Dynamic Resource Allocation Model for Distribution Operations using SDN

Shidrokh Goudarzi, *Member, IEEE,* Mohammad Hossein Anisi, *Senior Member, IEEE,* Hamed Ahmadi, *Senior Member, IEEE,* and Leila Mousavian, *Member, IEEE*

***Abstract*—In vehicular ad-hoc networks, autonomous vehicles generate a large amount of data prior to support in-vehicle applications. So, a big storage and high computation platform is needed. On the other hand, the computation for vehicular networks at the cloud platform requires low latency. Applying edge computation (EC) as a new computing paradigm has potentials to provide computation services while reducing the latency and improving the total utility. We propose a three-tier EC framework to set the elastic calculating processing capacity and dynamic route calculation to suitable edge servers for real-time vehicle monitoring. This framework includes the cloud computation layer, EC layer, and device layer. The formulation of resource allocation approach is similar to an optimization problem. We design a new reinforcement learning (RL) algorithm to deal with resource allocation problem assisted by cloud computation. By integration of EC and software defined networking (SDN), this study provides a new software defined networking edge (SDNE) framework for resource assignment in vehicular networks. The novelty of this work is to design a multi-agent RL-based approach using experience reply. The proposed algorithm stores the users' communication information and the network tracks' state in real-time. The results of simulation with various system factors are presented to display the efficiency of the suggested framework. We present results with a real-world case study.**

***Index Terms*—Resource allocation, Reinforcement learning, Transportation.**

I. Introduction

These days, vehicular communications provide a strong way to connect vehicles with different devices and users in improving transportation. Recently, vehicular cloud networks (VCN) has been proposed as combination of vehicular ad-hoc networks (VANET) and cloud computing to solve some of the exiting challenges of the vehicular networking such as storage, computing, etc. In order to address the challenges for efficient monitoring of vehicles in transit, a vehicular cloud based solution can be adopted. However, the vehicular cloud is a complex setting with soft and hard quality-of-service (QoS) needs on its services [1], [2]. This is due to some devices some devices have communication inside the vehicular cloud. This large number of communications increases the need for adoption of wireless technologies for relatively fast moving vehicles which is challenging. There is intermittent communication links requiring high-speed data transfer in a mobile topology, with dynamically altering demands for services with varying QoS requirements. Hence, dynamic and efficient resource provisioning is vital for successful vehicular clouds [3]. Since the traditional cellular networks have limitations, such as, inefficient,and unscalable packet forwarding, and QoS management, the software defined networking (SDN)-based cellular core was presented in [4]–[6]. To discrete all control functions from the forwarding information function, OpenFlow switches are employed in the SDN cellular core. All base stations (BSs) are controlled by an OpenFlow controller and switched through the OpenFlow protocol [7].

These tasks are difficult without a help from an intelligent internet of vehicles system in terms of edge computation (EC). Since mobile edge computing can employ different cloud resources such as storage and computational resources closer to smart devices/objects, EC has been considered as a highly promising technology for realizing and reaping the advantages of heterogeneous IoT applications [8]. EC inside the vehicular networks can reduce end-to-end delay [9], [10]. The application server is responsible to provide real-time services with constant bounds on the delay. Also, many computational tasks can be performed on small databases which lead to overhead reduction. In real-time vehicle monitoring, the EC is useful to offer a real-time and low latency transmission of the vehicles' information [11], [12]. In the existing literature, considering edge servers for real-time vehicle monitoring for connected vehicles were greatly overlooked. Traditional systems such as data loggers used inefficient passive data collection methods in sharing information among various supply chain parties while in an IoT environment offers real-time vehicle monitoring [13]. In order to manage the mutual interference between the device-to-device links and the cellular links, effective resource allocation mechanisms are needed. The resources assigned to satisfy the alteration in demands for services are adapted by dynamic resource assignment techniques [14].

The work in this paper is motivated by the need to manage the trajectory of vehicles while searching for an efficient channel allocation. We envisage that the suggested framework should be able to support many IoT use cases like mobile delivery and content caching, distributed big data processing. In such vehicular networks, both the bandwidth and computation resources should be efficiently utilized to improve the QoS of vehicular applications. Considering numerous resource-rich vehicles on the road, there is an opportunity for executing data processing and computation offloading on smart vehicles.

Sh. Goudarzi is with Centre for Artificial Intelligent (CAIT), Universiti Kebangsaan Malaysia, 43600 Bangi, Malaysia. e-mail: shidrokh@ukm.edu.my

M.H. Anisi and L. Mousavian are with School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, United Kingdom email:m.anisi@essex.ac.uk;leila.musavian@essex.ac.uk

H. Ahmadi is with Department of Electronic Engineering, University of York, York, YO10 5DD, United Kingdom. email: Hamed.Ahmadi@york.ac.uk.

In this paper, we investigate the integration between cloud edge into IoTs for monitoring the vehicles while in transit. We propose a novel architecture to address dynamic demands for the resources that empowered by EC and reinforcement learning (RL)-based techniques. In this work, by integration of EC and SDN, an innovative software defined edge (SDN-E) framework is proposed for resource assignment in vehicular networks. In our architecture, we provide an integrated mobile edge computation by exploiting the advantages of flexible and high mobility computing resource assignment of edges. In our vehicular edge multi-access networks framework, the distributed computation and collaborative task offloading can be constructed by the vehicles as edge computation resources.

RL is a strong tool in managing scenarios that consider real-world complexity and is useful for designing efficient dynamic resource provisioning heuristics for vehicular clouds [15], [16]. In fact, RL-based techniques can provide an ideal solution for designing a resource assignment model for vehicular clouds in real-time vehicles monitoring during transportation. Also, RL can solve optimization problems for an individual agent [15], but the application of RL to solve system optimization problems in dynamic, real-time vehicles monitoring is an open research problem. Traditionally, researchers have focused on optimization methods to solve real-time vehicles monitoring problems using prior knowledge, but these methods are not viable in dynamic environment that resources change frequently [17].

In the previous works, the QoS of vehicle-to-vehicle (V2V) links only includes the reliability of SINR and the latency constraints and capacity processing to reduce reallocation overhead and delay has not been considered thoroughly.To address these problems, we develop a reinforcement learning technique to learn an optimization policy to improve its optimization objective (cumulative reward) using experience reply dynamically. Currently, some centralized resource assignment mechanisms were established for V2V communications. Since the vehicles' information needs to be reported to the central controller to solve the resource assignment optimization problem, there is a large transmission overhead growing intensely based on the network's size that inhibits these approaches to scale to large networks. Hence, this work focus on decentralized resource assignment approaches with no central controllers to collect the network's information. We exploit deep RL to discover the mapping within the local observations such as local interference levels and channel state information (CSI), and the resource assignment solution.

The proposed architecture offers a dynamic resource assignment method to be reactive to dynamic demands for the services. We focus on resource provisioning in real-time vehicle monitoring. For effective resource provisioning, the demands are precisely examined for deducing the kind of required resources, the amount of each type of required resource and the placement of the resources in the vehicular cloud. The proposed multi-agent RL-based method follows three main objectives: (i) to provide the resources efficiently, (ii) to improve QoS for the end user, and (iii) to reduce delay and overhead for completing requested tasks. Our proposed model can be taken into account as a promising and operative method for managing the network resources such as time, infrastructure and spectrum The proposed model can be used in various types of applications, e.g., civil, military, agricultural applications and environmental remote sensing. Moreover, our model can be utilized for forest fire management, air quality and pollution assessment, coastal ocean observations, precipitation and cloud evaluation, and severe storm monitoring applications. Here, integrating the distributed IoTs and edge resources have the role of moving aggregators for IoT networks [18]. This paper is mainly organized as follows:

- We build a three-tier edge computing framework replacing conventional neural networks for computation and data processing tasks in the edge servers. The architecture uses SDN to simplify the dynamic programming process and manage network connectivity across the data center.
- We integrate distributed IoTs and EC to dynamically allocate the edge servers resources and adjust resources according to the real capacity of the application to reduce reallocation overhead and delay in vehicular clouds.
- We propose a RL-based model for resources allocation based on experience reply that restores experiences into a memory and then samples classes by greedy selection from experience pool for parameters training instead of using consecutive samples.
- We offer a multi-agent RL-based approach to encourage and exploit V2V link cooperation by sharing Q-values among agents to improve network level performance.

Other parts of this study are structured as follows. Correlated works on vehicular networks, cloud computation and IoT in the transportation domain is clarified in Section 2. In Section 3, the core modeling procedure are presented in detail. Section 4 addresses the problem formulation. The reinforcement learning model for resource assignment is explained in Section 5. Performance comparison among the state-of-the-art techniques are presented in Section 6. In the final part, a conclusion is provided in Section 7.

