,l)}$. And the steady state probability matrix, $\mathbf{P} = [P_{0,0}, P_{1,0}, ...P_{f_K,0}, P_{0,1}, P_{1,1}, ..., P_{f_K,1}]$, holds the following relation with $\mathbf{G}$,

$$\mathbf{PG} = 0 \quad \text{and} \quad \mathbf{Pe} = 1 \tag{12}$$

where $\mathbf{e}$ is a $2K_c$ dimension unit vector. The method in [36] could be exploited to solve the above equations and yield the following solutions,

$$\mathbf{P} = \mathbf{h}(\ \mathbf{I}\ - \mathbf{H} + \mathbf{eh})^{-1} \tag{13}$$

where $\mathbf{H} = \mathbf{I} + \mathbf{G}/min\{\mathbf{G}(k,k)\}$. $min\{\mathbf{G}(k,k)\}$ is the minimum value of the diagonal elements of matrix $\mathbf{G}$. h is an arbitrary vector or matrix $\mathbf{H}$. After obtaining the steady-state probability of $MMPP^{in}_{cS,j}/M/1/N_s$ system, the block probability is calculated by,

$$Pb_{N_s} = P_{N_s,1} + P_{N_s,2} \tag{14}$$

As shown in Fig. 3, the arrival matrix is a function of Pb_{N_s}. The Eq. (14) denotes that the steady-state probability distribution is a function of the parameters of the arrival process. It is difficult to calculate Pb_{N_s} directly from Eqs. (13) to (14). To address this issue, the method in [35] is used to apply Pb_{N_s} in Eqs. (12) to (14) to obtain an updated $\hat{P_{j_K}}$ and repeat this substitution until $\hat{P_{N_s}}$ approaches to Pb_{N_s}, defined as, $\left|\hat{Pb_{N_s}} - Pb_{N_s}\right| < \epsilon$, where ϵ is the stop threshold.

4.4 Latency for task execution in local servers

The tasks assigned to the local vehicle server has a high priority in the resource competition in the PQ processing. Thus, the effective service rate, μ^e_{vH}, is equal to the service rate of the local server, μ_{vS}, and the task processing in the local server of the kth vehicle for the high priority queue can be modelled as a $MMPP^{in}_{vS,k}/M/1/K_v$ with the buffer size of the high priority queue as K_v. If the buffer is full, the new arrival tasks will be dropped to avoid the system being overloaded and long-time response. Let $Pb_{vS,k}$ indicate the probability that a task finds the system is full in the kth vehicle. Then the effective tasks that finally receive the service is a fraction $1 - Pb_{vS,k}$ of the total arrivals. Let $Q^{in}_{vS,k}$ and $\Lambda^{in}_{vS,k}$ be the infinitesimal generator and the rate matrix of $MMPP^{in}_{vS,k}$ process. As the splitting of an MMPP generates a new MMPP process. Let $MMPP^{in\sim e}_{vS,k}$ denote the effective arrival process, and $Q^{in\sim e}_{vS,k}$ and $\Lambda^{in\sim e}_{vS,k}$ be the infinitesimal generator and the rate matrix of $MMPP^{in\sim e}_{vS,k}$. Based on the splitting principle of the MMPP process in [33], the infinitesimal generator and the rate matrix of $MMPP^{in\sim e}_{vS,k}$ could be calculated by,

$$Q^{in\sim e}_{vS,k} = Q^{in}_{vS,k} \quad \text{and} \quad \Lambda^{in\sim e}_{vS,k} = \left(1 - Pb_{vS,k}\right)\Lambda^{in}_{vS,k} \tag{15}$$

The derivation of block probability, $Pb_{vS,k}$, in vehicle server is similar to that of VEC server. While different from calculating Pb_{N_s} through interactive computation approach, the input of task arrival is not a function of the steady state probability and $Pb_{vS,k}$ could be calculated directly by inserting the $Q^{in\sim e}_{vS,k}$ and $\Lambda^{in\sim e}_{vS,k}$ in Eqs. (12) and (13) to calculate the transmission rate matrix, G, the steady state probability P. After obtaining the task block probability, the effective infinitesimal generator and rate matrix can be readily obtained in Eq. (15). Then the latency for task execution in the local server, Lat_{vS} can be achieved by applying the $Q^{in\sim e}_{vS,k}$, $\Lambda^{in\sim e}_{vS,k}$, μ^e_{vS} in Eqs. (6) and (7).

Finally, with the average latencies of a task executed in the ODM model (Lat_d), the local server (Lat_v), the nearby vehicle (Lat_n) and VEC servers (Lat_c), combining with the probabilities that the tasks will be executed in different destination servers, $\{\vartheta, \eta, \xi\}$, the average end-to-end latency could be obtained from Eq. (3).

5 Validation of the Model

In this section, we will first validate the accuracy of the analytical model and then use the proposed model as a practical tool to analyse the performance and system design of the mobile VEC.

