Multi-Agent Deep Reinforcement Learning Based Virtual Resource Allocation Through Network Function Virtualization in Internet of Things

Hurmat Ali Shah and Lian Zhao

***Abstract*—Resource allocation is a significant task in the emerging area of Internet of Things (IoT). IoT devices are usually low-cost devices with limited computational power and capabilities for long term communication. In this paper, network function virtualization (NFV) technique is used to access resources of the network and a reinforcement learning (RL) algorithm is used to solve the problem of resource allocation in IoT networks. The traffic of IoT network uses the substrate network which is available through NFV for its data transmission. The data transmission needs of the IoT network are translated to virtual requests and service function chain (SFC) are mapped to the substrate network to serve the requests. The problem of SFC placement while meeting the system constraints of the IoT network is a non-convex problem. In the proposed deep RL (DRL) based resource allocation, the virtual layer acts as a common repository of the network resources. The optimization problem of SFC placement under the system constraints of IoT networks can be formulated as a Markovian decision process (MDP). The MDP problem is solved through a multi-agent DRL algorithm where each agent serves a SFC. Two Q-networks are considered, where one Q-network solves the SFC placement problem while the other updates weights of the Q-network through keeping track of long-term policy changes. The virtual agents serving SFCs interact with the environment, receive reward collectively and update the policy by using the learned experiences. We show that the proposed scheme can solve the optimization problem of SFC placement through adequate reward design, state and action space formulation. Simulation results demonstrate that the multi-agent DRL scheme outperforms the reference schemes in terms of utility gained as measured through different network parameters.**

***Index Terms*—Internet of Things, network virtualization, optimization, machine learning, Q-learning, deep reinforcement learning, resource allocation.**

I. Introduction

Internet of Things (IoT) is becoming the next generation revolution in wireless communication technologies and in sensor networks. IoT devices are low-cost thus allocating costly dedicated spectrum to IoT network will be inefficient. Therefore, the problem of resource allocation to satisfy the operational constraints of the network becomes significant.

The authors are with the Department of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, Canada. e-mails: hurmatali.shah@ryerson.ca and l5zhao@ryerson.ca.

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under Grant STPGP-493787.

A. Motivation and Problem Statement

IoT can work over a substrate network of any communication technology as an extra layer [1]. IoT devices need to have a low-cost and efficient access to communication spectrum, which cannot be provided by the already saturated IST band, as this will degrade the quality of service (QoS) provided. IoT networks have to be latched onto available physical communication networks. To access resources of communication networks, softwarization of the networks is important, such that the IoT network can become an extended part of the available infrastructure. Resource allocation in this scenario, where IoT networks don't have their own dedicated resources, becomes important. The motivation for this work is to propose a framework for network function virtualization (NFV) based resource allocation for IoT networks.

The solution to the spectrum access problem for IoT networks is a combination of low-cost networks which can combine multiple network technologies in order to serve the IoT traffic [2]. To access the network substrates and to allocate resources jointly, centralized NFV is used extensively in next generation communication technologies such as intelligent communication systems, IoT and smart grids. The work which has attracted much attention is the placing of virtualized network functions (VNF) and service function chains (SFC) on the substrate network. NFV conceptualizes the physical network virtually through VNF and SFC. VNF is the virtual translation of the network functions which a physical node is able to provide. SFC is a directed list of instances of functions required by network traffic to traverse, i.e., it is a path from the sender to the destination which has to pass through different intermediate nodes providing required functions (VNFs) and meeting the system performance criteria.

IoT is an environment which is susceptible to learning. Reinforcement learning (RL) solutions and algorithms are most effective in situations where there are patterns and models which can be learned [3-5]. In the area of communications, the methods of deep reinforcement learning (DRL) can be effective in solving many challenges, particularly in emerging fields varying from sensor networks to IoT to heterogeneous networks. The problems of IoT, such as resource allocation, scheduling for transmission and power control in energy-constrained situations, can be devised as a Markov decision process (MDP) when DRL is employed. The combination of NFV with DRL can solve the problem of efficient resource allocation. NFV can employ virtualization technology

to consolidate multiple network equipment types onto industry standard high volume servers, switches and storage devices, which act as resources for the IoT [6-8]. While, DRL can learn the behavior of environment and allocate resources of the network accordingly.

In this work, a resource allocation scheme which employs DRL is used in NFV based IoT. It is considered that the spectrum available for transmission belongs to the substrate networks such that the resource allocation in terms of spectrum is a central task belonging to the NFV control and gateway. The IoT devices are connected with a physical IoT control center and over the IoT control central plane, i.e., the physical plane, the virtual plane, i.e. the cloud virtual network, operates. The proposed scheme deploys software defined radio (SDR) to connect cellular infrastructure and to control the operational parameters. A DRL based algorithm is also employed at the virtual controller to solve the problem of resource allocation. The algorithm learns the environment and then according to the reward function changes policy of scheduling and of power allocation. Through the DRL algorithm the complex optimization problem of resource allocation can be solved in a holistic way such that convergence of performance can be achieved.

B. Contributions

In this paper, SFC placing problem for IoT network is considered. The SFC placement is dynamic, as the real-time network congestion of nodes and links is taken into consideration. Different from static approach to SFC placement, the proposed scheme in this paper uses a temporally robust and proactive SFC placement by integrating the past information of the network into the SFC placement decision. The SFC placement problem is solved through DRL which avoids the need for exact channel state information (CSI) by learning behavior of the network. Therefore, the SFC placement is carried according to both demands of the IoT users and behavior of the environment. The contribution of this paper can be summarized as:

- We propose a SFC placement model which takes into account the stochastic information of both the link and node congestion. The SFC placement for IoT network is solved through a two stage decision: the first stage takes decision centrally where the IoT devices to be served are selected. In the second stage the virtual service requests are translated to the physical demand and mapped to the substrate network, while maximizing utility of the overall IoT network.
- We solve the SFC placement problem for IoT by modelling it through multi-agent RL. Because of the two stage conceptualization of the SFC placement problem, DRL is employed to solve the problem which improves the individual utility of the IoT device as well of the whole IoT network simultaneously. Two Q-networks in the DRL based algorithm have been considered with different weights. Through the weights of one Q-network, actions are taken, while on the basis of long-term behavior, the second Q-network adjusts the weights of the first Q-network. The use of multi-agent DRL ensures reliability in terms of long-term utility.
- We show that the multi-agent DRL algorithm through a proper reward design and training procedure enables the SFC placement controller to learn from interaction with the environment. The proposed algorithm is able to figure out an intelligent strategy through the target and actual Q-networks for working cooperatively such that the system performance in terms of utility is optimized.

The remaining of the paper is organized as below. Section II presents literature review. Section III presents the system model along with the system constraints and formulation of the optimization problem. Section IV considers solving the optimization problem through DRL. Section V presents simulation results and discussion, while Section VI concludes the paper.

II. Related Work

Cheng et al [9] considered a stochastic approach to NFV based solution where the hierarchical decision structure was exploited for a distributed solution. In [10], a network slice embedding was considered for 5G networks, where the optimization problem aimed to reduce the minimum of maximum number of service chains affected in case of network failure. Deterministic solutions are based on mixed integer linear programming (MILP) where authors in [11] and [12] proposed such solutions for the first time. In [11], a pareto-optimal solution was provided but the scalability issue of VNF orchestration was left unresolved. In [12], an integer linear programming (ILP) based VNF orchestration problem was solved through dynamic programming oriented heuristic for large instances.

VNFs are also employed in cloud networks. In [13], a content delivery network was proposed on the basis of multiple clouds and the VNF placement problem was formulated as two integer model. In [14] a distributed solution based on game theory was investigated, while in [15] a coalitional game was proposed to find the optimal number of slices of 4G or 5G networks to meet the network traffic. Dynamic solution to the SFC placement problem was proposed in [16] and [17], with focus on geo-distributed centers and migration respectively. These works find solution through ILP and decomposition approaches, thus they are not resilient to changes. The exact solution to problems of successive placement also is hard to obtain. To avoid assuming exact knowledge of the environment, a support vector regression based predictive model SFC placement was proposed in [18].