II. RELATED WORK

Edge has a key role in intelligent transportation systems (ITS) to achieve quick response and/or availability [19]. The idea of ITS as the application of the innovative information and communication technologies (ICT) to obtain a decrease in accidents and congestion was summarized in [20] which also showed more security in transport networks was created. In [21], it is highlighted that ITS faces significant challenges for designing and operating the enhanced global supply chain, such as, real-time data-based control, eventually affecting the resiliency and risk. The rising significance of wireless vehicular networks indeed related to the growth and acceptance of mobile wireless communications, in which through progresses in wireless channel modelling methods and the succeeding progress of complex digital transmission approaches, providing high data rate communications is possible while following the severe QoS requirements [22], [23]. According to [24], this is imperative and data overload is experienced by modern companies due to utilizing numerous disparate innovative technologies pursuing a unified functioning picture

for situational awareness. Also, the traditional cloud infrastructure includes geographically dispersed large-scale data centers with thousands of machines publicly available. Integrating this cloud infrastructure and its computation paradigm with VANET increases different vehicular clouds (VCs). Mostly, the vision is to connect the abundant resources within the on-board units (OBUs), road side units (RSUs) and integrating them with the apparently infinite resources of the conventional cloud data centers, [25]–[27], to mention a few.

ITS uses VANET to offer services primarily for vehicle monitoring. However, popular ITS services can improve the efficiency of vehicle monitoring during transportation by offering some infotainment services, like traffic sign detection and recognition, on-demand multimedia video streaming, etc [28], [29]. In a vehicular cloud computing system, vehicles are equipped with OBUs with computational, storage and communication resources. Vehicles communicate with each other as well as with RSUs. Also, each RSU can interact with a cloud infrastructure to offer a variety of modern services with strict QoS requirements [30]. With the development of vehicular technologies, some of the problems such as lack of essential information for monitoring the vehicles while in transit can be eliminated. Autonomous vehicles have abilities in sensing, data processing and communicating with other vehicles [31]. Also, these vehicles can exchange the data with the external environments using various protocols such as Transmission Control Protocol/Internet Protocol (TCP/IP), Next Generation Telematics Protocol (NGTP) and wireless application protocol (WAP) [32]. A new architecture based on vehicular cloud known as ITS-Cloud was proposed to expand vehicle-to-vehicle communication [33].

Access to remote computing resources in a centralized manner is not well suited with massive traffic of distributed edge devices. Therefore, pushing the computation, control and storage to the edge of the network is becoming a basic trend. A deeper look into the "Edge Computation" and we can see the current fast advance in EC (known as "Fog Computation" or Mobile Edge Computation as well in various literature) is aimed to push the storage and computation resources from the distant data center to the network edge to decrease the latency and the backbone burden. Moreover, new improvements in computation (for instance cloud/fog/edge computation) will result in significant impacts on vehicular networks. To make access to a mutual computation resources pool, cloud computation was used very widespread [25]. However, due to the large distance between the end user device and the cloud, low latency usages may not be ensured in cloud computation services in vehicular networks. To state these concerns, studies were performed on EC [26], [27] to organize computation resources closer to vehicles able to competently enhance the QoS for usages requiring concentrated computations and low latency [34]. Integration of SDN and EC was not studied in literature. SDNs offer novel possibilities for designing, securing, and operating data-intensive networks [35]. Nevertheless, realizing these advantages mainly requires the underlying infrastructure support. Recently, the task offloading task for mobile edge computation in a software defined ultra dense network (SD-UDN) to process locally or offload task on edge cloud for minimizing the task period was proposed in [36]. For reducing the expense of deployment and EC services overhead, leveraging the resources of underutilized mobile devices at the edge was proposed in current exertions [37]–[39]. There is not much literature focusing primarily on the MEC and SDN integration. Our study fills the gap in earlier studies on evaluating and designing an innovative architecture using a reinforcement learning technique to allocate resources efficiently for reducing delay and reassignment overhead in vehicular clouds. Such integration can provide a fully software-based framework for any system.

III. SDN-EC FRAMEWORK

Here, the details of the suggested model is explained. The suggested SDN-enabled heterogeneous vehicular network supported by EC is able to offer preferred reliability and data rates in the communication of vehicles to exchange information with everything (V2X) simultaneously. The conceptual architecture of the suggested SDN-EC framework, as shown in Fig.1, depends on the deep convergence of three layers including the cloud computation layer, edge computation layer, and device layer. The key parts in the suggested SDN-EC framework are presented in detail.

A. SDN

In this model, the SDN has been employed for four main reasons: First, using SDN the independent deployment of control, processing entities, and traffic forwarding are possible [40]. Second, the service efficacy of resources is improved by logically centralized control; Third, the network is made more active by the programmability, hence, the appropriate radio access interface is selected by the application to deliver information [41]; Fourth, for vehicle monitoring on the basis of the real-time road conditions, quick-response cloud service is essential. There is no capability in a vehicle, restricted by limited computation resources, in processing a huge volume of uploading traffic data that are captured by the roadside or onboard sensors; Therefore, this kind of tasks should be outsourced to the infrastructure. Various reconfigurable and programmable equipments and factors spanning ground network sections comprise the device layer generally including gateways, vehicles, routers, base stations, and different IoT devices. A hypervisor extracts and virtualizes the device layer resources and the upper-layer computation and storage resources, while ultimately combined into a virtual resource pool. A generalization to the controllers is offered by the hypervisor taking within driving the underlying infrastructures [42]. SDN Controller has two main modules. The first module is for task monitoring. This module collects all vehicles task information. Also, this module advises to compute tasks locally or offload them to edge for processing. Another module is edge server's module. This module collects information of distributed edge servers. This information shows how much memory and CPU is available on the server side and how much the server is loaded.

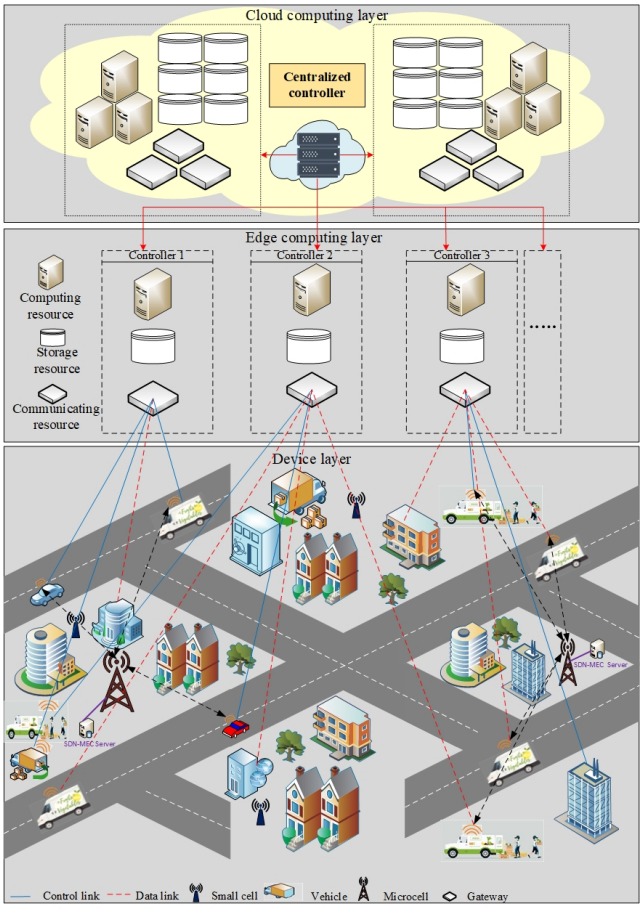

Fig. 1: Architecture of the proposed SDN-EC framework.

B. EC nodes

Multiple virtual EC nodes are created and operated on the similar original infrastructures via an appropriate mapping within virtualized network resources and physical infrastructures. A part of the total virtualized resources is utilized by each EC node for computation tasks and it is individually monitored by the upper-layer controllers. Basic computation functionalities for IoT applications are provided jointly by EC and cloud layers. To control resource management and computation task assignment at various scales, a hierarchical computation architecture is utilized. To process locally, the small-scale delay-sensitive computation tasks in a dispersed mode, the edge calculating layer is more appropriate. The computation, storage, and communication resources are arranged in this layer, close to the users and dispersed through the network edges.

C. SDN-EC controller

Controllers in the lower layer edge are only able to observe and control their individual virtual networks while managed by the upper layer cloud regulators. The resources scale in the cloud computation layer is larger in comparison to the ones in the EC layer; the cloud regulator with central intelligence and global network information manages the resources regularly. To assist the various QoS requirements of IoT applications,

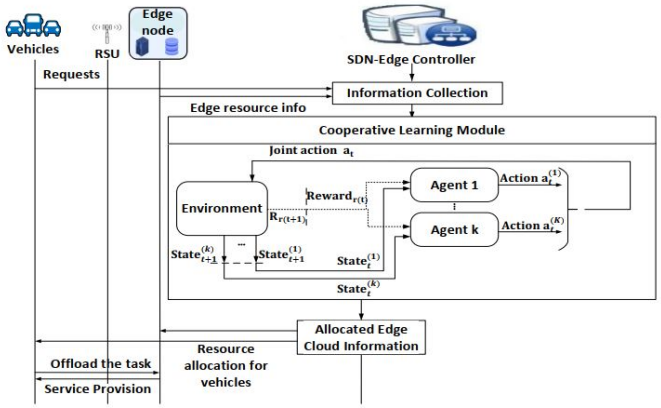

Fig. 2: Operation process in SDN-EC framework.

the upper-layer controllers allocate the resources to each lower-layer controller, then regulates the resource allocation according to traffic demands dynamically.