5.1 Accuracy validation and performance evaluation

To validate the accuracy of the proposed analytical model, we developed an event-driven VEC system model based on Objective Modular Network Testbed in C++ (Omnet++) [37]. As shown in Fig. 2 , the VEC network in the simulation consists of the modules of offloading decision making, vehicle server, V2V and V2I communication channel, edge load balancer and multiple MEC servers. 95% confidence criterion is used in the simulator to determine the steady state of the simulation experiments for data collection. For each network configuration, the simulation runs 100 times and the total requests are set to be 10^7 [16]. Both the broadcast and unicast modes of DSRC protocol are used in this work to establish communications with the neighbour vehicles. We have performed extensive simulation experiments to estimate the accuracy of the proposed analytical models with different system configurations, such as the different service rates of the various processing modules, buffer size, different MMPPs traffic input, the number of the applications in each vehicle, the number of the servers in VEC domain, and vehicle mobility. However, for the sake of specific illustration and without loss of generality, the results are presented for the performance validation with the following system parameters, the service times for the modules of ODM, VS, CLB, CS, $1/\mu_{dM}$, $1/\mu_{vS}$, $1/\mu_{cB}$, and $1/\mu_{cS}$, to process a task are set to be 0.005s, 0.025s, 0.001s, 0.01s [16]. The number of the servers in one VEC domain is set to 8 [15]. The task size is set to be 2 MB and the channel bandwidth is 5 MHz [38]. The VEC domain radius is set to be 2 km and vehicles' speed is 30km/h. The probability that a task will be executed at the local server, nearby vehicles and VEC servers are set to be [0,1]. The buffer sizes of VS and CS are 32/16/8/4.

5.1.1 Prediction accuracy of the proposed model by varying the rates of traffic arrival

Fig. 4 shows the performance results of the end-to-end latency predicted by the proposed analytical model against those of the simulation experiments by varying the traffic