RL and DRL are extensively deployed for cellular and other wireless network models and solutions. In [19], the varying dynamics of conventional network traffic was leveraged to transmit the IoT traffic. RL has been applied to a wide variety of problems in the domain of wireless communications such as in [20-22]. Machine learning (ML) formulations fit well in environment with dynamic variations and were applied in [23] for resource management while in [24] DRL was used for dynamic spectrum access. A ML based distributed power allocation scheme is used in [24] and [25] where the devices cooperated without knowing the exact CSI. Considering rapid transitions and changes in vehicular networks, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3022572, IEEE Internet of Things Journal

exact solution through sequential decision making is hard. Thus, the study of vehicular communications has benefited the most as there are many cutting-edge solutions on the basis of RL to the problems ranging from mobility, resource allocation and vehicle-to-vehicle and vehicle-to-infrastructure communications [26-29].

ML based solutions for SFC placement problem in IoT networks are unexplored in the open literature. On the other hand, learning techniques for other purposes have been applied to IoT networks. A distributed learning based approach was considered in [30] for IoT devices with low memory and computational capabilities to enable diverse IoT devices to share limited communication resources. In [31], virtual machine allocation in cloud-fog computing was considered, while [32] presents a detailed survey of applying machine learning in IoT networks. Industrial IoT is an important area where vulnerability to attacks has to be guarded against. Such a scheme was investigated in [33]. Machine learning techniques for intrusion detection in IoT security based systems were surveyed in [34].

III. System Model and Problem Formulation

A. System Model

Assume an IoT network consisting of N IoT devices where each device has to transmit its data but is not allocated a dedicated spectrum. The IoT network is connected to a cellular network via a common fusion center. A cloud network controls both the IoT network and radio access for the IoT network to the cellular network. As the IoT devices have different QoS requirements such as delay and data rate, therefore the delay-sensitive applications are granted priority in terms of transmission while applications which are delay-tolerant can be scheduled later. An IoT network can have various IoT devices, from sensor devices with regular transmission intervals to downloading update to sensor devices requiring large data rates to home entertainment systems with higher QoS requirements. This diversity of delay and data requirements, thus allows the IoT networks to use the spectrum according to different QoS needs. Therefore, it leads to the assumption that the spectrum is always occupied and IoT devices transmit data according to the IoT device scheduling.

The physical substrate network is conceptualized as provider of VNFs by a cloud controller as shown in Fig. 1, where the physical network is separated from the cloud network. In the virtual network layer, the physical cellular network is mapped as virtual nodes with virtual links among them. In the virtual network services layer, the service requests from the IoT network are translated into a set of SFCs containing VNFs to be mapped on to physical nodes. SFC has a combined representation of both the physical nodes and the physical links available among them. As can be seen in Fig. 1, the nodes in virtual network layer are mapped onto the physical nodes in cellular network. Each SFC serves one virtual service request, while each IoT device which is being served can have more than one SFC at the current time slot. The physical nodes can host parts of multiple SFCs depending on the capacities available on them. The virtual links in the virtual network layer are mapped onto and use the physical spectrum of the cellular network nodes.

TABLE I
List of important notations.

V_L	Set of physical nodes
E_L	Set of physical links
V_s	Set of virtual nodes
E_s	Set of virtual links
R_L	All physical resources
m_s	Consolidated node and link capacities
$\lambda_{x,y}^z$	Path from node x to y over link z
$\prod^s$	All paths for s
f	Virtual network function
$a_{f\rightarrow v_L}$	Mapping VNF f to physical node v_L
$a_{z\rightarrow e_L}$	Mapping virtual link z to physical link e_L
s	Service function chain
α_i	Variable deciding serving of IoT device i
$c_.(t)$	Capacity at node or link at time t
$k_.(t)$	Cost of per unit resource at a node or link a time t
O_t	System state at time t
A_t	Action taken at time t
U_t	Utility function at time t
π	Policy function
M_t	Reward returned at time t

SFC mapping is elaborated through a sample example in Fig. 1. In the 'virtual network services layer' part of the system model, as shown in Fig. 1, virtual services are designed and then mapped to physical network as SFCs. In the first virtual service layer, as given in the figure, the IoT node which is served is represented as 'v_s'. Two virtual links 'z_1' and 'z_2' are allocated to virtual service 1. The virtual links traverse different virtual nodes. The virtual nodes are then mapped to the physical nodes in the cellular network. The number over each physical node, as shown in the figure, corresponds to the respective virtual node the physical node is hosting. 'Path 1' and 'Path 2' are the physical links which implement the virtual links 'z_1' and 'z_2', respectively. The virtual node and link placement should satisfy the system constraints as in terms of the capacity available, the cost incurred, interference, transmission power, and fairness. The important notations used in the paper are listed in Table 1.

The physical substrate network can be represented as a directed network graph as $G_L = (V_L, E_L)$. The network functions which can be offered by the nodes belonging to the substrate network are represented as tp. The nodes which can provide the services of type tp are represented as $v_L^{tp} \in V_L$. The consolidated resources of the node are represented as $R_L^v = \{(tp, v)/\vee tp, v \in v_L^{tp}\}$, which means the set of all type of services which can be handled by node v. All the substrate resources are represented as $R_L = R_L^v \cup E_L$, whereas E_L represents all physical links.

In the same way the virtual network can be visualized as a directed graph as $G_s = (V_s, E_s)$. A request s with service rate demand α_i is a directed graph with start node $v_{snode} \in V_w$ and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3022572, IEEE Internet of Things Journal

end node $v_{enode} \in V_w$. This service request, s, is implemented as SFC. Since each node is well-defined, consolidated node and link capacities are used as $m_s : v_s \cup E_s \rightarrow R_L \geq 0$. This implies that the service s uses consolidated resources of the nodes which serve s, i.e., v_s and the links, i.e., E_s.

A mapping $g_s, s \in S$ is a tuple (g_s^V, g_s^E) of functions. The function $g_s^V : V_s \rightarrow V_L$ maps each network function to a single substrate node. The function $g_s^E : E_s \rightarrow E_L$ maps edges between network functions in the substrate network. Moreover, $g_s^V \in V_s^{tp(i)} \vee tp \in V_s$, i.e., the service should be held at the node which has the physical resources available for the service at the current time slot. Also, g_s^E creates a path which embeds service s such as $\lambda_{x,y}^z \in \prod^s$. The substrate nodes to which virtual nodes, (x, y), are mapped over physical link z are given by (v_{snode}, v_{enode}). While, $\prod^s$ is a combination of all paths which are created for serving the request s.

The mapping of service request s is translated into a single, or if the demand requires, multiple paths. The VNF, $f \in g_s^V \cup g$, is the virtual translation of the physical node used to serve service request s. The mapping function $\lambda_{x,y}^z$ is a combination of two mapping variables: $a_{f \rightarrow v_L}$ which translates the function g_s^V into mapping. The variable $a_{z \rightarrow e_L}$ places the virtual link over the substrate link where $z_{x,y} \in g_s^E$ is the virtual link to serve services (for sake of simplicity $z_{x,y}$ is henceforth represented as z). $a_{f \rightarrow v_L}$ and $a_{z \rightarrow e_L}$ are given as

$$a_{f \rightarrow v_L} = \begin{cases} 1, & \text{if } f \text{ is placed over } v_l; \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

and

$$a_{z \rightarrow e_L} = \begin{cases} 1, & \text{if } z \text{ is placed over } e_l; \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The SNR over each link z for the i-the device is given as

$$\gamma_{i,z}^s(t) = \frac{I^{i,z}(t) p_i(t)}{\sigma_z^2(t) + \sum\limits_{\forall\, i'} I^{i',z}(t) p_{i'}(t),} \tag{3}$$

where $I^{i,z}(t)$ and $p_i(t)$ are the channel gain over the z-th channel and the transmission power at time t for the i-th IoT device, respectively. σ_z^2 is the noise variance while $I^{i',z}(t)$ and $p_{i'}(t)$ are the channel gains of all other devices over channel z and the transmission power of all other devices, respectively. The data rate achieved over service s at time t, is denoted by $\beta^s(t)$ and is given as,

$$\beta^s(t) = \sum_{\forall\, z \in \lambda} p_i(t) B_i^z(t) \log_2(1 + \gamma_{i,z}^s(t)) \tag{4}$$

where B_i^z is the bandwidth assigned.