Each vehicle sends its context to the SDN-EC server's database through BSs that are armed with wireless OpenFlow protocol. The vehicle's parameters are speed, IDs of neighboring vehicles, direction and location that can be derived from GPS signaling. Also, the beacons of neighboring vehicles can be received. SDN controller of SDN-EC are able to find suitable path between vehicles and notifies vehicles to establish path in their routing tables. Also, neighboring SDN-EC servers are able to exchange the stored contents of their databases.

The detailed operation process in SDN-EC framework is described in Fig. 2. In the proposed framework, vehicles are armed with IEEE 802.11p network interface and the cellular network interface. The vehicles describe their requirements and mobility information. Then, the amount of available resources will be updated by SDN-EC controller. In the next step, the cooperative learning module is designed to solve the problem of optimal actions to minimize delay within a completion time limit by dynamically allocate the edge servers resources and adjust resources according to the real capacity of the application to reduce reallocation overhead in vehicular clouds.

IV. PROBLEM FORMULATION

As shown in Fig. 2, the SDN controller is controlling the network. Since the SDN controller can collect all information of the network such as load, latency and speed of processing, it can formulate the optimal strategies for total delay reduction. We assume that each vehicle has some tasks that can be processed locally or offloaded to an EC by a wireless channel. The total request rate that is processed by the EC should be less than a maximum acceptance rate of EC. The controller compares the amount of task computation with the EC computation capability based on delay of vehicle's task. The task is decided to offload to EC or computed locally by vehicle. We consider q_i as the required CPU cycles per bit. In case of local computing, we assume r_i^l is the ratio of task computed locally and s_i is size of computation task, q_i is the

is the required CPU cycles per bit. Thus, the required time for local computation task is calculated as follows:

$$time_i^v = r_i^l s_i \frac{q_i}{l_i^v} \tag{1}$$

Also, the required time for computing a task to an EC is divided into two parts; offloading time and executing time. The data offloading from v_i to EC_j is computed as follows:

$$time_{i,j}^{offloading} = (1 - r_i^l)\frac{s_i}{r_{i,j}^v} \tag{2}$$

As the edge is assumed to have limited computing capability, the computation rate of each EC server is presented as $r_{i,j}^{LC}$. The execution time on the EC is:

$$time_i^{execution} = r_i^{EC} s_i \frac{q_i}{r_{i,j}^{LC}} \tag{3}$$

Because EC has limited capacity, it is not able to execute all tasks of vehicle. The r_j is the data rate from EC in the system to offload the overloaded requests of EC for further execution.The computation time of the overloaded requests can be calculated as follows:

$$time_i^{over.tasks} = (1 - r_i^l - r_i^{EC})\frac{s_i}{r_j} \tag{4}$$

Now, we explain the main optimization objective. The objective is to minimize the total delay of local computation and edge computing modes. The problem can be mathematically written as:

$$\begin{aligned} \min_{r_i^l, r_i^{EC}, r_{i,j}^{LC}} \sum_{i=1}^{N} [& r_i^l s_i \frac{q_i}{l_i^v} + (1 - r_i^l)\frac{s_i}{r_{i,j}^v} + \\ & r_i^{EC} s_i \frac{q_i}{r_{i,j}^{LC}} + (1 - r_i^l - r_i^{EC})\frac{s_i}{r_j}] \end{aligned} \tag{5}$$

subject to

$$\mathrm{C}_1 : \sum_{i \in N} b_i \leq Total_{\mathrm{RBs}}, \quad \forall i \in N \tag{6}$$

$$\mathrm{C}_2 : \sum_{i \in N} r_{i,j}^{LC} \leq r_j^{LC.max}, \quad \forall i \in N \tag{7}$$

$$\mathrm{C}_3 : t_{v_i v_j} \leq d_{v_i v_j}, \quad \forall v_i, v_j \in N; v_j \in M \tag{8}$$

$$\mathrm{C}_4 : 0 \leq r_i^l \leq 1, \quad \forall i \in N \tag{9}$$

$$\mathrm{C}_5 : 0 \leq r_i^{EC} \leq 1 - r_i^l, \quad \forall i \in N \tag{10}$$

where r_i^l is the ratio of task computed locally, r_i^{EC} is the ratio of task computed in the EC, CC_{ij}^{EC} is the computing capability of EC, s_i is size of computation task, q_i is the is the required CPU cycles per bit, and r_j is the data rate from EC in the system to offload the overloaded requests of EC for further execution. In the set of constraints, constraint C_1 indicates the communication resource constraint of wireless networks. Because, the resource blocks (RBs) which used for communication among N vehicles are limited. The b_i shows the quantity of resource blocks which allocated to task T_i and Total_{RBs} is the total number of resource blocks. Constraint C_2 shows that the total edge computing resources which assign to all tasks must be less than the maximum computation capability of edge $r_j^{LC.max}$. Because, EC server has limited processing capacity and cannot offer high computing resources. As the edge is assumed to have limited computing capability, the computation rate of each EC server is presented as $r_{i,j}^{LC}$. Constraint C_3 indicates that delay constraint for task execution time in edge computing mode must be less than the link duration $\mathrm{d}_{v_i v_j}$. In addition, there is a set of vehicles M that are equipped with EC servers. If the vehicle v_i offloads its task to vehicle v_j and its task execution time is less than the $\mathrm{d}_{v_i v_j}$, then vehicle v_i can achieve its computation result. C_4 indicates that the ratio of local task computation should be between 0 and 1. C_5 shows that the ratio of edge task computation should be between 0 and one minus ratio of local task computation.

The conventional method for resource allocation problem is not efficient due to as a large number of communications increases then the QoS requirement increases and this issue leads to the high infeasibility ratio of the conventional algorithm. As a result, it's a NP-hard problem, and it is intractable to find the global optimum solution of the overall network. Also, the computational complexity and signaling overhead increase highly with the number of vehicles, and the optimum solution needs to be delivered from the baseband unit to vehicles within the channel coherence time. This can be achieved using reinforcement learning that can determine the optimal solution to problems by evaluating the results of previous actions. In the next section, we formulate the resource allocation optimization problem as a deep reinforcement learning process.

V. RESOURCE ALLOCATION MODEL

In the proposed model, we deal with the problem from a strengthening learning viewpoint and express any specific optimizing process as a policy. We consider a finite-state Markov decision process with continuous state and action spaces defined by the tuple (S, A, p_0, p, c, γ), S refers to set of states, A refers to set of actions,p_0 is the probability density over initial states; p is the transition possibility density and it is defined as $\mathrm{p} : \mathrm{S} \times \mathrm{A} \times \mathrm{S} \to \mathbb{R}^+$ conditional probability density over successor conditions considering the current action and state, $\mathrm{c} : \mathrm{S} \to \mathbb{R}$ is a function defined for mapping state to cost and γ is the discount factor and it's defined between 0 and 1. The agent obtains $\mathrm{U}(t)$ in state $\mathrm{s}_i(t)$ when action $\mathrm{a}_i(t)$ is performed in time slot t. The objective is to learn a conditional possibility density $\imath^* : \mathrm{S} \times \mathrm{A} \to \mathbb{R}^+$ over actions based on the current state, in a way that estimated cumulative cost is minimized.

$$p^* = \arg^{\min}_p E_{S_0, a_0, S_1, a_1, \ldots . s_T} \left[\sum_{t=0}^{T} \gamma^t c(s_t) \right], \tag{11}$$

The problem of finding the policy with minimum cost is identified as the policy search problem. We use reinforcement learning to learn the policy p. We explain the cost function that penalizes policies exhibiting unwelcome performances over their executing. If the optimization processes converge quickly, they reward. Otherwise, we penalize those that converge slowly. We can calculate the total executive time of a

process. The executive time of a process is equivalent to the last completed task of a process. We show the computation speed of the process at execution time on node k by $(\mathrm{CS})_{\mathrm{k}}$. Let $\mathrm{P_T}$ show the finish time of last completed task of a process on a node and it is explained as follows:

$$P_T = \max \ \{E(t_p) + C_p + n_p \times (CS)_k)\}, \qquad (12)$$

where, $E(t_p)$ defines the initial finish time of a task t_p. The $\mathrm{C_p}$ donates the cost of data transmission in a process. Also, n_p donates the number of executed instructions for a task. For the transferring data d, the communication cost is $\mathrm{Com_{c(ij)}}$ from node in location i to node in location j. If i=j then $\mathrm{Com_{c(ij)=0}}$. The cost of transferred data d with size of s during the time t, with the bandwidth $\mathrm{BW_{ij}}$ is as follows:

$$C_p = \frac{s}{\mathrm{BW_{ij}}} \times \mathrm{Com}_{c(ij)}, \qquad (13)$$

where s denotes the size of the periodically generated V2V payload in bits.