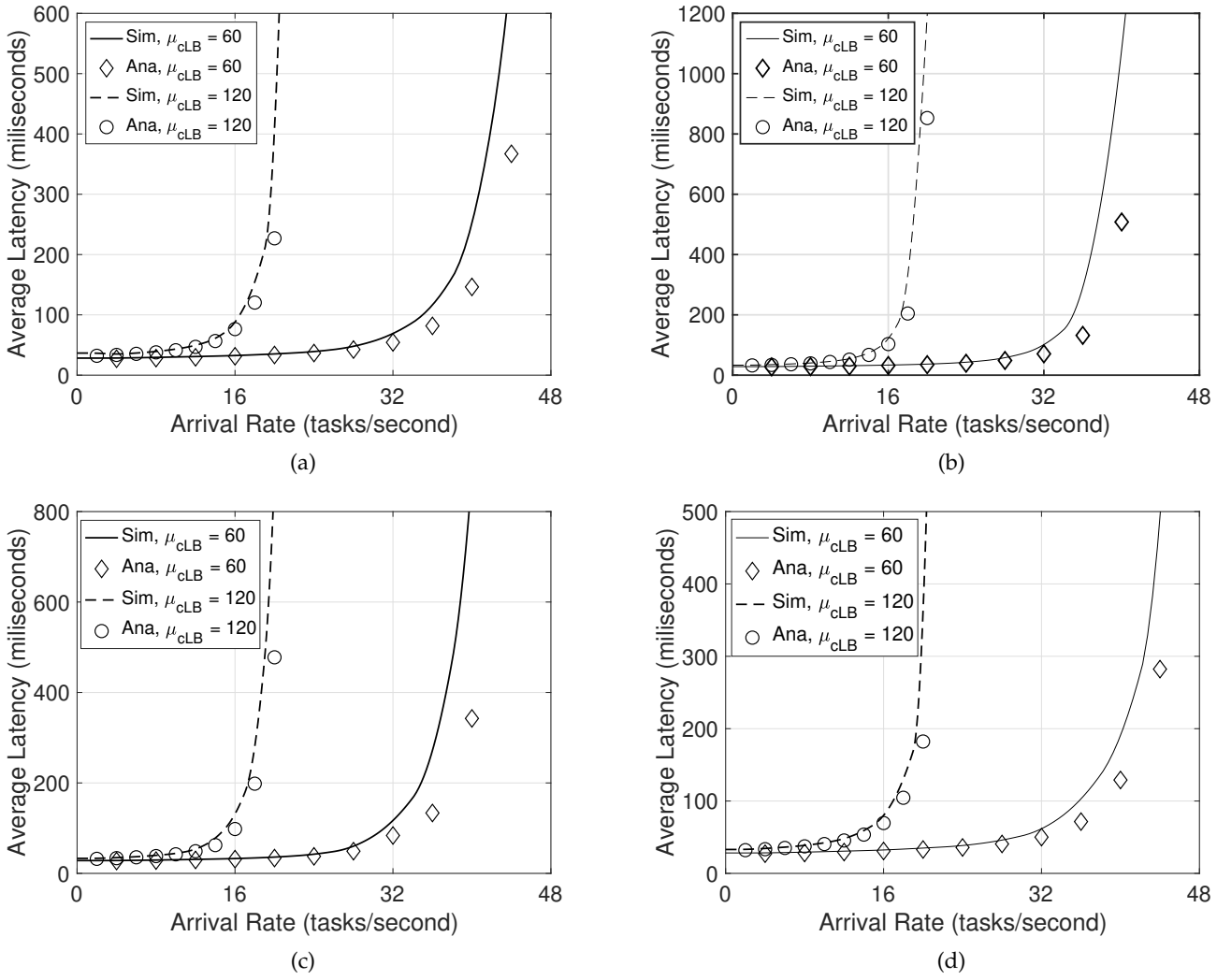

Fig. 4: Latency predicted by the model and simulation: T_{dM} = 0.005, T_{vS} = 0.0.0125, T_{cB} = 0.001, T_{cS} = 0.01, ξ = 0.15, η = 0.55, K_v = 16, K_c = 32, N_v = 6, N_c = 8, (a) φ_{1k} = 0.8, φ_{2k} = 0.5, (b) φ_{1k} = 0.7, φ_{2k} = 0.4, (c) φ_{1k} = 0.3, φ_{2k} = 0.2, (d) φ_{1k} = 0.9, φ_{2k} = 0.7.

arrival rate. In these figures, the horizontal axis represents the traffic rate when the MMPP process is in the state of "1". For the sake of presentation clarity, the traffic rate is set to be zero when the MMPP process is in the state of "2". The vertical axis represents the end-to-end latency. The transmission rates $\varphi^{in}_{tG,1}$ and $\varphi^{in}_{tG,2}$ of the infinitesimal generator, Q^{in}_{tG}, are presented in the title description. Furthermore, Fig. 5 depicts the end-to-end latency by varying the value of ϑ and the number of vehicles participating in task offloading. As shown in this figure, the performance results achieved by the proposed analytical model match well with these obtained from the simulation experiments when the system runs under the unsaturated condition. While the system is approaching the saturation condition, there is a slight gap between the results predicted by the analytical model and these collected from the simulation experiments. These prediction errors stem from the relaxation processing adopted in the model derivation, e.g., the PQ decomposition in Eq. (9) and the MMPP split processing in Eq. (15). Otherwise, it is difficult to achieve a conservative and closed-form latency prediction.

5.1.2 Prediction accuracy of the proposed model by varying the burstiness of traffic arrival

Furthermore, we investigate the accuracy of the proposed analytical model with different burstiness of traffic arrival. To model the burstiness of traffic arrival, the squared coefficient of variation (SCV) is widely used to measure the traffic burstiness, which is expressed as, $C_S^2 = 1 + \frac{2(\lambda_1-\lambda_2)^2\varphi_1\varphi_2}{(\varphi_1+\varphi_2)^2(\lambda_1\lambda_2+\lambda_1\varphi_2+\lambda_2\varphi_1)}$. It can be observed that for the traditional Poisson process, $\lambda_1 = \lambda_2$ and $C_S^2 = 1$. Different from the Poisson process, MMPP could capture the bursty feature of the traffic arrival with different values of traffic arrival rates, λ_1 and λ_2, and the transmission rates from/to "1" to/from "2", φ_1 and φ_2. By varying the values of C_S^2, the end-to-end latency predicted by the analytical model and collected from simulation experiments are shown in Table II. Furthermore, we compared the performance accuracy of the proposed analytical model with that of the classical queueing theory-based approach with Poisson traffic arrival

TABLE 2: Average end-to-end latency by varying the burstiness of traffic arrival

						$\overline{Lat_{e2e}}$(milliseconds)		
C_S^2	$\bar{\lambda}$	λ_1	λ_2	σ_1	σ_2	Sim	Ana_Poisson	Ana_MMPP
1	8	8	8	0.7	0.7	30.52811145	29.40215564	—
2	8	15.14980315	3.531373033	0.8	0.5	31.08424112	29.40215564	29.54174171
5	8	18.6509668	1.343145751	0.8	0.5	31.71442412	29.40215564	30.15915666
10	8	27.18729572	0.804764106	0.8	0.3	33.40557162	29.40215564	31.43594912
20	8	28.59287252	0.277672805	0.8	0.3	33.89283464	29.40215564	31.95688699
50	8	70.55274253	0.180907184	0.8	0.1	59.80592358	29.40215564	53.26107401
100	8	71.66126125	0.042341356	0.8	0.1	61.40599251	29.40215564	54.19060525
150	8	135.1984105	0.050099347	0.8	0.05	186.6320839	29.40215564	147.8558889
200	8	135.5956753	0.025270297	0.8	0.05	188.1466454	29.40215564	150.3185931