The mapping function then is given as $\lambda_{\cdot}^{\cdot} = \{a_{f \rightarrow v_L}, a_{z \rightarrow e_L}\}$. It was shown in [9] that this mapping problem is NP-hard. Hence, the mapping problem requires non-polynomial time to solve, even when other constraints like available resources, capacities and power transmission requirements, are not taken into consideration. In subsequent sections the mapping problem will be represented as part of the optimization problem maximize system utility.

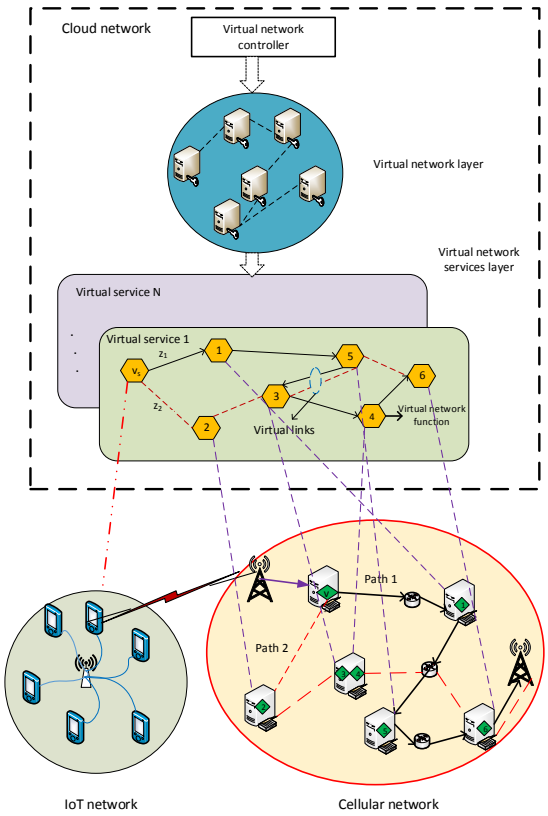

Fig. 1. Basic system model.

B. System Constraints

The first constraint is that the sum of all rate allocated for all virtual paths which are mapped at the node and link should be below the consolidated node and link capacity,

$$\sum_{\lambda \in \prod^s} \lambda_{x,y}^z \beta_s \leq m_s, \quad \forall \, \mathrm{S}. \tag{5}$$

The above constraint takes consideration of consolidated capacities of links and nodes but it is also needed that the summation of all VNFs mapped to a node or a link should be below the individual node and link capacities at time t as,

$$\sum_{\forall\, f} \lambda_{\cdot}^{\cdot}{}_x \leq c_x(t) \tag{6}$$

where $c_x(t)$ is the capacity of an individual node or link. Equation (7) states that the summation of all VNFs mapped to a link should be less than the available capacity,

$$\sum_{\forall\, f} \lambda_{\cdot}^{z} \leq c_z(t). \tag{7}$$

The resource allocation problem is competitive in its nature and is formulated as a competitive game in literature. The introduction of a fairness constraint converts the competitive game into a cooperative one. The constraint is formulated as,

$$\beta_{lower} \leq \beta_s \leq \beta_{upper}, \tag{8}$$

which states that the rate allocated to a service s should be within a lower and upper bound. To constrain the cost of each

mapped link to be fair, if $d(x, y)$ is the acceptable number of hops between the start and end node, a constraint is introduced which finds a substrate path with affordable number of hops as,

$$L(\sum_{\forall(x,y,z)} \lambda^z_{x,y}) - 1 \leq d(x, y),\ (x, y) \in V_L, z \in E_L. \quad (9)$$

Each VNF should be assigned to one substrate node and link only. It is stated as,

$$\sum_{v_l \in V_s} a_{f \to v_l} = 1,\ \sum_{e_l \in E_s} a_{z \to e_l} = 1. \quad (10)$$

The interference caused over a link if it is used for transmission of data belonging to device i should be below the maximum interference threshold denoted by I_{max}. This is important for IoT as IoT devices are in the same vicinity and may use the same or adjoining channels for their data transmission. The constraint is given as,

$$\sum_{\forall\, z \in \lambda} I^{i',z}(t) p_{i'}(t) \leq I_{\max}. \quad (11)$$

IoT devices have limited power available, therefore, the power for the i-th IoT device, i.e., p_i, has to be budgeted. The limitation on the maximum transmission power, P_{max}, takes into consideration energy restrained environment of the IoT. Thus,

$$\sum_{\forall\, s} p_i \leq P_{\max}. \quad (12)$$

C. Utility Function

Utility of the system is the gain achieved from mapping the SFCs and the data rate achieved while subtracting the cost incurred over mapping to the substrate. Let a_i be a binary variable indicating whether IoT device i is being served or not, b^s be the benefit which is achieved over SFC s, n_i be the total number of SFCs belonging to IoT device i, $k_{\cdot}(t)$ be the price of using resource at an individual node or link and ρ be the ratio of served IoT devices to the total number of active IoT devices with data to transmit. The optimization problem is to maximize the objective function for utility at time t as given in (13). The mapping problem is a mixed integer linear programming (MILP) problem. The problem is formulated with specific constraints and requirements of IoT network, cost formulation of the nodes and links, and placement in terms of the specific requirements of IoT network. Similar MILP problem was solved through stochastic decomposition methods in [9]. The MILP problem in this paper like in [9] is NP-hard.

The utility function in (13) depends on three optimization variables. The first optimization variable is a_i, which selects the IoT devices to serve at time t. The second optimization variable is β^s, which is the data rate but is controlled through the transmission power of the IoT devices, i.e., p_i as well by the channel gains over the selected paths given by the third optimization variable. While, the third optimization variable is $\lambda^z_{x,y}$, which determines the path which can be one of the multiple paths designed for virtual service request implemented as SFC s.

IV. Multi-Agent Deep Reinforcement based Resource Allocation

The optimization problem given in (13) is non-convex and NP-hard, thus the solution to it is non-deterministic, inexact and with only an approximate solution possible in polynomial time. The exact solution, therefore, to the problem in (13) cannot be found due to uncertainty in CSI and the rapidly changing wireless environment. The mathematically exact solution also is vulnerable to the curse of scalability where any increase in the number of system components, in this case the number of IoT devices and available nodes and links of the substrate, makes finding the optimal solution prohibitively exhaustive. The problem of resource allocation for IoT with optimal or pareto-optimal solutions which can be learned from environment can be found through RL. The RL based solutions are both mathematically and computationally affordable.

Sequential decision making, in the face of uncertainty of channel conditions and changing dynamics of the wireless environment, is hard given the multi-constraint requirement of the maximization problem in (13). The sequential decision making problem, where for each time horizon the approximate solution to (13) is found, can be solved if the problem is translated to MDP. After formulating the problem as MDP, it can be solved through DRL. The DRL based solution can also take into consideration the long-term dynamics of the wireless environment and hence being robust to future changes. Q-learning (QL) is one of the most popular and computationally affordable algorithms to solve multi-agent problems, while deep QL (DQL) solves DRL problems which have more than one layer of decision making. Experience replay makes DQL robust because it takes into consideration the long-term changing dynamics of the system. Key parts of the proposed multi-agent DQL (MA-DQL) scheme are presented in detail below.

A. State and Action Space

In the MA-DQL formulation of the resource allocation problem given in (13), virtual network agents map the SFCs. The virtual network agent takes an action given a state and receives a reward for the action taken. The system state for the whole network is updated as a centralized perspective from the cloud network. The virtual network cloud controller selects IoT devices to be served. Thus SFCs of the devices to be served are mapped by virtual network agents onto physical network. Given a state O_t, a virtual network agent takes a consolidated action (combined with action of the cloud controller) as A^N_t for the SFC belonging to the N-th IoT device and receives a reward which in this case is the utility, calculated in (13), as U^N_{t+1}. The probability to transition to the next state, O_{t+1}, is given as $Pr(O_{t+1}, U_t | O_t, A_t)$ according to policy function given as $\pi(A_t|O_t)$.