The goal of this study is to reward the processes with quick converge and penalize the processes with slow converge to provide services for users. An agent chooses the transmission power and frequency band level incurring only small interference to all vehicle-to-infrastructure (V2I) links as well as other V2V links and preserve adequate resources to satisfy the need for the latency restrictions. There are mobile edge computation servers (E), base stations $(\mathrm{BS})$and requested contents (RC) that are manage by a centralized controller which provide services for vehicles and flexible resource control by separating the information and control functions as well as the innovative concept of fundamental infrastructures. Since the downlink channel conditions of base stations and computation abilities of mobile edge computation servers are changing dynamically, the channels of vehicles and their connected BSs/RSUs are modeled as finite-state Markov channels and a large amount of system states should be analysed. The best decision should be made to allocate the resources to multiple vehicles according to the recent state of the system.

A. Deep Q-learning

The deep Q-Network can solve this problem and manage the system effectively. Deep Q-learning has the ability to receive complex high dimensional information as input data and assigning a best action for each input statistics in a given state x. The proposed model, firstly, all states from per B, E and RC should be collected, and all input data are sent to the deep Q-Network.Then, a feedback of the best policy p^* for assigning the demanded resources for a vehicle is obtained.

The agent using the neural network (NN) to characterize Q function is known Q-network that is defined as $Q(x, a; \theta)$. The factor θ represents the neural network weights, a represents the action in a given state x and the Q-network is qualified through updating θ at each iteration to estimate the real values of Q. By determining θ, Q-values, $Q(x, a)$, will be the outputs of the deep neural networks. The deep neural networks can represent sophisticate mappings between the channel information and the desirable output in terms of numerous training data that will be utilized for determining Q-values. Furthermore, as a result of the weight functions particularity in all benchmarks, we present a new scoring function.The Q-network updates its weights, θ, at each iteration for minimizing the following function as a loss function (L) extracted from the same Q-network with old weights on a data set D_s,

$$L(\theta) = \sum_{Q \in D_s} \left(Q(x, a, \theta) - [r + \gamma \max_{a'} Q(x', a', \theta) \mid x, a] \right)^2 \qquad (14)$$

The stochastic gradient descent is applied to take an action a in state x for moving to x' and achieve reward r on each new example (x,a,x',r). The scoring function is defined to compare the weights of vehicles. The weights are corresponded to the vehicles' priority. When the quantity of energy consumed and number of fulfilled vehicles and downloaded bits is achieved, the RSU provide the channel access.

In Q-network, the experience of agent (indeed the deep Q-learning) stored in a memory at each time slot.The agents cooperate with each other in order to learn how to share their sensory data and act as the scout for each other. Actually, utilizing a multi-agent RL network makes possible for the agents to communicate and share their capabilities and learn from one another. Episode sharing can be utilized for communicating the state, action, and reward triples within the reinforcement learners. The multi-agent RL network consisting of numerous novel agents, cooperation and information sharing among these agents can decrease search time for the delay minimizing solution.

Also, the parameters of Q-network are updated with specimens from the memory at each time instant. A Q-value is a state-action pair function in deep reinforcement learning procedure which returns a real value. Exploration is required to make sure each action in each state is sampled. An agent must select the actions that it has found make maximum rewards given actions it has tried in the past. Though, for learning those actions, it must to do exploration to try fresh actions that it has not achieved before. At some time, the agent must use what it has learned (exploitation) in order to get rewards.

A common method for exploration is to select random actions with minor probability, this is known as ε-greedy exploration. We defined a ε-greedy policy to make balance in the exploitation and exploration. In other words, this policy is used to make balance between the reward maximization on the basis of the knowledge already identified by trying new actions to find unknown knowledge.The greedy procedure to improve the initial solution.

B. Deep Q-network algorithm in VANET

The Q-network training algorithm is demonstrated in Algorithm 1. To achieve the states, actions, and reward and best policy functions need to be described. The proposed algorithm can be implemented in the SDN-EC server's SDN controller.

State: Each base station(BS) as an agent should collect the status of EC server. The agent can determine the optimal amount of spectrum resource based on collected information such as average of SINR. Then, all information should be assembled into a constructed system. After that, the BS should send the recorded states to an agent and receives feedback to allocate resources based on proper strategy for a vehicle. For a base station $b \in \{1, 2, 3, ..., B\}$, a mobile edge computing server $e \in \{1, 2, 3, ..., E\}$, for vehicle v_s at time instant $t \in \{0, 1, 2, ..., T-1\}$, the environment state is defined as,

$$s_t = \{y, n, p, \mathrm{SINR}_c, \mathrm{SINR}_e, \mathrm{SINR}_b\}, \tag{15}$$

where y is the triggered tier transmitter, n shows the accessible resources, p represents the transmission power of the triggered tier transmitter, $\mathrm{SINR_c}$,$\mathrm{SINR_e}$, $\mathrm{SINR_b}$ are the SINR measured at a triggered receiver in the centre band, edge band and macro receiver BS, respectively. The SINRs of the V2I link and the V2V link over the sub-band are expressed as $\mathrm{SINR_{V2I}}$ and $\mathrm{SINR_{V2V}}$.

The capacities of the V2I links and the V2V links over the sub-bands are then obtained as,

$$C_{V2I} = B\log(1 + \mathrm{SINR}_{V2I}), \tag{16}$$

and

$$C_{V2V} = B\log(1 + \mathrm{SINR}_{V2V}), \tag{17}$$

where B is the bandwidth of each spectrum sub-band. The reward is defined to maximize the total processing capacity of all nth V2I links, defined as $\sum_n C_n[n]$.

Action: The action for a specific vehicle v_s at time instant t is determined as follows:

$$a_{v_s} = \{n, p_c, p_e, m\}, \tag{18}$$

where m is the transmission modulation order, p_c and p_e show the chosen transmission power of the triggered transmitters positioned at the centre band and edge band, respectively.The list of actions performed by the agent have a finite impact on the delay.

Reward: In the proposed model, the problem of resource allocation optimizing has formulated as a deep reinforcement learning procedure. The optimization problem is to achieve the minimal total cost. The objective of reinforcement learning procedure is to obtain the highest reward which is inversely associated with total cost and it's defined as $\frac{TC_l - TC_{(s,a)}}{TC_l}$.When the cost of local calculation is shown by TC_l and the total cost of system is presented as $TC_{(s,a)}$. In each stage, an agent calculates and store the $Q(s,a)$ in the Q-table. In the proposed model, each base station BS should pay for employing the spectrum that is determined as δ_b per Hz. Also, the computation fee should pay for computation task to be executed on the mobile edge computation server. This value at eth mobile edge computation server is explained as τ_e per Joule. Furthermore, the charge for each vehicle v_s to access to an available network for computing a task at base station BS is defined as ϕ_{v_s} per bps. Also, required number of CPU cycles for completing each task for specific vehicle is defined as q_{v_s}. In our framework, the reward function consists of three parts, namely, reliability of SINR, the capacity of the V2I and V2V links, and the delay condition. The latency condition is represented as a penalty. The action defines if the reward will be obtained. Our objective is to meet the delay and the goal of RL is to obtain the maximum rewards. We measure the delay after each action. We define a reward function based on delay in executing a task and can be defined as:

$$R_t = \frac{1}{delay_{minimum}^{current}} \tag{19}$$

where $delay_{minimum}^{current}$is the minimum delay in task execution in current state. Then, the reward for a particular vehicle v_s is defined as:

$$\begin{aligned} R_{v_s} = \sum_{b=1}^{B} R_{v_{s,b}}(t) + \sum_{e=1}^{E} R_{v_{s,e}}(t) - \gamma_p(T_0 - T_r) = \\ \sum_{b=1}^{B} a_{v_{s,b}}(t)(\phi_{v_s} b_{v_{s,e}} - \delta_b b_{v_{s,e}}) + \\ \sum_{e=1}^{E} a_{v_{s,e}}(t)(\phi_{v_s} r_{v_{s,e}} - \tau_e q_{v_s} e_{v_{s,e}}) \end{aligned} \tag{20}$$

where γ_p is the penalty weight, $\mathrm{T_0}$ is the maximum tolerable latency and $\mathrm{T_r}$ the remaining time to meet the latency constraints. The $(\mathrm{T_0} - \mathrm{T_r})$ denotes the transmission time and the penalty grows as the transmission time increases. The system reward is defined as the maximum efficiency (E) from a v_s at time t in equation (4). The Deep Q-network aims at finding an ideal policy for maximizing the efficiency of model, and the cumulative efficiency is calculated as,

$$R_{v_s}^{\mathrm{long}} = \max E \sum_{t=0}^{T-1} \epsilon^t R_{v_s}(t) \tag{21}$$

where ϵ^t reaches 0 when t is large enough. To terminate the process, a threshold can be set. The proposed mechanism uses the gathered state data to assign resources and to control interference. Since the utilized learning method is cooperative, it requires distribution of the state-action data from the consistent agent Q-table to the neighbours entirely and receive their state-action data. The exploitation stage is supported by state-action data, where the action is chosen based on the highest value of Q, recursively updated as follows,

$$\begin{aligned} Q_{v_s}(s_{v_s}, a_{v_s}) = (1-\alpha) Q_{v_s}(s_{v_s}, a_{v_s}) \\ +\alpha \left(r_{v_s}(s_{v_s}, a_{v_s}) + \gamma_{l \in A_{v_s}}^{\max} Q(s_{v_s *}, l) \right) \end{aligned} \tag{22}$$

where $0 < \alpha \leq 1$ is the learning rate, s_{v_s} is the present condition of the specific vehicle y, and $s_{v_s *}$ represents the preceding state of the specific vehicle v_s.