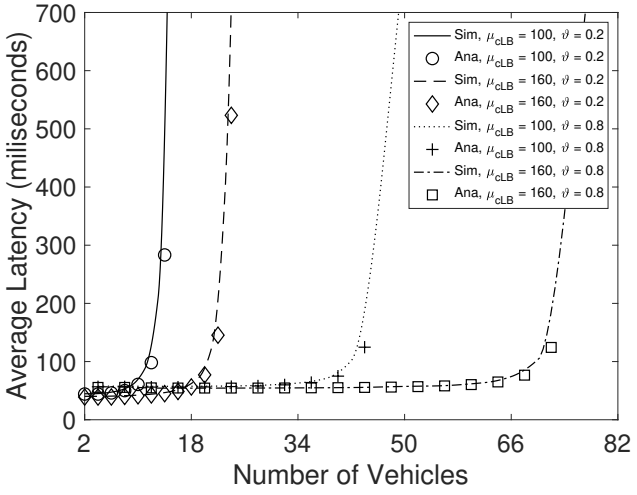

Fig. 5: Latency predicted by the proposed analytical model and simulation experiments with varying numbers of vehicles: λ_{1k} = 16, λ_2 = 0, φ_{1k} = 0.7, φ_{2k} = 0.35, T_{dM} = 0.005, T_{vS} = 0.033, T_{cB} = 0.001/0.00625, T_{cS} = 0.01, ξ = 0.2, η = 0.6, K_v = 16, K_c = 32, N_c = 8.

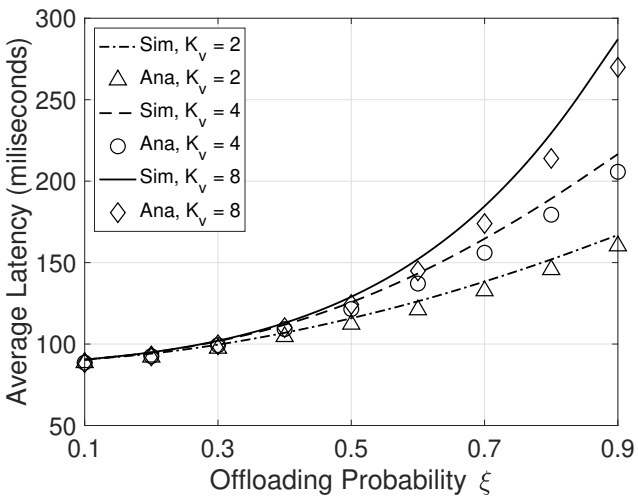

Fig. 6: The impact of the task execution probability (ξ) on the end-to-end latency: λ_{1k} = 30, λ_{2k} = 0, φ_{1k} = 0.7, φ_2 = 0.35, T_{dM} = 0.0033, T_{vS} = 0.05, T_{cB} = 0.001, T_{cS} = 0.01, $\eta = (1-\xi)/2$, K_v = 4/8/16/32, K_c = 32, N_v = 4, N_c = 8.

[15] in Table II. It can be seen that the proposed analytical model outperforms the classical queueing theory approach in terms of the prediction accuracy with different traffic burstiness. This accuracy improvements comes from the unique design in the proposed model, which can capture the bursty feature of the arrival traffic and resource consumption competition among multiple vehicular applications.

5.2 Performance analysis

After validating the accuracy of the proposed model above, we will use this model to analyse the design and parameter optimisation of VEC systems in this subsection.

5.2.1 The impact of the PQ scheduling

To provide differentiated services for the local vehicle and nearby vehicles, the PQ based scheduling algorithm is used in this study to serve the task arrival. This subsection uses the developed analytical model to investigate the impact of the PQ strategy on end-to-end performance. Fig. 6 shows

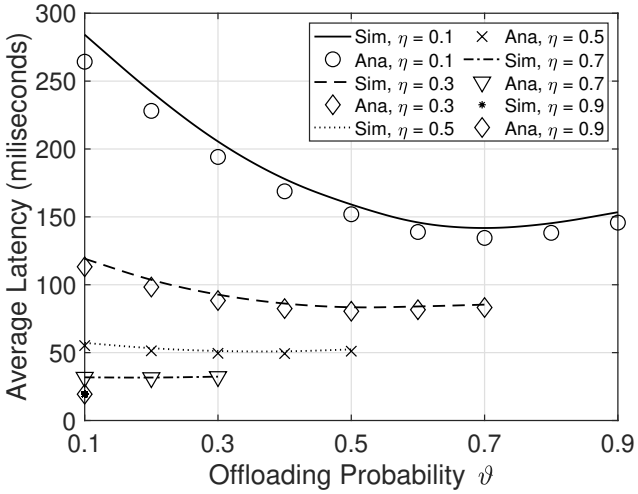

Fig. 7: The impact of the task execution probability (η and ϑ) on the end-to-end latency: λ_{1k} = 16, λ_2 = 0, φ_{1k} = 0.8, φ_{2k} = 0.5, T_{dM} = 0.0033, T_{vS} = 0.05, T_{cB} = 0.001, $\xi = 1-\eta-\vartheta$, T_{cS} = 0.002, K_v = 16, K_c = 32, N_v = 4, N_c = 8.