The state space consists of updated capacities of both the substrate nodes and the links, new serving requests and the service requests not served in the previous time slot. The combined capacities at time t for the nodes and links are represented as $C(t)$. The new service requests to be served are

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3022572, IEEE Internet of Things Journal

$$U_t = \rho \max_{a_{i,\beta_s}, \lambda^z_{x,y}} \sum_{\forall i} a_i [\sum_{s \in n_i} \beta^s (b^s - \sum_{(x,y) \in V_L} \sum_{z \in E_l} \sum_{\lambda^z_{x,y} \in \Pi^s} c_x(t)k_x(t) + c_y(t)k_y(t) + c_z(t)k_z(t))\lambda^z_{x,y})]$$
$$\text{s.t (5) - (12).} \tag{13}$$

F_{new} and the leftover from the previous time slot are F_{old}. The state of the system is thus given as

$$O_t = \{C(t), F_{new}, F_{old}\}. \tag{14}$$

The action space consists of selecting IoT devices to be served, the power allocated to each device as the data rate in (4) depends on the transmission power of the IoT devices and the path selected for each service request. The joint action set, if F_{ser} is the total service requests to be served, is represented as

$$A_t = \left\{a_i, \forall_{F_{ser}} \lambda^z_{x,y}, p_i\right\}. \tag{15}$$

B. Reward Design

The reward which is based on the utility in (13) is myopic. However, DQL handles the long-term system stability by taking into account long-term cumulative award. The goal of DQL is to learn an optimal policy $\pi^*(A_t|O_t)$ which maximizes the expected return from any initial state O_t. The return is given as,

$$M_t = \sum_{i=1}^{N} \sum_{k=0}^{\infty} \gamma^k U_{t+k+1}, \ 0 \leq \gamma \leq 1 \tag{16}$$

while γ is the discount factor. The reward function is long-term utility, where utility is calculated in eq. (13). From eq. (13) it can be seen that the utility is dependent upon action taken and the system's state space at the time. The reward function, thus, is dependent implicitly upon the action and state space through the utility function. The SFC placement problem depends upon the substrate network's resources available, which is considered through system constraints of the optimization problem in eq. (13). In eq. (16) the general form of long-term reward is used as has been done in Q-learning and other machine learning techniques. However, the problem related nature and specifics of the utilization maximization problem are also included implicitly in the reward function given in eq. (16), through eq. (13).

C. MA-DQL Training Procedure

A time horizon is considered here which calculates and updates the weights for the current action-value and the effect of the long-term short memory (LTSM) of the system states. Let D be the replay memory where the transition tuple $(O_t, A_t, M_{t+1}, O_{t+1})$ are stored. The deep learning can be modeled as a function of the form $out = func(in; \theta, w) = \phi(in, \theta)^{Transpose} w$. The parameter θ is used to learn ϕ from a variety of functions, such as utility function here, while w maps from mini-batch experiences from D to ϕ by adjusting the weights of each action-mapping parameter, i.e., a_i, $\lambda^z_{x,y}$ and p_i. The parameter w is long-term weight adjustment of θ over time horizon T. The network parameterized by $w^{'}$ is the target network which has its weights duplicated from learning Q-network through w. The parameters of the target Q-network are updated periodically and fixed for a few updates. The parameters of the action-taking network, also called evaluated Q-network, i.e., the network parametrized here by θ, are adjusted through the weights of the target network.

Q-learning is concerned with learning the action-value which is the value returned on the action taken in a given state. The state-action value function is represented as $Q_\pi(O_t, A_t)$. It gives the expected return, while starting from state O and taking action A according to policy π, and thereafter following the policy π for future action taken on the basis of state transitions. The action-value function for Q-learning is formally expressed as,

$$Q_\pi(O, A) = \mathrm{E}_\pi(M_t | O_t = O, A_t = A). \tag{17}$$

Once the state-action value function $Q_\pi(O, A)$ is found then it is easy to obtain the optimal policy $\pi^*(A_t|O_t)$. The state-action value function can have two definitions: 1) the total accumulated reward while starting in the current state; and 2) going to next states according to the given policy. In the state-action value function, the expected accumulated rewards are calculated while taking an action in the current state, and then, in the next state taking other actions according to the given policy. The value function adjusts the policy function in the agent on the basis of observations from the environment. With the stochastic approximation of the learning rate and the assumption that all state-action pairs are updated, the state-action value function in Q-learning converges to the optimal policy π^* with probability 1. In the training phase, the cloud controller and virtual network agents take the current observation of the state as input and return the value functions corresponding to all actions. The Q-network runs for multiple episodes at each training step and the state-action space is explored with soft-policies, i.e., ε-greedy, which means that the action with maximal estimated value is chosen with probability $1 - \varepsilon$ and a random action is taken with probability ε. Following the state transition due to action taken, transition tuple is collected as $(O_t, A_t, M_{t+1}, O_{t+1})$ and stored in replay memory D. At each episode, a mini-batch of experiences from D are uniformly sampled from the memory for updating w with stochastic gradient-descent methods. The sum-squared error is minimized as,

$$\sum_{D} \left[M_{t+1} + \gamma Q_\theta \left(O_{t+1}, A^{'}; w^{'}\right) - Q\left(O_t, A_t; w\right)\right]^2. \tag{18}$$

DQL estimates the system and action space rather than computing them directly as is done in policy gradient methods, such as actor-critic algorithms. The dynamic nature of the virtual agents adds a layer of complexity which makes directly computing the system parameters harder, therefore, DQL based estimation is used. Temporal difference (TD) error is calculated for the estimation of the system space and action space parameters through the evaluated and target network. The weights of the action-taking Q-network are updated through the TD-error with respect to the target network. Multiple agents are considered in this scheme where each agent learns the environment locally, i.e., through local parameters and weights, and learns local optimal policy which is fed into by the global policy. Each agent takes a local optimal action and on the basis of updated observed state the global policy calculates the global action parameter which is to select the IoT devices to be served. We demonstrate why MA-DQL performs better than single-agent or conventional DQL based schemes in the next section.

Algorithm 1 Resource Allocation through Multi-Agent DQL

1: Model the network with wireless environment, IoT devices, substrate nodes and links
2: Initialize Q-network randomly
3: **for** each time horizon **do**
4: Input all the IoT devices with resource request
5: Update all available resource for each type i.e., $\forall_{tp} V^{tp}$
6: The cloud controller decides which IoT devices to serve i.e., whether $a_i = 1$ or $a_i = 0 \, \forall \, i$
7: Design for each $a_i = 1$, s_i such that $s_i = \left(g^V_{s_i}, g^E_{s_i}\right)$ where $V \in V^{tp} \, \forall \, s$
8: Select which SFCs to serve as F_{ser} from F_{new} and F_{old}
9: **for** $\forall \, f \in F_{ser}$ call Algorithm 2 to find $\lambda^z_{x,y}$ where (x, y, z) are returned by Algorithm 2
10: Observe O_t
11: Select p_i and $\lambda^z_{x,y}$ from step 9 and update action-value for current myopic action as A^{myopic}_t with weight w
12: Update action-values for all actions sampled from D with weight $w^{'}$
13: Chose an action with maximal estimated action-value with probability $1 - \varepsilon$ and a random action out of A^{myopic}_t and D with probability ε
14: **end for**
15: All virtual network agents take action and receive reward as M_{t+1}
16: Update all $C(t)$
17: Observe O_{t+1}
18: Store $(O_t, A_t, M_{t+1}, O_{t+1})$ in the replay memory D
19: Uniformly sample mini-batches from D
20: On the basis of step 18, update the weights and minimize the error between Q-network and learning targets using stochastic gradient descent.
21: **end for**

D. MA-DQL Algorithm

In Fig. 2, the basic flowchart of the proposed multi-agent DRL scheme is depicted. The cloud controller takes initial action which is to select the IoT devices to serve in the current time slot. The virtual network agents then implement SFCs of the selected devices. On the basis of joint action, node and link capacities of the system are updated. While, on the basis of reward returned, the policy is updated. Future actions are taken according to the updated policy.

The proposed algorithm for MA-DQL based resource allocation is presented in Algorithm 1. Algorithm 1 is a centralized perspective on the network and the observations made are for the whole system. Thus there is no instability for two reasons. Firstly, because of the nature of the resource allocation problem which is formulated as a cooperative game. Secondly, because the resource allocation occurs holistically. In Algorithm 2, a myopic action is taken in the current time slot and the output is returned to Algorithm 1 in Step 9 for further decision taking. The task of resource allocation is also made efficient through multiple agents working to assign the resource allocation to the SFCs through the virtual network agents.