The process begins with gathering the network state data. The state information exchange process then is engaged where

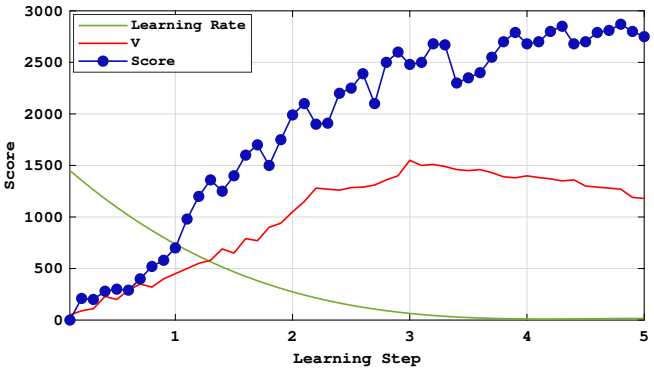

Fig. 3: Deep Q-network with a decreasing learning rate.

the row of its Q-table is shared by each vehicle corresponding to its present state and the optimal action. Simultaneously, it obtains the present state and all other vehicles' optimal actions. This aids the vehicle to control its joint action with the highest cumulative Q-value in the exploitation stage as follows,

$$a_{v_s} = \arg\max \sum Q\left(s_{v_s}, a_{v_s}\right) \tag{23}$$

Regarding this observation, through choosing the action on the basis of (12), the global value of Q will be maximized.

A further improvement is needed in achieving optimal learning rate and exploration rate. We start a learning rate of 0.005 and decreases exponentially. We use a rule at each epoch as follows:

$$\alpha_{n+1} = 0.98\,\alpha_n \tag{24}$$

It should be noted that the average Q-value function V reduced when the learning rate is lowered. We manage to improve the score of the Q-value function when the learning rate is decreased. The learning rates used in our work are tested between $0 < \alpha \leq 1$ and we set $\alpha = 0.07$ for rewarded solution and $\alpha = 0.01$ for the punished solution. At the beginning of process, the score is very low. While the time passes, the score is adding but has fluctuation. The reason for this fluctuation is that the operator is still learning the system's parameters and try find the best parameters. Also, it reduces errors and instabilities when decreases exploration. The Fig. 3 shows that higher rate helps the algorithm to get out of the local optimum. Fig. 4 shows that the exploration rate can be adopted during the training process in the ε-greedy action selection until the agent was able to get out of the local optimum.

Algorithm 1: Deep Q-network algorithm in VANET

Step 1: Initialization
Step 1.1: Initialize the experience replay memory.
Step 1.2: Initialize the main Q-network with weights θ.
Step 1.3 Initialize the target deep Q-network with weights $\theta^{-} = \theta$.
Step 2. **for** *episode* $b = 1, ..., B$ **do**
  Step 2.1 Receive the initial observation state s_1.
  Step 2.2 **for** $t = 1, 2, .., T$ **do**
    for $Train : 1, 2, 3, ..., R$ **do**
      Step 2.3 Select a random probability p.
      Step 2.4 Select a_t as,
      if $p \leq \varepsilon$ **then**
        select a random action a_t
      else
        $a_t = arg\ max_a\ Q(x, a; \theta)$
      Step 2.5 Execute action a_t in the system, gain the reward r_t,and the subsequent statement s_{t+1} to the next state x_{t+1}.
      Step 2.6 Store the experience (x_t, a_t, r_t, x_{t+1}) into the experience replay memory.
      Step 2.7 Get a batch of U samples (x_i, a_i, r_i, x_{i+1}) from the reply memory.
      Step 2.8 Compute the target y_t^{-} from the target network, $y_t^{-} = r_i + \varepsilon\, Q\left(x_{i+1}, arg\ max_{a'}\ Q\left(x_{i+1}, a'; \theta^{-}\right)\right)$
      Step 2.9 Minimize the loss L(θ) for updating the main deep Q-network $L(\theta)$, $L(\theta) = \frac{1}{U} \sum_i \left(y_i^{-} - Q(x_i, a_i; \theta)\right)^2$
      Step 2.10 do a gradient descent step on $L(\theta)$
      Step 2.11 Train = Train + 1, If Train $\leq R$, go to Step 2.3.
      Step 2.12 $t = t + 1$, if $t \leq T$, go to Step 2.3.
        `// T represents the pre-set number of epochs in an episode.`
      Step 3 Let $k = k + 1$. If $k \leq K$, go to Step 2.
        `// K indicates the pre-set maximum number of episodes.`
      Step 4 Return the value parameters θ in the Q-network.

C. Complexity Analysis

In this section, we analysis the computational complexity of proposed algorithm. The complexity of a RL-based algorithm mainly depends on the state space size, the structure of states and the primary knowledge of the agents [43]. If prior knowledge is available to an agent, the search time can be reduced significantly. In our algorithm, by sharing Q-values among agents, a prior knowledge is provided for agents to reduce their search time. Accordingly, the sharing of Q-tables by agents, which are corresponding to its present state and the optimal action can reduce the overhead. Also, to perform accurate sharing, we used weighted functions to achieve a lower complexity. The complexity of this operation mainly depends on the number of the available computation node. In a single agent network [44] that uses the ϵ-greedy policy with

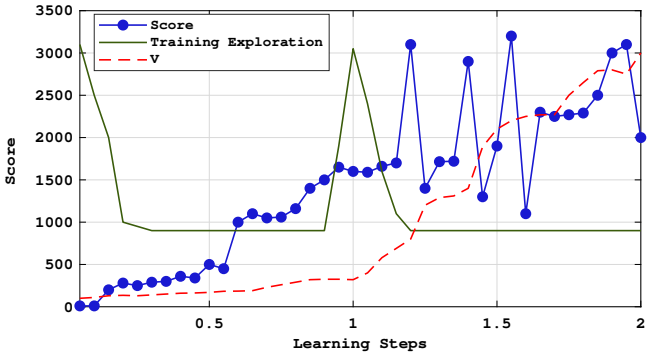

Fig. 4: Deep Q-network during the training process.

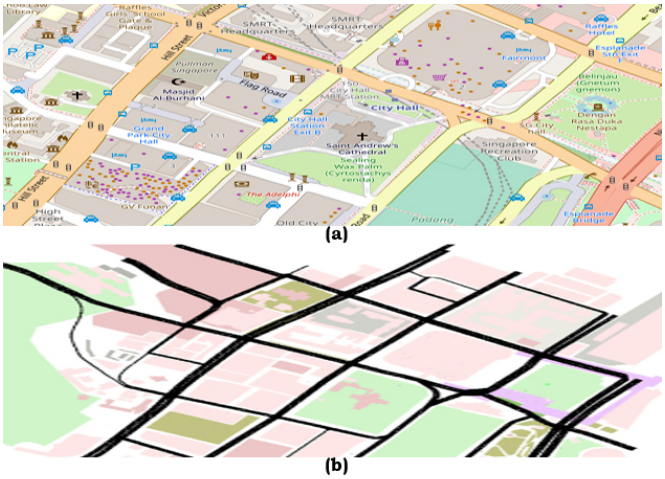

Fig. 5: Singapore Map in OpenStreetMap(a) and SUMO(b).

fixed ϵ in the size of the state-space $|S|$, the needed time for convergence is bounded by $O\left(|S|\, log\, |S|\, \frac{log(\frac{1}{\epsilon})}{\epsilon^2}\right)$, when the limit number of iterations has been reached. Our proposed RL-based algorithm for resources allocation restores experiences into a memory and then samples classes by greedy selection from experience pool for parameters training instead of using consecutive samples. Thus, the use of the greedy selection with threshold leads to significant reduction of the system complexity. Also, in the Q-network in our work, the activation function uses relu, and after the model is trained. In our multi-agent RL-based algorithm, with the number of actions $N_{actions}$ in each iteration, the computational complexity is $O\left(|S|\, N_{actions}\, \frac{log(\frac{1}{\epsilon})}{\epsilon^2}\right)$ and is linear in state-space size. On the other hand, the complexity of exhaustive search for optimal solution is $O\left(N_{actions}^{K}\right)$, where K is number of the available computation node.

D. A Case Study of Distribution of Food Products

In this section, we develop a case study in order to show the effectiveness of the proposed framework. The efficient distribution of food products, such as fruits, is important for reducing the product spoilage rate during transportation. One of the vital components of this system is the possibility of monitoring vehicles in the transportation process. There are public concerns with monitoring of fresh products during in transit and the spread of microbiological hazards as a potential health issue. One of the challenges is determining how to monitor and control the vehicles through the transportation process. In traditional marketing channels, obtaining information on supermarkets of the fresh produce, such as, the quantities of fruits and vegetables, and delivery time, are difficult tasks [45]. Limitations occurring within the fresh products involve lasting the trucks idle over a long time and constrained abilities of monitoring of vehicles in transit. Efficient monitoring of various supply chain elements can prevent delivery of spoiled material to customers. In fresh distribution of products, efficient communication between vehicles can reduce delay and congestion. The developments in the internet of things (IoT) and cloud computing have offered potential solutions to solve the problems faced by the growing transportation issues such as in-transit fresh produce [46].