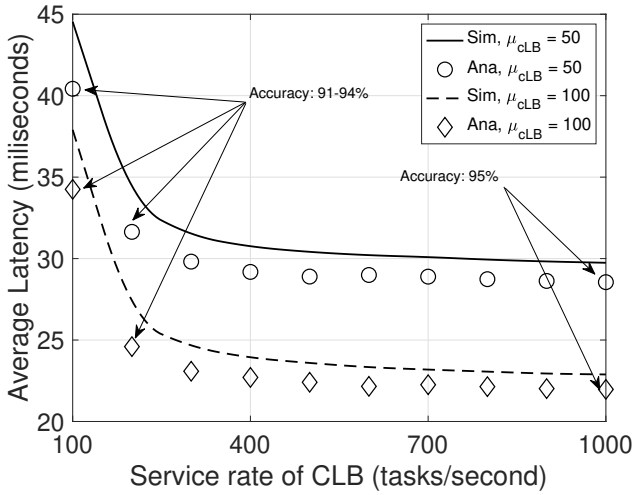

Fig. 8: The impact of CLB resources allocation on the end-to-end latency with different server service time: λ_{1k} = 30, λ_2 = 0, φ_{1k} = 0.7, φ_{2k} = 0.35, T_{dM} = 0.005, T_{vS} = 0.005, T_{cS} = 0.02/0.01, ξ = 0.15, η = 0.55, K_v = 4, K_c = 32, N_v = 4, N_c = 8.

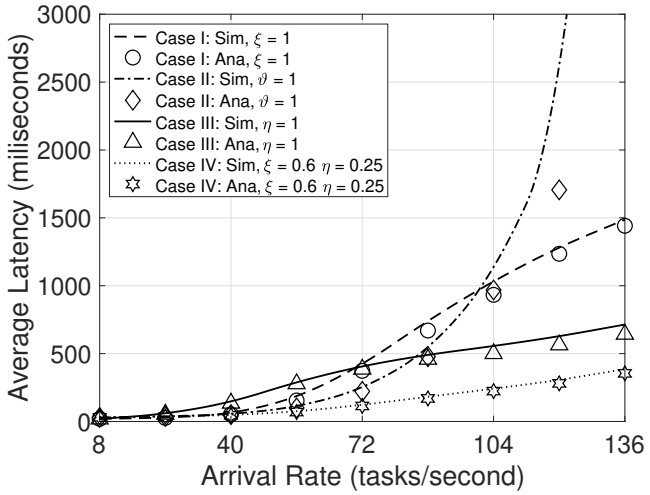

Fig. 9: The impact of the PQ task execution on the end-to-end latency with different buffer size: φ_{1k} = 0.8, φ_{2k} = 0.3, T_{dM} = 0.005, T_{vS} = 0.0125, T_{cB} = 0.0001, T_{cS} = 0.005, K_v = 16, K_c = 32, N_v = 4, N_c = 3.

the results of end-to-end latency by pouring different proportional traffic to the higher priority queue. The results reveal that the more tasks executed in the onboard vehicle server, the longer waiting time the packets from the neighbour vehicle would experience. This is because, for a PQ scheduling algorithm, the packets in the lower priority queue can be served only when the high priority queue becomes empty. Therefore, if the high priority queue is quite busy, the packets in the lower queue would experience a long time of waiting, which would result in delay outage for some mission-critical vehicular applications. Therefore, to address this issue, the task offloading should be conducted to be aware of the busyness level of the servers of the nearby vehicles. If the nearby vehicle is in a state of high occupation, the ODM module should send more tasks to the VEC server for execution.

Furthermore, as the probability (ξ, η and ϑ) that a task is executed at the local vehicle, neighbour vehicles and MEC servers, is dynamic, it is required to systematically investigate the average end-to-end latency. Therefore, we further evaluate the accuracy of the proposed analytical model with different combinations of ξ, η and ϑ. Specifically, the values of ϑ are set to be 0.1 to 0.9 with an interval of 0.1, η is in the set of [0.1, 0.3, 0.5, 0.7, 0.9], and $\xi = 1 - \vartheta - \eta$. The results are shown in Fig. 7, from which we can observe that the proposed analytical model can accurately and systematically analyse the average end-to-end latency of offloading decision making in VEC systems. In addition, it is observed that the average latency decreases inversely proportional to the increase of the probability that tasks are sent to MEC servers. This is because more computation resources are usually deployed at the network edge compared with vehicles. This strategy would significantly shorten the latency of task execution when the V2I links remain stable.

5.2.2 *Impact of the edge resource allocation*

The tasks offloaded to the edge computing platform will be scheduled to different edge servers for execution. Therefore, the resource allocation in the edge server plays an important role in IoV application performance guarantee. To investigate the impact of resource allocation at edge computing servers on the end-to-end performance of the IoV applications, Fig. 8 presents the average latency predicted by the proposed analytical model by varying the average service rate at the CLB module. It can be seen that the increased processing capacity of CLB has two impacts on the end-to-end performance. On one hand, the reduced end-to-end latency is easily understandable, because with the given arrival traffic rate, the improved processing capability or service rate means the reduced service time at the server and shortened waiting time. As shown in Fig. 8, the increases in both service rates of the CLB and CS bring a significant reduction of the average end-to-end latency. While after the service rate of CLB reaches 200 tasks/second, the rate of latency reduction becomes quite slow. This is because the waiting time in the edge computing platform accounts for a small part of the overall latency.