The training procedure can be computationally intensive in Algorithm 1, but it can be performed off-line over multiple time horizons for various channel conditions, levels of powers allocated to the IoT devices, capacities of nodes and links, as well for different number of nodes and links in the substrate network. The top-layer of NFV formation can remain similar and hence the changing dynamics of the substrate network are abstracted from the functioning of IoT network. Once the Q-network is trained and optimal policy is obtained, running and mapping the IoT requests to the substrate network in an optimal way becomes straightforward. Moreover, the training needs to be performed again only when there are significant changes in the substrate network or the wireless environment.

Algorithm 1 is explained through the depiction in Fig. 2 as following. The virtual network controller in Fig. 2 takes Steps 1-6. After deciding which IoT devices and SFCs to serve, the virtual network controller forms the virtual network agents to map the SFCs. This corresponds to Step 9 in Algorithm 1. The virtual network agents implement Steps 9-15. The block diagrams which update the capacities and return reward after interacting with the environment are given in Steps 16 and 17. Steps 18-20 are depicted in the policy update block and these steps are controlled by the virtual network controller. As has been explained earlier, Algorithm 2 is implemented by virtual network agents in Step 9 of Algorithm 1.

It is noted that the proposed scheme also does not have explicit communication among the DQL agents. However, the DQL agents are fed the trajectory of policy changes of other agents implicitly through the cloud controller as the updating of Q-table and policy formulation take place. Furthermore, the DQL agents are created in response to the service requests, hence they are dynamic, but the result of each DQL agent is stored and reflected in the combined Q-table and policy function [34]. The DQL agents are software elements in the cloud server which provide distributed solution to the demand

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3022572, IEEE Internet of Things Journal

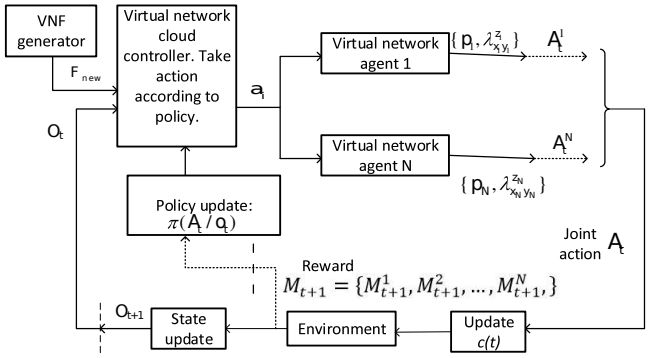

Fig. 2. Multi-agent DRL based resource allocation interaction scheme.

for resource allocation of each SFC mapping. It is also noted that there can be conflicting resource allocation and placement decisions over the substrate networks. However, this will result in nil reward. The states and the consequent action which resulted in a nil reward will be avoided through the training process where the optimal policy is learned.

Each agent has its own DQN, which is high-dimensional. This high-dimensional output is too large to be given as input to DQNs of other agents. However, the policy changes of other agents (which are actions taken given a state) can be taken into account by an agent if only those state-action pairs are considered as input to the DQN of an agent which actually occurs in its own memory. This technique was proposed by [34] and is called multi-agent fingerprint. The low-dimensional fingerprint is designed in such a way that it smooths over the training phase, thus allowing the model to generalize across experiences of the other agents as they execute policies of varying quality as they learn. Two factors thus determine the design of the fingerprint, i) training iteration number which then determines the training memory of an agent, ii) an exploration rate to select state-action pair from the training window. We have taken the exploration rate to be the same as ϵ, which is the same rate as for exploration of soft policies of our scheme. Through this low-dimensional fingerprint the DQNs of multiple agents interact and cooperate with each other.

V. Results and Discussion

In this section, simulation results are presented to validate the performance of the proposed MA-DQL scheme. For simulation purpose, a variation of the framework for SFC placement used by [18] is considered. A network is designed using Matlab R2016b and the nodes along with edges of the network are defined through different parameters which represent the features and capacities of the corresponding substrate network. These features are as: remaining link capacity (r_1), transmission power (r_2), node processing capacity (r_3), node-link consolidated capacity (r_4), and distance of the node from the IoT network (r_5). These five features are combined to instantiate one training vector as $\boldsymbol{r} = \{r_1, r_2, r_3, r_4, r_5\}$. The cost of the resources used is determined through Algorithm 2 for each link and node as corresponding for each IoT device. Theoretically the nodes can provide any service needed and

Algorithm 2 Myopic Action through Cost Calculation

1: Input: s_i
2: Calculate the price of each link and cost of serving s_i on each node and link as
For link: From the previous history of period $T^{'}$ find the average amount of bandwidth used over link z as $\forall\, z,\ h_z(t) = \frac{\sum_{t^{'} \le t-1} B_z^{t^{'}}}{|T^{'}|}$ where $B_z^{t^{'}}$ is the amount of bandwidth used on link z till current time t, while $\left|T^{'}\right|$ is the total time passed in the current time horizon. The per unit cost of using link z at time t is then: $k_z(t) = \frac{cf_z(t)}{h_z(t)}$, where $cf_z(t)$ is cost function which is pre-fixed and gives the price of bandwidth used and it is defined: if $z^{'}$ is the amount of bandwidth used on z then $cf_z(t) = e^{z^{'}}$.
For node: Calculate for all $v_l \in V_{s_i}^{tp}$ the cost of using per unit resource at v_l as $k_{v_l} = \gamma_{i,v_l} + h_{v_l}(t)$ where $h_{v_l}(t)$ is defined similarly as $h_z(t)$.
3: Combine the per unit price of each node with per unit price of link available over the node as $c_s : k_{v_l} \rightarrow k_z$
4: From c_s find $(x, y, z) = \min(c_s)$
5: Repeat step 4 till $y = v_{enode}$
6. Return $\lambda_{x,y}^{z}\ \forall (x, y, z)$

thus the placement becomes finding a chain of nodes which can meet different service criteria. However, for simulation we have considered only one resource type to be available as in [36]. The maximum interference threshold is fixed at 5 dBm by default. The radio bandwidth is assumed to be 1 GHz.

The deep Q-learning network for each agent and cloud controller consists of a network of 3 fully connected hidden layers containing 500, 300 and 150 neurons respectively. A trajectory of policy changes of other agents, as proposed by [34], which is carried through a low-dimension fingerprint, is included as input to an agent. Both the state space and action space are continuous because of capacity of nodes and links, and of the power allocated to each device, respectively. The state space is discretized till the reward changes negligibly as in [19] while the action space is discretized through limiting the range of power allocated to 4 discrete levels, i.e., $[31.3, 20, 12, 5]$ dBm. After discretization, Q-function in (17) is computed.

The learning rate is determined by checking it for return in Q-values. Each agent's Q-network is trained for 3000 episodes with changing both the exploring rate, ϵ, and the long-term weight of the deep learning i.e., w, linearly over the first 2000 episodes if the return on Q-value does not change beyond a threshold, otherwise for 2500 episodes. The Q-network retains the exploration rate and long-term weights at which the Q-table converged for the remaining episodes. When the learning rate stabilizes then it is used for rest of the training. To determine the number of epochs, it is varied according to the convergence behavior of the proposed scheme. When the gain in utility converges to a point, then the number of epochs at which convergence was reached is used as the number of training epochs. The rectified linear

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3022572, IEEE Internet of Things Journal

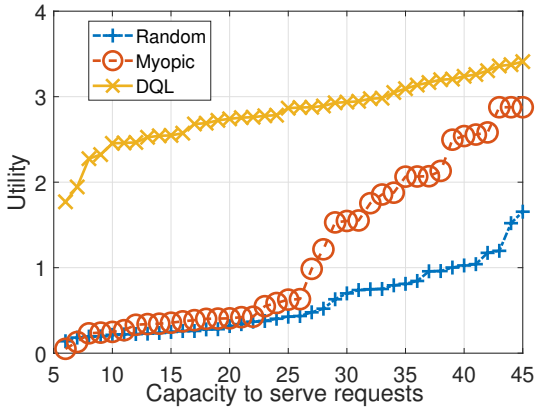

Fig. 3. Performance comparison with respect to capacity of the substrate network.