We investigate the integration between cloud edge into IoTs for monitoring the fresh produce while in transit. We utilized Simulation of Urban Mobility (SUMO) [47] and MATLAB as two main platforms. SUMO is an open source microscopic traffic simulator. A reduced area of the downtown area of Singapore is downloaded from OpenStreetMap [48], as shown in Fig. 5. There is a real traffic scenario with a large number of vehicles in a reduced area. To perform the experiment, the MATLAB software is employed as interface with TraCI [49]. The geographical areas in OSM were exported to individual map (.osm) files. The irrelevant data such as parks and pedestrian walkways were removed from maps by Java Open Street Map (JOSM). Then, the realistic road scenario as working map imported from OpenStreetMap into the SUMO traffic simulator.All base stations and vehicles are accidentally dispersed in the MBS's area. During the simulations, it's supposed that there are four base stations, six mobile edge computation servers while normalizing the bandwidth of each base station.

The Markov model is considered for wireless channels within the vehicles and base stations. It is considered that the channels between vehicles and BSs/RSUs are modeled as finite-state Markov channels and they are real time-varying channels. The vehicles are dropped in the lane randomly according to the spatial Poisson process and each plan for communicating with the three adjacent vehicles. Therefore, the number of V2V links is three times of the number of vehicles. The deep Q-network in our work is a completely connected neural network with 5 layers and 3 hidden layers. In the 3 concealed layers, the number of neurons is 500, 250, and 120, respectively. We also use ϵ-greedy policy for balancing the exploitation and exploration and adaptive moment estimation technique (Adam) for training [50]. In addition, we fix the V2V payload packet size s in the training stage to be of 2×1060 bytes, but vary the sizes in the testing stage to verify robustness of the proposed method. The simulation parameters for all models are given in Table I.

In this model, each V2V link is considered as an agent and spectrum are chosen in terms of the transmission power and spectrum are chosen in terms of the channel circumstances and data shared from the neighbors at each time slot. Also, the received SNR is an appropriate factor for reflecting the

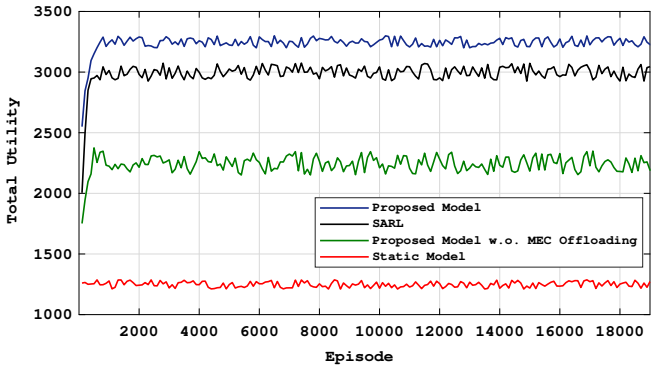

Fig. 6: Convergence performance of various patterns.

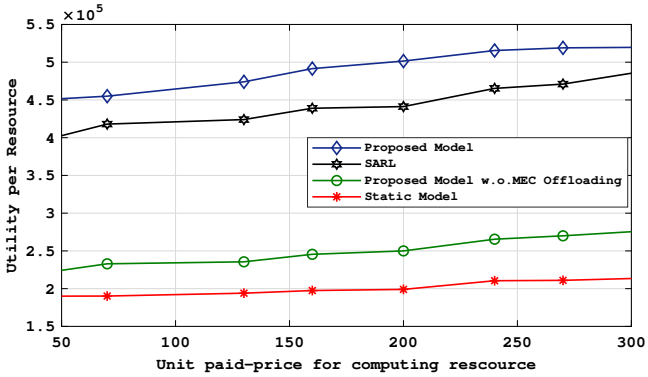

Fig. 7: The utility per resource for computing resources.

quality of a channel. A random variable $r^b_{v_s}$ is defined to model the received SNR of the wireless channel linking vehicle v_s and BS b.QoS queues are configured by the SDN controller at the ports of network edge switches. The Floodlight SDN controller [51] has been extended as a resource monitoring module to calculate the delay and traffic level in each link periodically. Also, OVSDB queue management module [52] has been employed to enable dynamic queue creation.

When implementing the queues, utilizing Open Flow queuing action, particular traffic flows can be routed over particular queues to satisfy the associated bandwidth needs. In the proposed architecture, the centralized SDN controller plans time slots in time-division multiple access (TDMA) for RSUs to periodically communicate with the controller. Also, RSUs assign time slots to vehicles in their transmission range for network information collection [53].

VI. RESULTS AND DISCUSSION

For performance comparison, three models namely static model, the single agent reinforcement learning (SARL) algorithm [54] and proposed model without mobile edge computing servers are used for comparison with the proposed model. The state of system is supposed to be static, not changing dynamically. The queues are configured in two different ways. For static configuration, the queues are configured at a fixed number of queues with some maximum bitrates, and for dynamic configuration, the queues are on-demand and allocating one or more flows in each queue. Edge computing offloading is not considered in the model without mobile edge computing servers, and vehicles can only do the calculation tasks occasionally. In SARL model, only one link operates as an agent updates its action at each moment and makes its own allocation decisions optimally to minimize interference under latency constraints.

Based on Fig. 6, we can find that at the start of the learning procedure, the overall effectiveness of various scenarios in the suggested system is very low. Increasing the number of the episodes, there will be an increment in the overall utility until reaching a fairly stable value (almost 6500 in the recommended scheme), as shown in Fig. 6. This indicates the convergence performance of the suggested system. It is also observed that not considering the networking and computing together, the overall utility of other scenarios is less.

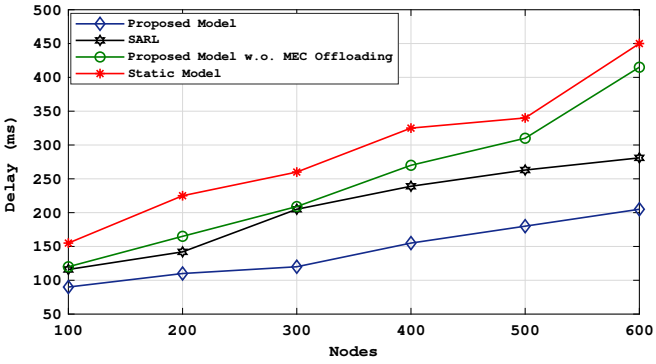

Fig. 8: The normal delay in opposition to the number of nodes.

Fig. 7 indicates the expected utility per resource against the unit paid-price to use computing resources ϕ_{v_s} in different methods. The utility from computing resources extends by the increase of the unit paid-price to use computation resources that leads to the increase of the utility. Our proposed scheme has the best utility because it can make superior strategies.

Fig. 8 indicates the association between the average delay performance and the number of nodes. It is indicated that regarding the number of users, the average delay performances of all models is increased monotonically. The reason is the increment in the time over the task uploading, processing and queuing, as more users attempt to connect to the similar offload computation tasks and edge nodes. Nonetheless, the achieved results indicated that through a considerable margin by effective use of the integrated resources, the average delay can be decreased. By reaching the number of users to 600, the average delay obtained by the suggested scheme is 51% less compared to the baseline model.

Fig. 9 shows the success connections against the number of users by calculating the cumulative distribution function (CDF). The CDF is an increasing function against the time but, we derived the CDF against the number of users, so that the CDF is decreasing. Here, the success connections indicate the ratio of users with accepted computing demands positively by mobile edge computing nodes. It is stated that increasing the number of users to 1000, as a result of the access and backhaul-associated limitations, only 14% of service demands can be effectively connected in the standard structure. In con-

TABLE I: PARAMETER VALUES

Parameter	Value
BS antenna height	24m
Vehicle speed	35 km/h
BS receiver noise figure	5 dB
Vehicle antenna gain	4 dBi
V2I transmit power	24 dBm
V2V transmit power	[24, 10,-100] dBm
δ_b	2 units/MHz (Units paid for wireless spectrum usage)
ϕ_{v_s}	10 unit/Mbps (Unit for unit paid-price of using computing resources)
SINR threshold	1 dB
Latency constraints for V2V links T_0	100 ms
Carrier frequency	2 GHz
τ_e	100 units/J (Units of energy consumed by EC servers)
q_{v_s}	100 M cycles (Cycles of CPU for each task)
Learning Rate (α)	dynamic
Exploration Rate (ϵ)	dynamic
Number of Resources blocks	20
Path loss	$PL = 127 + 37\,log(d)$, d = distance between underlay transmitter and UE
V2V links	4
V2I links	4

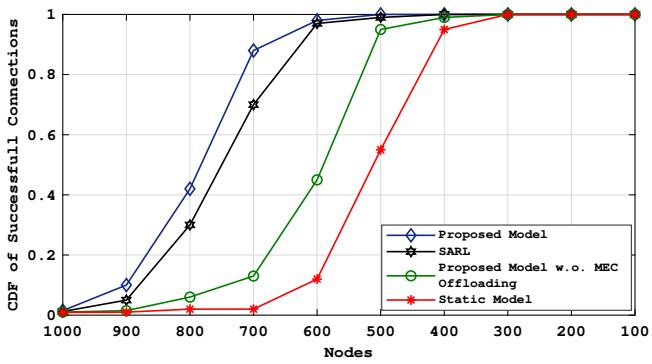

Fig. 9: CDF of effective service connections against the number of users.

trast, the suggested scheme overtakes the baseline system by 89%, since QoS performances of mobile edge users able to be successfully improved by the integrated resources with large-scale coverage while offering more consistent communication links compared to the traditional static connections.