5.2.3 *Impact of the dynamic offloading strategies*

The offloading strategies play an important role in the performance of VEC systems. The offloading decision making is related to a variety of factors, e.g., VEC service types, wireless channel conditions, vehicle and VEC server processing capabilities and so on. In this subsection, we investigate the impact of the dynamic offloading strategies on the performance of the end-to-end transmission latency. Specifically, four offloading strategies are compared to analyse which strategy suffers from the worst-case end-to-end latency, including Case I (processing all tasks at ego vehicle), Case II (transmitting all tasks to nearby vehicles), Case III (offloading all tasks to VEC servers) and Case IV (delivering 15% of tasks to nearby vehicles, 60% VEC server and 25%

ego vehicles). The results are depicted in Fig. 9. It can be seen that Case IV provides the best performance in terms of the average end-to-end latency. This performance improvement comes from the rich computation resources (including local servers, neighbour vehicles, and MEC servers) that can be jointly used to accomplish the task execution. For the other three offloading strategies, when the arrival rate is smaller than a certain threshold, i.e., 72 tasks/second, Case III suffers from the worst-case end-to-end transmission latency. This is because, for the moderated traffic arrival, the local servers have enough computing resources to handle the tasks generated by the local applications. In this case, offloading all tasks to VEC servers (Case III) would push up the average end-to-end latency. Furthermore, when the arrival rate reaches 120 tasks/second, the prediction accuracy of the end-to-end latency for Case II in Fig. 9 decreases from 94% to 81%. This performance degradation is caused by the saturation phenomenon in VEC systems, where the neighbour vehicles are overloaded by the offloaded tasks. However, it is worth noting that compared with the results reported in the literature [39] [40] [41], which suffer from accuracy loss of 20-35% under saturated working scenarios, the results and accuracy achieved by the proposed model are superior.

6 Conclusion and Future Work

Due to the importance of the communication and computation infrastructure for edge intelligence of IoV applications, this paper developed a novel analytical model for mobile VEC systems with bursty traffic arrival. The analytical model was developed with the aim of capturing the key features of mobile VEC, including vehicle cooperation, V2V/V2X communication links, traffic burstiness and server load sharing. The average end-to-end latency was derived based on the developed analytical model. Comprehensive simulation experiments were conducted and the experimental results demonstrated that the system performance predicted by the analytical model matches well with those obtained by the simulation experiments. Furthermore, the developed model was used as a cost-effective method to analyse the resource allocation strategy at the edge server and offloading policy at vehicles.

The future work can be explored from three important directions. Firstly, in addition to the average end-to-end latency of task offloading strategies in VEC systems that can be captured by the analytical model presented in this paper, it is nontrivial to investigate the worst-case performance metrics caused by the system uncertainty. In this regard, we plan to exploit the Stochastic Network Calculus (SNC) theory to capture the instantaneous and accumulative features of traffic arrivals and service provisioning for investigating the worst-case performance of VEC systems. Secondly, since partition-based task offloading is also a promising computation offloading strategy, we will develop a new analytical model to study the performance of partition-based task offloading in VEC systems. Designing accurate analytical models to investigate the performance of partition-based offloading is extremely challenging. The main difficulty lies in how to capture the behaviour of main-task partitioning and subtask aggregation in the processes of the model abstraction, Markov-chain establishment, and MMPP-based steady-state analysis. Thirdly, offloading tasks among multiple vehicles can raise concerns of security and privacy issues, especially for mission-critical vehicle services. In this context, more research endeavours are required to remove the hinders of the security and privacy issues before implementing the proposed analytical model in the practical IoV systems.

7 Acknowledgements

This work was partly supported by the EU Horizon 2020 INITIATE project under the Grant Agreement No. 101008297 and the National Natural Science Foundation of China under the Grant Agreement No. 61972074.

References

[1] P. Margie, S. Richard, S. David, M. Dinesh, H. Adnan, J. Eva, and M. Colin. World report on road traffic injury prevention. *World Health Organization Geneva*, 2004.

[2] A. Moubayed, A. Shami, P. Heidari, A. Larabi, and R. Brunner. Edge-enabled v2x service placement for intelligent transportation systems. *IEEE Transactions on Mobile Computing*, 20(4):1380–1392, 2020.

[3] X. Huang, R. Yu, S. Xie, and Y. Zhang. Task-container matching game for computation offloading in vehicular edge computing and networks. *IEEE Transactions on Intelligent Transportation Systems*, 2020 (Early Access).

[4] Y. J. Ku, P. H. Chiang, and S. Dey. Real-time qos optimization for vehicular edge computing with off-grid roadside units. *IEEE Transactions on Vehicular Technology*, 69(10):11975–11991, 2020.

[5] Q. Yuan, J. Li, H. Zhou, T. Lin, G. Luo, and X. Shen. A joint service migration and mobility optimization approach for vehicular edge computing. *IEEE Transactions on Vehicular Technology*, 69(8):9041–9052, 2020.