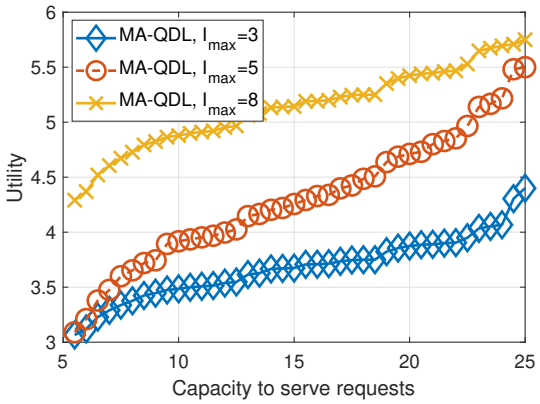

Fig. 4. Effect of maximum interference threshold on system's performance.

unit (ReLU) is taken as activation function and is given by $fun(in) = max(in, 0)$. If our problem was of a classification kind then other activation functions, such as 'softmax' would have been considered. However, we have used the ReLU activation function because it is suited the most to action-space of our proposed formulation and has been considered extensively in literature for problems of similar kinds, such as in [22], [29], [34] and [35].

We compare our proposed MA-DQL resource allocation and SFC placement scheme against two baseline schemes: a baseline random scheme and a myopic scheme. The baseline random scheme selects IoT devices to be served randomly. From the available links and nodes, a path is created to serve the service request while the transmission power is selected according to the interference threshold and the transmission energy available. The myopic scheme is variation of Algorithm 2 presented before, i.e., it takes an optimal decision on the basis of cost function in the current time slot only. The proposed multi-agent DQL scheme on other hand takes a placement decision both on the basis of the policy variation of other agents as well as the future time horizon, i.e., long-term effect of the policy decision in the current time slot.

The number of IoT devices considered for Fig. 3 and Fig. 4 is 5. The reason for keeping the number of IoT devices low for these two simulations is to keep the value of ρ in (13) near to 1. The low number of IoT devices in Fig. 3 and Fig. 4 allows the performance to be checked for the number of requests the schemes are able to serve as against the available capacity of the substrate network.

In Fig. 3, the performance improvement of the proposed scheme can be seen as compared to the other schemes. The utility achieved is the long-term utility as given in (16). The number of requests to be served here is varied between 15 and 25 for each episode to have a good reflection of performance evaluation. The random scheme performs the worst and gives very limited utility when the capacity of the substrate network is low because the cost of the SFC placement is not taken into consideration, hence any gain which is achieved is off-set by the cost both of the link and nodes which is incurred by the placement. On the other hand, myopic scheme's performance greatly improves when the capacity of the substrate network goes beyond 25. The proposed scheme outperforms the other two schemes not only when the capacity of the substrate network can meet all the requests but also when the capacity is lower than the number of requests. The significant improvement is because of avoiding links, nodes and paths which have been overused, thus, reducing the cost as well as selecting the ones which can give better end-to-end SNR.

The performance of both the proposed and myopic schemes is not going to be affected in a significant way even when the underlying system's capacity is increased to a very high point, let's say 100. The reason is because the utility function is dependent upon the number of request arrivals and the ratio of the IoT devices served. The increase in the system capacity increases the ratio of the IoT devices served as well as providing paths which have better utility performance in terms of cost. However, if the request arrival rate is fixed then a large increase in the system's capacity to serve requests will not affect the utility performance significantly. The reason is that the availability of the best paths will reach saturation beyond a point, which in this case is 45 for a request arrival rate varying between 15 and 25.

In Fig. 4, we evaluate the effect of I_{max}, as given in (11), on the system's utility performance. When the interference threshold is increased the tolerance for interference caused by a transmission also increases and hence the maximum power which satisfies (11) can be used for transmission. The maximum transmission power is not always used because combined with other transmissions over the same or adjoining channel, it will result into interference in other transmissions and hence data will be lost as also shown by [37]. It can be seen from the figure that when the maximum interference threshold is allowed the system's utility will not move linearly with the increase in the substrate network's capacity. It will increase only nominally when the substrate network capacity is around 30 and afterwards.

Fig. 5 compares the effect of different capacities of the substrate network against the number of service requests which have to be served. For Figs. 5-7, the number of IoT devices is kept at 15. It has to be noted that the cloud controller also serves requests of other networks as well as the IoT

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3022572, IEEE Internet of Things Journal

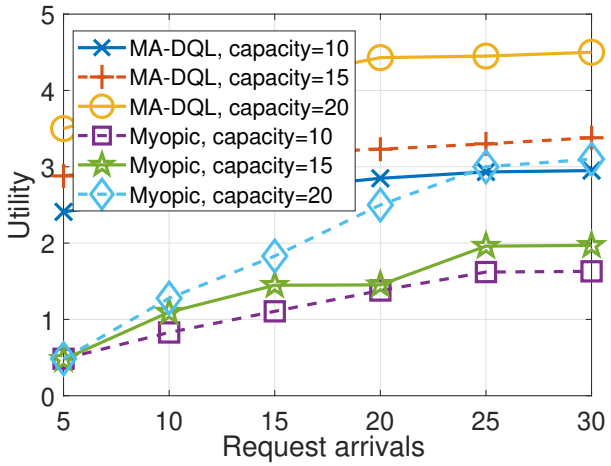

Fig. 5. Effect of number of request arrivals on system's performance.

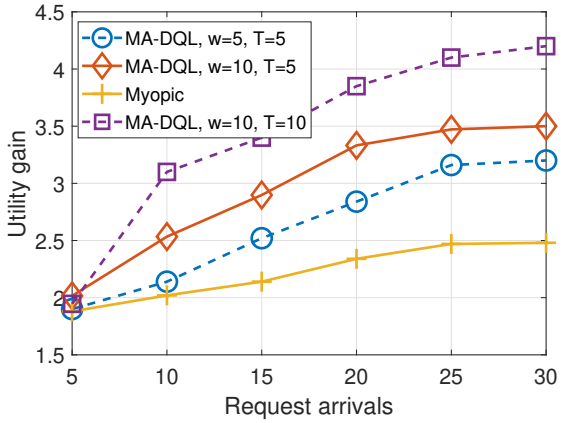

Fig. 6. Effect of different parameters on utility gain.

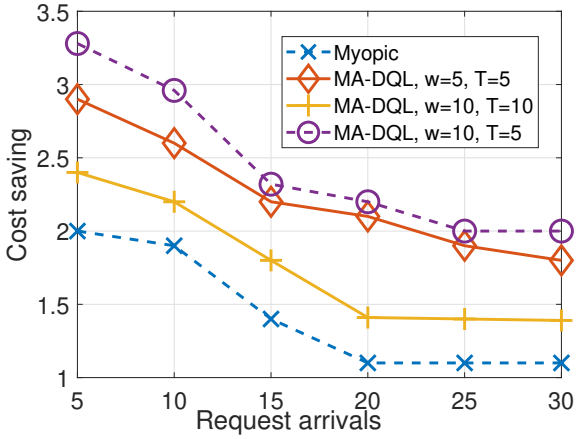

Fig. 7. Effect of different parameters on cost gain.

network considered here. Therefore, as the number of IoT devices increases with capacity of underlying substrate network fixed, performance of the system is affected as explained above because of ρ in (13). The proposed MA-DQL scheme outperforms the myopic scheme even with substrate having the lowest capacity for the proposed scheme while the myopic scheme has the highest capacity available till the number of request arrivals is at or below 20. The performance of all schemes improves with the increase in number of requests because the utility function as used returns summation of return of each request, similarly like that of myopic scheme.

For Fig. 6 and Fig. 7, we check the gain achieved by the proposed MA-DQL scheme and the myopic scheme over the baseline random scheme. The gain represents the ratio of the utility, as in Fig. 6, and cost, as in Fig. 7, of the given schemes over that of the random scheme. The proposed MA-DQL scheme is also compared for different parameter settings of the long-term deep learning weight and the future time horizon. The myopic scheme still achieves gain as it is assumed that the capacity of underlying network is capable to serve all requests. From Fig. 6, it can be seen that the proposed scheme performs better when a higher weight is given to the historical trace of policy changes of the agents, which is represented by w. The placement is done while keeping in consideration the longer future time horizon, which is represented by T. The flattening of utility gain at around the value of 7 in the MA-DQL scheme with optimal parameter setting is because of the saturation reached after the best available paths become to converge in terms of SNR achieved, cost of the placement and the limit of maximum transmission power reached. While in Fig. 7, it can be seen that the cost gain tends to lower as the number of request arrivals increases. The reason is that when the number of requests to be served is low then the paths and links are not overused and thus, the best links available will have low costs so they can be used to achieve higher utility. From both figures it can be seen that the utility gain is achieved through very little drop in the cost saving by the proposed scheme.