VII. CONCLUSIONS

In this work, new software defined edge framework for resource assignment in vehicular networks has been proposed. According to the programmable control standard created by SDN, an integrated framework is able to optimally allocate networking and computing resources. The resource assignment approach is formulated as an optimization problem. In the proposed model, integrated edge computing framework, each agent can learn how to satisfy the vehicle to vehicle communications while the total delay can be significantly reduced. Our simulation results indicate that the our resource assignment approach serves all vehicles in a suitable manner by sharing the learning experiences among all agents to compute vehicles tasks. The analysis of a case study has demonstrated the effectiveness of the proposed framework against compared models. Studies on air-ground integrated mobile edge computation framework are proposed as the futures work, through using the advantages of high flexible and mobility computation resource assignment of vehicles and UAVs.

ACKNOWLEDGMENT

The authors wish to thank the Ministry of Higher Education Malaysia and Universiti Kebangsaan Malaysia (grant ID: PP-FTSM-2020) for supporting and funding this work.

REFERENCES

[1] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani, "Software-defined networking for rsu clouds in support of the internet of vehicles," *IEEE Internet of Things journal*, vol. 2, no. 2, pp. 133–144, 2014.

[2] R. Du, P. Santi, M. Xiao, A. V. Vasilakos, and C. Fischione, "The sensable city: A survey on the deployment and management for smart city monitoring," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1533–1560, 2018.

[3] M. R. Rahimi, N. Venkatasubramanian, S. Mehrotra, and A. V. Vasilakos, "On optimal and fair service allocation in mobile cloud computing," *IEEE Transactions on Cloud Computing*, vol. 6, no. 3, pp. 815–828, 2015.

[4] A. Brogi and S. Forti, "Qos-aware deployment of iot applications through the fog," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1185–1192, 2017.

[5] H. Cai, B. Xu, L. Jiang, and A. V. Vasilakos, "Iot-based big data storage systems in cloud computing: perspectives and challenges," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 75–87, 2016.

[6] S. Bera, S. Misra, and A. V. Vasilakos, "Software-defined networking for internet of things: A survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1994–2008, 2017.

[7] A. Lara, A. Kolasani, and B. Ramamurthy, "Network innovation using openflow: A survey," *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 493–512, 2013.

[8] X. Chen, A. Li, X. Zeng, W. Guo, and G. Huang, "Runtime model based approach to IoT application development," *Frontiers of Computer Science*, vol. 9, no. 4, pp. 540–553, 2015.

[9] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through vm migration and transmission power control," *IEEE Transactions on Computers*, vol. 66, no. 5, pp. 810–819, 2016.

[10] Y. Liu, X. Weng, J. Wan, X. Yue, H. Song, and A. V. Vasilakos, "Exploring data validity in transportation systems for smart cities," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 26–33, 2017.

[11] G. Sun, Z. Xu, H. Yu, X. Chen, V. Chang, and A. V. Vasilakos, "Low-latency and resource-efficient service function chaining orchestration in network function virtualization," *IEEE Internet of Things Journal*, 2019.

[12] G. Sun, R. Zhou, J. Sun, H. Yu, and A. V. Vasilakos, "Energy-Efficient Provisioning for Service Function Chains to Support Delay-Sensitive Applications in Network Function Virtualization," *IEEE Internet of Things Journal*, 2020.

[13] M. Huang, A. Liu, N. N. Xiong, T. Wang, and A. V. Vasilakos, "An Effective Service-Oriented Networking Management Architecture for 5G-Enabled Internet of Things," *Computer Networks*, p. 107208, 2020.

[14] M. Verma, G. Gangadharan, N. C. Narendra, R. Vadlamani, V. Inamdar, L. Ramachandran, R. N. Calheiros, and R. Buyya, "Dynamic resource demand prediction and allocation in multi-tenant service clouds," *Concurrency and Computation: Practice and Experience*, vol. 28, no. 17, pp. 4429–4442, 2016.

[15] J. Wang, J. Hu, G. Min, W. Zhan, Q. Ni, and N. Georgalas, "Computation offloading in multi-access edge computing using a deep sequential model based on reinforcement learning," *IEEE Communications Magazine*, vol. 57, no. 5, pp. 64–69, 2019.

[16] M. Qiu, W. Dai, and A. V. Vasilakos, "Loop Parallelism Maximization for Multimedia Data Processing in Mobile Vehicular Clouds," *IEEE Transactions on Cloud Computing*, vol. 7, no. 1, pp. 250–258, 2016.

[17] Y. Sun, M. Peng, and S. Mao, "Deep reinforcement learning-based mode selection and resource management for green fog radio access networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1960–1971, 2018.

[18] S. Goudarzi, N. Kama, M. H. Anisi, S. Zeadally, and S. Mumtaz, "Data collection using unmanned aerial vehicles for internet of things platforms," *Computers & Electrical Engineering*, vol. 75, pp. 1–15, 2019.

[19] C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos, "Fog computing for sustainable smart cities: A survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, pp. 1–43, 2017.

[20] A. E. C. Mondragon, E. S. C. Mondragon, C. E. C. Mondragon, and F. Mung'au, "Estimating the performance of intelligent transport systems wireless services for multimodal logistics applications," *Expert Systems with Applications*, vol. 39, no. 4, pp. 3939–3949, 2012.

[21] G. R. Zomer and N. Anten, "Vision of the crucial role of its for efficient and green european logistics," 2008.

[22] H.-N. Dai, R. C.-W. Wong, H. Wang, Z. Zheng, and A. V. Vasilakos, "Big data analytics for large-scale wireless networks: Challenges and opportunities," *ACM Computing Surveys (CSUR)*, vol. 52, no. 5, pp. 1–36, 2019.

[23] E. Ahmed, I. Yaqoob, I. A. T. Hashem, I. Khan, A. I. A. Ahmed, M. Imran, and A. V. Vasilakos, "The role of big data analytics in Internet of Things," *Computer Networks*, vol. 129, pp. 459–471, 2017.

[24] A. E. C. Mondragon, C. S. Lalwani, E. S. C. Mondragon, C. E. C. Mondragon, and K. S. Pawar, "Intelligent transport systems in multimodal logistics: A case of role and contribution through wireless vehicular networks in a sea port location," *International Journal of Production Economics*, vol. 137, no. 1, pp. 165–175, 2012.

[25] H. Zhang, Q. Zhang, and X. Du, "Toward vehicle-assisted cloud computing for smartphones," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 12, pp. 5610–5618, 2015.

[26] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal *et al.*, "Mobile-edge computing introductory technical white paper," *White paper, mobile-edge computing (MEC) industry initiative*, pp. 1089–7801, 2014.

[27] C. Liang, Y. He, F. R. Yu, and N. Zhao, "Energy-efficient resource allocation in software-defined mobile networks with mobile edge computing and caching," in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2017, pp. 121–126.

[28] Z. Huang, Y. Yu, J. Gu, and H. Liu, "An efficient method for traffic sign recognition based on extreme learning machine," *IEEE transactions on cybernetics*, vol. 47, no. 4, pp. 920–933, 2016.

[29] J. Wan, J. Liu, Z. Shao, A. V. Vasilakos, M. Imran, and K. Zhou, "Mobile crowd sensing for traffic prediction in internet of vehicles," *Sensors*, vol. 16, no. 1, p. 88, 2016.

[30] C.-M. Huang, M.-S. Chiang, D.-T. Dao, H.-M. Pai, S. Xu, and H. Zhou, "Vehicle-to-infrastructure (v2i) offloading from cellular network to 802.11 p wi-fi network based on the software-defined network (sdn) architecture," *Vehicular Communications*, vol. 9, pp. 288–300, 2017.

[31] S. Misra, P. V. Krishna, V. Saritha, H. Agarwal, A. V. Vasilakos, and M. S. Obaidat, "Learning automata-based fault-tolerant system for dynamic autonomous unmanned vehicular networks," *IEEE Systems Journal*, vol. 11, no. 4, pp. 2929–2938, 2015.

[32] A. Iwai and M. Aoyama, "Automotive cloud service systems based on service-oriented architecture and its evaluation," in *2011 IEEE 4th International Conference on Cloud Computing*. IEEE, 2011, pp. 638–645.

[33] S. Bitam and A. Mellouk, "Its-cloud: Cloud computing for intelligent transportation system," in *2012 IEEE global communications conference (GLOBECOM)*. IEEE, 2012, pp. 2054–2059.

[34] N. Kumar, S. Zeadally, and J. J. Rodrigues, "Vehicular delay-tolerant networks for smart grid data management using mobile edge computing," *IEEE Communications Magazine*, vol. 54, no. 10, pp. 60–66, 2016.