[6] F. Li, X. Song, H. Chen, X. Li, and Y. Wang. Hierarchical routing for vehicular ad hoc networks via reinforcement learning. *IEEE Transactions on Vehicular Technology*, 68(2):1852–1865, 2019.

[7] H. Ye, G. Y. Li, and B. F. Juang. Deep reinforcement learning based resource allocation for v2v communications. *IEEE Transactions on Vehicular Technology*, 68(4):3163–3173, 2019.

[8] Q. Qi, J. Wang, Z. Ma, H. Sun, Y. Cao, L. Zhang, and J. Liao. Knowledge-driven service offloading decision for vehicular edge computing: A deep reinforcement learning approach. *IEEE Transactions on Vehicular Technology*, 68(5):4192–4203, 2019.

[9] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang. Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles. *IEEE Transactions on Vehicular Technology*, 69(4):4298–4311, 2020.

[10] K. Yang, T. Jiang, Y. Shi, and Z. Ding. Federated learning via over-the-air computation. *IEEE Transactions on Wireless Communications*, 19(3):2022–2035, 2020.

[11] W. Miao, G. Min, Y. Wu, H. Huang, Z. Zhao, H. Wang, and C. Luo. Stochastic performance analysis of network function virtualization in future internet. *IEEE Journal on Selected Areas in Communications*, 37(3):613–626, 2019.

[12] G. Min and M. Ould-Khaoua. A performance model for wormhole-switched interconnection networks under self-similar traffic. *IEEE Transactions on Computers*, 53(5):601–613, 2004.

[13] H. Wu and K. Wolter. Stochastic analysis of delayed mobile offloading in heterogeneous networks. *IEEE Transactions on Mobile Computing*, 17(2):461–474, 2018.

[14] H. Lee and J. Lee. Task offloading in heterogeneous mobile cloud computing: Modeling, analysis, and cloudlet deployment. *IEEE Access*, 6:14908–14925, 2018.

[15] L. Liu, S. Chan, G. Han, M. Guizani, and M. Bandai. Performance modeling of representative load sharing schemes for clustered servers in multiaccess edge computing. *IEEE Internet of Things Journal*, 6(3):4880–4888, 2019.

[16] A. Bonadio, F. Chiti, and R. Fantacci. Performance analysis of an edge computing saas system for mobile users. *IEEE Transactions on Vehicular Technology*, 69(2):2049–2057, 2020.

[17] F. Lyu, H. Zhu, N. Cheng, H. Zhou, W. Xu, M. Li, and X. Shen. Characterizing urban vehicle-to-vehicle communications for reliable safety applications. *IEEE Transactions on Intelligent Transportation Systems*, 21(6):2586–2602, 2020.

[18] Q. Yang, T. Chen Y. Liu, and Y. Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology Journal*, 10(2), 2019.

[19] E. Grigoreva, M. Laurer, M. Vilgelm, T. Gehrsitz, and W. Kellerer. Coupled markovian arrival process for automotive machine type communication traffic modeling. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6, 2017.

[20] A. Ali, R. Pinciroli, F. Yan, and E. Smirni. Batch: Machine learning inference serving on serverless platforms with adaptive batching. In *2020 International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, pages 972–986, 2020.

[21] O. Puñal, I. Aktaş, C. Schnelke, G. Abidin, K. Wehrle, and J. Gross. Machine learning-based jamming detection for ieee 802.11: Design and experimental evaluation. In *2014 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pages 1–10, 2014.

[22] Y. Zhu, M. Sheng, J. Li, D. Zhou, and Z. Han. Modeling and performance analysis for satellite data relay networks using two-dimensional markov-modulated process. *IEEE Transactions on Wireless Communications*, 19(6):3894–3907, 2020.

[23] Y. Li and W. Tu. Traffic modelling for iot networks: A survey. In *2020 10th International Conference on Information Communication and Management*, pages 4–9, 2020.

[24] H. Okamura, S. Miyata, and T. Dohi. A markov decision process approach to dynamic power management in a cluster system. *IEEE Access*, 3:3039–3047, 2015.

[25] X. Huang, D. Ye, R. Yu, and L. Shu. Securing parked vehicle assisted fog computing with blockchain and optimal smart contract design. *IEEE/CAA Journal of Automatica Sinica*, 7(2):426–441, 2020.

[26] Z. Zhou, H. Yu, C. Xu, Z. Chang, S. Mumtaz, and J. Rodriguez. Begin: Big data enabled energy-efficient vehicular edge computing. *IEEE Communications Magazine*, 56(12):82–89, 2018.

[27] Shiqiang Wang, Rahul Urgaonkar, Ting He, Kevin Chan, Murtaza Zafer, and Kin K. Leung. Dynamic service placement for mobile micro-clouds with predicted future costs. *IEEE Transactions on Parallel and Distributed Systems*, 28(4):1002–1016, 2017.