Fig. 8 shows the cumulative reward per training episode. It can be seen that with increasing the number of training episodes, the reward increases, validating the effectiveness of the training algorithm. The convergence behavior of the training algorithm can be observed around 2500 episodes where after some fall it goes back to the prior level. It can also be observed that the utility per episode improves continuously with the increase in number of training episodes. The more the training continues, the more the policy reaches the optimal value until the performance in terms of return converges. The fluctuation is the result of the changes in the SNR of different available channels which change over time. The training is done at the cloud controller. On the basis of policy at each agent the reward is returned. The returned reward updates policy of the agent. The policy of cloud controller is updated on the basis of policies of all agents while the training continues till convergence.

Fig. 9 shows the effect of number of IoT devices. The proposed scheme have parameter setting of $T = 5$ and $w = 10$. For a fixed substrate network capacity, increasing the number of IoT devices degrades the performance because the substrate network also serves other networks, hence the ratio of the served IoT devices to the total number of IoT devices, ρ, as given in (13), will decrease affecting the total utility of the IoT network. Nevertheless, it can be seen that the proposed scheme

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3022572, IEEE Internet of Things Journal

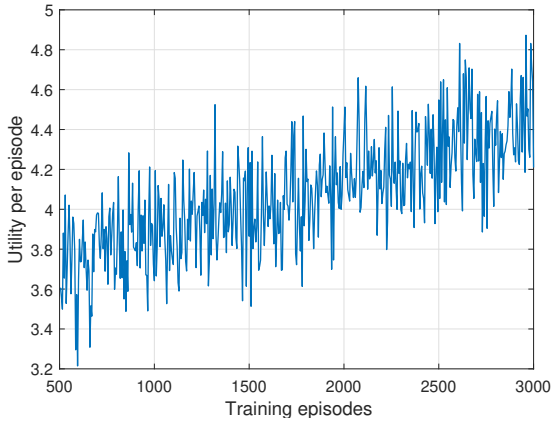

Fig. 8. The return for each training episode.

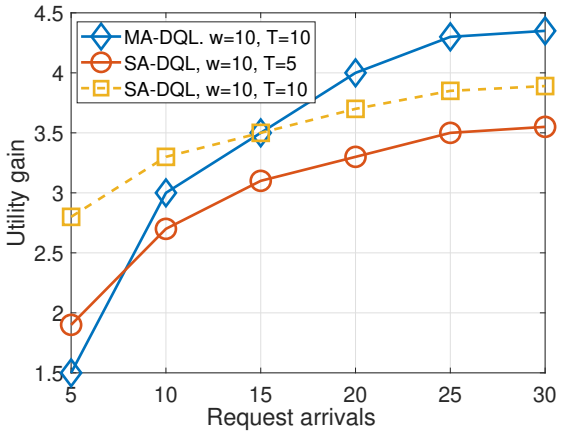

Fig. 10. Comparison with SA-DQL.

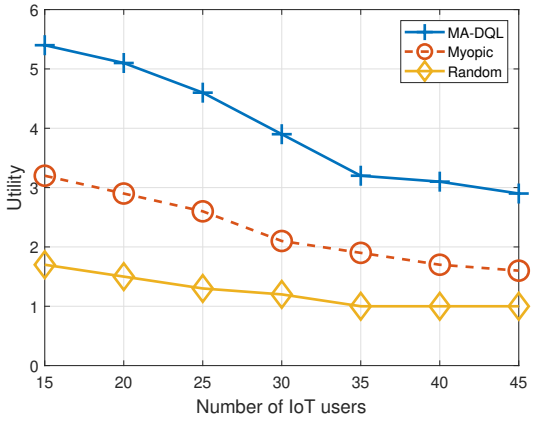

Fig. 9. Performance comparison with respect to number of IoT devices.

outperforms other schemes significantly. When the number of IoT devices is very high then the IoT devices will form multiple IoT networks. The solution to the spectrum access and NFV problem there can be found federated clouds or other such techniques. The problem when large number of IoT devices is present has to be solved at inter-network level. In this paper, the utility maximization problem is considered from the perspective of the individual IoT devices. When there are multiple IoT networks competing or cooperating for the network resources then the whole IoT network becomes an agent and thus the problem has to be solved accordingly. Here the maximum number of IoT devices is taken to be 45 as a heuristic because of the maximum available capacity at which the performance gain saturates as given in figure 3.

Finally in Fig. 10, the traditional DRL scheme, represented by SA-DQL, is compared with the proposed MA-DQL resource allocation scheme. SA-DQL implements the resource allocation algorithms of the MA-DQL but only with a single agent. The central single agent implements all the SFC mapping requests and upgrades the policy through value function on the basis of observations from the environment. As can be seen from Fig. 10, SA-DQL acts well when the request arrival rate is low but its performance degrades when the request arrival rate increases. The reason is that single agent has limited interaction with the environment as a whole for observation, rather than the fine-grained interaction of the multi-agents which takes into consideration the particulars of all the system's substrate resources and parameters. The multi-agent approach gives the whole picture of the environment but it also traces the paths and gives the costs, and the level of congestion at individual nodes and links. Therefore, SA-DQL is unable to avoid congesting some links and nodes thus increasing their costs, as compared to the MA-DQL scheme, when the request arrival rate is high. When the request rate is low to medium, then SA-DQL with parameter settings of w=10 and T=5 performs well than the MA-DQL scheme with optimal settings (as taken from Fig. 6).

VI. Conclusion

In this paper, we have proposed a DRL based solution to the SFC placement problem, which is a significant aspect of network virtualization for IoT networks. The specific performance constraints of IoT networks such as power and interference limits are considered in the problem formulation. The virtualization of the substrate network capacities is exploited to achieve maximum performance gain for IoT networks. The system performance is measured through utility achieved which is a holistic function taking into consideration revenue gain as well cost of both nodes and links. The utility function formulated is a non-convex problem. This problem is solved through a DRL algorithm by considering the SFC placement problem as a distributed problem which is fed into by a central policy. The placement decision for each service request is taken by a DQL agent and decisions of all the agents are combined to take a combined decision. The policy is updated on the basis of reward. Two Q-learning functions are considered. One Q-learning function takes decision in the current decision time slot while the other keeps trace of the long-term trajectory of policy changes. On the basis of the long-term DQL weights, the parameters of the action-taking Q-learning function are updated.

For future work, a more detailed map of the IoT network can be considered. Different spectrum access technologies such as OFDM and cognitive radio tools can also be exploited in

combination with virtualization techniques to investigate effect on the performance gain. A totally distributed system model can also be considered where the problem of SFC forming can be decided in distributed fashion but with coordination among different virtual network agents. A solution which is totally distributed can have applications in other areas such as vehicular and maritime underwater communications.

REFERENCES

[1] C. Liang, F. R. Yu, and X. Zhang, "Information-centric network function virtualization over 5G mobile wireless networks," *IEEE network*, vol. 29, no. 3, pp. 68-74, 2015.

[2] S. Sun, M. Kadoch, L. Gong, and B. Rong "Integrating network function virtualization with SDR and SDN for 4G/5G networks," *IEEE network*, vol. 29, no. 3, pp. 54-59, 2015.

[3] N. C. Luong, D. T. Luong, D.T. Hoang, S. Gong, D. Niyato, P. Wang, Y-C. Liang, and D.T. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 4, pp. 3133-3174, 2019.

[4] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms", in proc. *Advances in Neural Information Processing Systems*, pp. 1008-1014, 2000.

[5] R.S. Sutton and A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 1998.

[6] S. Sun, Y. Ju, and Y. Yamao, "Overlay cognitivee radio OFDM system for 4G cellular networks," *IEEE Wireless Communications*, vol. 20, no. 2, pp. 68–73, 2013.

[7] J. Batalle, J. Ferrer Riera, E. Escalona, and J. A. Garcia-Espin, "On the Implementation of NFV over an OpenFlow Infrastructure: Routing Function Virtualization," in proc. *IEEE SDN for Future Networks and Services*, pp. 1–6, Nov. 2013.

[8] L. R. Battula, "Network security function virtualization(NSFV) towards cloud computing with NFV over openflow infrastructure: challenges and novel approaches," in proc. *IEEE International Conference Advances in Computing, Communications and Informatics*, pp. 1622–28, 2014.