[35] A. A. Abbasi, M. A. Al-qaness, M. A. Elaziz, A. Hawbani, A. A. Ewees, S. Javed, and S. Kim, "Phantom: Towards Vendor-Agnostic Resource Consolidation in Cloud Environments," *Electronics*, vol. 8, no. 10, p. 1183, 2019.

[36] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 587–597, 2018.

[37] M. Chiang and T. Zhang, "Fog and iot: An overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.

[38] T. Nishio, R. Shinkuma, T. Takahashi, and N. B. Mandayam, "Service-oriented heterogeneous resource sharing for optimizing service latency in mobile cloud," in *Proceedings of the first international workshop on Mobile cloud computing & networking*. ACM, 2013, pp. 19–26.

[39] U. Drolia, R. Martins, J. Tan, A. Chheda, M. Sanghavi, R. Gandhi, and P. Narasimhan, "The case for mobile edge-clouds," in *2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing*. IEEE, 2013, pp. 209–215.

[40] J. Wan, S. Tang, Z. Shu, D. Li, S. Wang, M. Imran, and A. V. Vasilakos, "Software-defined industrial internet of things in the context of industry 4.0," *IEEE Sensors Journal*, vol. 16, no. 20, pp. 7373–7380, 2016.

[41] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, "Are we ready for sdn? implementation challenges for software-defined networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 36–43, 2013.

[42] Z. Zhou, J. Gong, Y. He, and Y. Zhang, "Software defined machine-to-machine communication for smart energy management," *IEEE Communications Magazine*, vol. 55, no. 10, pp. 52–60, 2017.

[43] S. D. Whitehead, "A complexity analysis of cooperative mechanisms in reinforcement learning." in *AAAI*, 1991, pp. 607–613.

[44] M. J. Kearns and S. P. Singh, "Finite-sample convergence rates for q-learning and indirect algorithms," in *Advances in neural information processing systems*, 1999, pp. 996–1002.

[45] C. Alimentarius, "Fresh fruits and vegetables," *Rome, WHO and FAO*, pp. 101–106, 007.

[46] N. Ranasinghe, K. Karunanayaka, A. D. Cheok, O. N. N. Fernando, H. Nii, and P. Gopalakrishnakone, "Digital taste and smell communication," in *Proceedings of the 6th international conference on body area networks*. ICST (Institute for Computer Sciences, Social-Informatics and ..., 2011, pp. 78–84.

[47] D. Krajzewicz, G. Hertkorn, C. Rössel, and P. Wagner, "SUMO (Simulation of Urban MObility)-an open-source traffic simulation," in *Proceedings of the 4th middle East Symposium on Simulation and Modelling (MESM20002)*, 2002, pp. 183–187.

[48] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, 2008.

[49] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, "TraCI: an interface for coupling road traffic and network simulators," in *Proceedings of the 11th communications and networking simulation symposium*, 2008, pp. 155–163.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[51] R. Wallner and R. Cannistra, "An sdn approach: quality of service using big switch's floodlight open-source controller," *Proceedings of the Asia-Pacific Advanced Network*, vol. 35, pp. 14–19, 2013.

[52] B. Pfaff and B. Davie, "The open vswitch database management protocol," Tech. Rep., 2013.

[53] S. Goudarzi, W. H. Hassan, M. H. Anisi, M. K. Khan, and S. A. Soleymani, "Intelligent technique for seamless vertical handover in vehicular networks," *Mobile Networks and Applications*, vol. 23, no. 6, pp. 1462–1477, 2018.

[54] H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep reinforcement learning based resource allocation for V2V communications," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3163–3173, 2019.

Shidrokh Goudarzi received her Ph.D. degree in communication system and wireless network from Malaysia-Japan International Institute of Technology (MJIIT), Universiti Teknologi Malaysia (UTM). In 2014, She received three-year full scholarship to study Ph.D. at (UTM).Then, She joined the Department of Advanced Informatics School at Universiti Teknologi Malaysia as a Postdoctoral Fellow from 2018 to 2019. Currently, she is a senior lecturer at Universiti Kebangsaan Malaysia (UKM). She also serves as reviewer for Canadian Journal of Electrical and Computer Engineering, KSII Transactions on Internet and Information Systems Journal,Journal of Engineering and Technological Sciences, Mathematical Problems in Engineering and IEEE Access. Her research interests are in wireless networks, artificial intelligence, machine learning, next generation networks, Internet of Things (Iot) and Mobile/distributed/Cloud Computing.

Mohammad Hossein Anisi is an Assistant Professor at the School of Computer Science and Electronic Engineering, University of Essex and head of Internet of Everything Laboratory. Prior to that, he worked as a Senior Research Associate at University of East Anglia, UK and Senior Lecturer at University of Malaya, Malaysia where he received 'Excellent Service Award' for his achievements. His research has focused specifically on real world application domains such as energy management, transportation, healthcare and other potential life domains. As a computer scientist, he has designed and developed novel architectures and routing protocols for Internet of Things (IoT) enabling technologies including wireless sensor and actuator networks, vehicular networks, heterogeneous networks, body area networks and his research results have directly contributed to the technology industry. He has strong collaboration with industry and been working with several companies in the UK with the focus on monitoring and automation systems based on IoT concept capable of reliable and seamless generation, transmission, processing and demonstration of data. He has published more than 90 articles in high quality journals and several conference papers and won two medals for his innovations from PECIPTA 2015 and IIDEX 2016 expositions. He has received several International and national funding awards for his fundamental and practical research as PI and Co-I. Dr Anisi is an associate editor of a number of journals including 'IEEE Access', 'Ad Hoc and Sensor Wireless Networks', 'IET Wireless Sensor Systems', 'International Journal of Distributed Sensor Networks'. 'KSII Transactions on Internet and Information Systems journals' and 'Journal of Sensor and Actuator Networks'. He has been guest editor of special issues of the journals and Lead organizer of special sessions and workshops at IEEE conferences such as IEEE CAMAD, IEEE PIMRC, IEEE VTC and etc. He has been also serving as executive/technical committee member of several conferences. Hossein is Fellow of Higher Education Academy and Senior Member of IEEE. He is also a technical committee member of Finnish-Russian University Cooperation in Telecommunications (FRUCT), Senior Member of Institute of Research Engineers and Doctors (the IRED), Member of ACM, IEEE Council on RFID, IEEE Sensors Council, IEEE Systems Council and International Association of Engineers (IAENG).

Hamed Ahmadi is an assistant professor in the department of Electronic Engineering at University of York, UK, He is also an adjunct assistant professor at the school of Electrical and Electronic Engineering, University College Dublin, Ireland. He received his Ph.D. from National University of Singapore in 2012 where he was a funded PhD student at Institute for Infocomm Research, A-STAR. Since then he worked at different academic and industrial positions in Republic of Ireland and UK. Dr. Ahmadi has published more than 50 peer reviewed book chapters, journal and conference papers. He is a member of editorial board of IEEE Access, Frontiers in Blockchain and Springer Wireless Networks. He is a senior member of IEEE, Fellow of UK Higher Education Academy, and Networks working group co-chair and a management committee member of COST Action 15104 (IRACON). His current research interests include design, analysis, and optimization of wireless communications networks, the application of machine learning in wireless networks, airborne networks, wireless network virtualization, blockchain, Internet-of-Things, cognitive radio networks, and small cell and self-organizing networks.

Leila Musavian received her PhD degree in Telecommunications from Kings College London, UK. She is currently working as Deputy Pro-Vice-Chancellor for Research at the University of Essex and as a Reader in Telecommunications at the School of Computer Science and Electronic Engineering, University of Essex. Prior to that, she was a Lecturer at InfoLab21, Lancaster University (2012-2016). She was a Research Associate at McGill University (2011-2012), a research associate at Loughborough University, UK (2009-2010) and a post-doctoral fellow at INRS-EMT, Canada (2006-2008). She has worked as the Deputy Director of Research of the School of Computer Science and Electronic Engineering, University of Essex (2017-2019).

Her research interests lie in Radio Resource Management for Low latency communications, 5G/B5G, next-generation wireless networks, radio resource allocations, Energy Harvesting, Green Communication and cross-layer design for delay QoS provisioning. She is an editor of IEEE TRANSACTIONS OF WIRELESS COMMUNICATIONS. She was Associate Editor of Wiley's Internet Technology Letters. She was Executive Editor of Wiley's Transactions on Emerging Telecommunications Technologies between 2016-2019. She has been lead chair UHS5G WP in IEEE GLOBECOM 2018, UHSLLS WP in IEEE WCNC 2019, lead chair for URLLC Special Session in IEEE PIMRC 2018, TPC Co-Chair of CorNer 2016 (in conjunction with ISWCS 2016) and Co-Chair of mmWave 5G (STEMCOM 2016) and TPC member of several conferences including IEEE ICC, IEEE GLOBECOM, IEEE WCNC, IEEE ICCCN, IEEE PIMRC, ChinaCom. She is currently the Workshop Co-Chair of VTC-Spring-2020 and the Wireless Communications Symposium Lead Co-Chair for IEEE ICC 2021, Montreal, Canada.