[28] X. Xu, Y. Xue, X. Li, L. Qi, and S. Wan. A computation offloading method for edge computing with vehicle-to-everything. *IEEE Access*, 7:131068–131077, 2019.

[29] Z. Wang, Z. Zhong, D. Zhao, and M. Ni. Vehicle-based cloudlet relaying for mobile computation offloading. *IEEE Transactions on Vehicular Technology*, 67(11):11181–11191, 2018.

[30] W. Miao, G. Min, Y. Wu, H. Wang, and J. Hu. Performance modelling and analysis of software-defined networking under bursty multimedia traffic. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 12(5s):1–19, 2016.

[31] H. Lee and D. Cho. Capacity improvement and analysis of voip service in a cognitive radio system. *Queueing Systems Theory and Applications*, 59(4):1646–1651, 2010.

[32] H. Ferng and J. Chang. Departure processes of bmap/g/1 queues. *IEEE Transactions on Vehicular Technology*, 39(2/3):109–135, 2001.

[33] G. Min, Y. Wu, and A. Y. Al-Dubai. Performance modelling and analysis of cognitive mesh networks. *IEEE Transactions on Communications*, 60(6):1474–1478, 2012.

[34] A. Heindi. Decomposition of general queueing networks with mmpp inputs and customer losses. *Performance Evaluation*, 51(2):117–136, 2003.

[35] X. Jin and G. Min. Modelling and analysis of priority queueing systems with multi-class self-similar network traffic: A novel and efficient queue-decomposition approach. *IEEE Transactions on Communications*, 57(5):1444–1452, 2009.

[36] G. Min, J. Hu, and M. E. Woodward. Performance modelling and analysis of the txop scheme in wireless multimedia networks with heterogeneous stations. *IEEE Transactions on Wireless Communications*, 10(12):4130–4139, 2011.

[37] A. W. Malik, K. Bilal, S. U. Malik, Z. Anwar, K. Aziz, D. Kliazovich, N. Ghani, S. U. Khan, and R. Buyya. Cloudnetsim++: A gui based framework for modeling and simulation of data centers in omnet++. *IEEE Transactions on Services Computing*, 10(4):506–519, 2017.

[38] X. Tao, K. Ota, M. Dong, H. Qi, and K. Li. Performance guaranteed computation offloading for mobile-edge cloud computing. *IEEE Wireless Communications Letters*, 6(6):774–777, 2017.

[39] N. Qi, N. I. Miridakis, M. Xiao, T. A. Tsiftsis, R. Yao, and S. Jin. Traffic-aware two-stage queueing communication networks: Queue analysis and energy saving. *IEEE Transactions on Communications*, 68(8):4919–4932, 2020.

[40] F. Mehmeti and T. Spyropoulos. Performance modeling, analysis, and optimization of delayed mobile data offloading for mobile users. *IEEE/ACM Transactions on Networking*, 25(1):550–564, 2017.

[41] W. Ni, J. A. Zhang, Z. Fang, M. Abolhasan, R. P. Liu, and Y. J. Guo. Analysis of finite buffer in two-way relay: A queueing theoretic point of view. *IEEE Transactions on Vehicular Technology*, 67(4):3690–3694, 2018.

Wang Miao received his Ph.D. degree in Computer Science from the University of Exeter, United Kingdom in 2017. He is currently a Postdoctoral Research Associate at the College of Engineering, Mathematics, and Physical Sciences of the University of Exeter. His research interests focus on Vehicle Edge Computing, Unmanned Aerial Networks, Wireless Communication Networks, Software Defined Networking, Network Function Virtualisation, Performance Modelling and Analysis.

Geyong Min is a Professor of High Performance Computing and Networking in the Department of Computer Science within the College of Engineering, Mathematics and Physical Sciences at the University of Exeter, United Kingdom. He received the PhD degree in Computing Science from the University of Glasgow, United Kingdom, in 2003, and the B.Sc. degree in Computer Science from Huazhong University of Science and Technology, China, in 1995. His research interests include Future Internet, Computer Networks, Wireless Communications, Multimedia Systems, Information Security, High Performance Computing, Ubiquitous Computing, Modelling and Performance Engineering.

Xu Zhang received the B.S. degree in communication engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2012 and the Ph.D. degree from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 2017. He is an Associate Professor with the School of Electronic Science and Engineering, Nanjing University, Nanjing, China. His research interests include content delivery networks, network measurement, and cloud computing.

Zhiwei Zhao received his Ph.D. degree from the College of Computer Science, Zhejiang University in 2015. He is currently an associate professor at the College of Computer Science and Engineering in the University of Electronic Science and Technology of China. His research interests focus on wireless computing, heterogeneous wireless networks, protocol design and network coding. He is a member of IEEE and ACM.

Jia Hu a Senior Lecturer in Computer Science at the University of Exeter. He received his Ph.D. degree in Computer Science from the University of Bradford, UK, in 2010, and M.Eng. and B.Eng. degrees in Electronic Engineering from Huazhong University of Science and Technology, China, in 2006 and 2004, respectively. His research interests include edge-cloud computing, resource optimization, applied machine learning, and network security.