[9] X. Cheng, Y. Wu, G. Min, and A. Y. Zomaya, "Network function virtualization in dynamic networks: A stochastic perspective," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2218-2232, 2018.

[10] P. M. Mohan and M. Gurusamy, "Resilient VNF placement for service chain embedding in diversified 5G network slices, " in proc. *IEEE Globecom*, 2019.

[11] M. Sevil, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in proc. *IEEE 3rd International Conference on Cloud Networking*, 2014.

[12] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating virtualized network functions," *IEEE Transactions on Network and Service Management* , vol. 13, no. 4, pp. 725-739, 2016.

[13] I. Benkacem, T. Taleb, M. Bagaa, and H. Flinck, "Optimal VNFs placement in CDN slicing over multi-cloud environment," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 616-627, 2018.

[14] S. D'Oro, L. Galluccio, S. Palazzo, and G. Schembra, "Exploiting congestion games to achieve distributed service chaining in NFV networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 2, pp. 407-420, 2017.

[15] M. Bagaa, T. Taleb, A. Laghrissi, A. Ksentini, and Hannu Flinck, "Coalitional game for the creation of efficient virtual core network slices in 5G mobile systems," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 469-484, 2018.

[16] Y. Jia, C. Wu, Z. Li, F. Le, and A. Liu, "Online scaling of NFV service chains across geo-distributed datacenters," *IEEE/ACM Transactions on Networking (TON)*, vol. 26, no. 2, pp. 699-710, 2018.

[17] V. Eramo, E. Miucci, M. Ammar, and F. G. Lavacca, "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *IEEE/ACM Transactions on Networking*, vol. 25, no. 4, pp. 2008-2025, 2017.

[18] L. Gupta, M. Samaka, R. Jain, A. Erbad, D. Bhamare, and C. Metz, "COLAP: A predictive framework for service function chain placement in a multi-cloud environment," in proc. *IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 1-9, 2017.

[19] S. Chinchali, P. Hu, T. Chu, M. Sharma, M. Bansal, R. Mishra, M. Pavone, and S. Katti, "Cellular network traffic scheduling with deep reinforcement learning," in proc. *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[20] H. Ying, Z. Zheng, F. R. Yu, N. Zhao, H. Yin, V. C. M. Leung, and Y. Zhang, "Deep-reinforcement-learning-based optimization for cache-enabled opportunistic interference alignment wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11 , pp. 10433-10445, 2017.

[21] Y. He, F. R. Yu, N. Zhao, and H. Yin, "Secure social networks in 5G systems with mobile edge computing, caching, and device-to-device communications," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 103-109, 2018.

[22] L. Liang, H. Ye, and G. Ye Li, "Spectrum Sharing in Vehicular Networks Based on Multi-Agent Reinforcement Learning," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2282-2292, 2019.

[23] H. Liang, Z. Zhang, J. Zhang, Q. Li, S. Zhou, and L. Zhao, "A novel adaptive resource allocation model based on SMDP and reinforcement learning algorithm in vehicular cloud system," *IEEE Transactions on Vehicular Technology*, vol. 68, No. 10, pp. 10018-10029, 2019.

[24] N. Ye, X-M. Li, H. Yu, L. Zhao, W. Liu, and X. Hou, "DeepNOMA: A unified framework for NOMA using deep multi-task learning," *IEEE Transactions on Wireless Communications*, vol. 19, no. 4, pp. 2208-2225, 2020.

[25] H-H. Chang, H. Song, Y. Yi, J. Zhang, H. He, and L. Liu, "Distributive Dynamic Spectrum Access Through Deep Reinforcement Learning: A Reservoir Computing-Based Approach," *IEEE Internet of Things Journal*, vol. 6, no.

2, pp. 1938-1948, 2018.

[26] F. Wang, J. Gao, M-S. Li, and L. Zhao, "Autonomous PEV charging scheduling using deep-Q network and Dyna-Q reinforcement learning," *IEEE Transactions on Vehicular Technology*, to appear, 2020.

[27] B. Zheng, P. He, L. Zhao, and H-W. Li, "A hybrid machine learning model for range estimation of electrical vehicles", in proc. *IEEE Globecom*, 2016.

[28] M. Li, L. Zhao, and H. Liang, "An SMDP-based prioritized channel allocation scheme in cognitive enabled vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 66, No 9, pp. 7925-7933, 2017.

[29] M. S. Li, J. Gao, L. Zhao, and X. Shen, "Deep reinforcement learning for collaborative edge computing in vehicular networks," *IEEE Transaction on Cognitive Communications and Networking*, early access, 2020.

[30] T. Park and W. Saad, "Distributed learning for low latency machine type communication in a massive internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5562-5576, 2019.

[31] Q. Li, L. Zhao, J. Gao, H. Liang, L. Zhao, and X. Tang, "SMDP-based coordinated virtual machine allocations in cloud-fog computing systems," *IEEE Internet of Things Journal* , vol. 5, No. 3, pp. 1977-1988, 2018.

[32] F. Samie, L. Bauer, and J. Henkel, "From cloud down to things: An overview of machine learning in internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4921-4934, 2019.

[33] M. Zolanvari, M. A. Teixeira, L. Gupta, K. M. Khan and R. Jain, "Machine learning-based network vulnerability analysis of industrial internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6822-6834, 2019.

[34] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Communication Surveys and Tutorials*, vol. 21, no. 3, pp. 2671-2701, 2019.

[35] J. Forester, N. Nardelli, G. Farquhar, T. Afouras, P. H. S. Torr, P. Kohli, and S. Whiteson, "Stabilising experience replay for deep multi-agent reinforcement learning," in proc. *International Conference on Machine Learning (ICML)*, pp. 1146-1155, 2017.

[36] X. Cheng, Y. Wu, G. Min, and A. Y. Zomaya, "Network function virtualization in dynamic networks: A stochastic perspective," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2218-2232, 2018.

[37] H. A. Shah and I. Koo, "A novel physical layer security scheme in OFDM-based cognitive radio networks," *IEEE Access*, vol. 6, pp. 29486-29498, 2018.

Hurmat Ali Shah received his B.Sc and M.Sc degrees from University of Engineering and Technology, Peshawar in 2010 and 2013, respectively. He completed his Ph.D. degree at the School of Electrical and Computer Engineering, University of Ulsan, Korea in 2019. He was Postdoc Fellow for a brief period at Inha University, Korea. He is with School of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, Canada as Postdoc Fellow since September 2019. In his Ph.D., he focused on spectrum sensing, resource allocation and physical layer security for cognitive radios. His research interests are in intelligent IoTs, massive IoTs, next generation wireless communications particularly 6G, network virtualization, space-terrestrial integrated networks, deep learning and wireless AI.

Lian Zhao (S'99-M'03-SM'06) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Canada, in 2002. She joined the Department of Electrical, Computer, Biomedical Engineering at Ryerson University, Toronto, Canada, in 2003 and has been a Professor in 2014. Her research interests are in the areas of wireless communications, resource management, mobile edge computing, caching and communications, and vehicular ad-hoc networks.

She has been selected as an IEEE Communication Society (ComSoc) Distinguished Lecturer (DL) for 2020 and 2021; received the Best Land Transportation Paper Award from IEEE Vehicular Technology Society in 2016, Top 15 Editor Award in 2015 for IEEE Transaction on Vehicular Technology, Best Paper Award from the 2013 International Conference on Wireless Communications and Signal Processing (WCSP), and the Canada Foundation for Innovation (CFI) New Opportunity Research Award in 2005.

She has been serving as an Editor for IEEE Transactions on Vehicular Technology, IEEE Transactions on Wireless Communications, and IEEE Internet of Things Journal. She served as a co-Chair of Wireless Communication Symposium for IEEE Globecom 2020 and IEEE ICC 2018; Local Arrangement co-Chair for IEEE VTC Fall 2017 and IEEE Infocom 2014; co-Chair of Communication Theory Symposium for IEEE Globecom 2013. She served as a committee member for NSERC (Natural Science and Engineering Research Council of Canada) Discovery Grants Evaluation Group for Electrical and Computer Engineering 2015 to 2018. She is a licensed Professional Engineer in the Province of Ontario, a senior member of the IEEE Communication Society and Vehicular Society.