Optimising resource allocation for virtual network functions in SDN/NFV-enabled MEC networks

Nahida Kiran | Xuanlin Liu | Sihua Wang | Changchuan Yin

Abstract
Network function virtualisation (NFV), software defined networks (SDNs), and mobile edge computing (MEC) are emerging as core technologies to satisfy increasing number of users’ demands in 5G and beyond wireless networks. SDN provides clean separation of the control plane from the data plane while NFV enables the flexible and on-the-fly creation and placement of virtual network functions (VNFs) and are able to be executed within the various locations of a distributed system. In this paper, VNF placement and resource allocation (VNFPRA) problem is considered which involves placing VNFs optimally in distributed NFV-enabled MEC nodes and assigning MEC resources efficiently to these VNFs to satisfy users’ requests in the network. Current solutions to this problem are slow and cannot handle real-time requests. To this end, an SDN-NFV infrastructure is proposed to tackle the VNFPRA problem in wireless MEC networks. Our aim is to minimise the overall placement and resource cost and also to minimise the total number of VNF migrations. A genetic based heuristic algorithm is proposed. The superior performance of the proposed solution is confirmed in comparison with four existing algorithms, i.e. resource utilisation-single objective evolutionary algorithm (RU-SOEA), genetic non-bandwidth link allocation algorithm (GA-NBA), random-fit placement algorithm (RFPA), and first-fit placement algorithm (FFPA). The results demonstrate that a coordinated placement of VNFs in SDN/NFV enabled MEC networks can satisfy the objective of overall reduced cost. Simulation results also reveal that the proposed scheme approximates well with the optimal solution returned by Gurobi and also achieves reduction on overall cost compared to other methods.

1 | INTRODUCTION
Mobile edge computing (MEC) technology has been introduced to reduce resource demands and user experienced delays by moving computing, networking, and storage capabilities from the core cloud to the mobile edge network. By creating a cloud-like environment which has computing, storage, and networking functions on the base stations that are adjacent to the users, users can place requested contents or virtual network functions (VNFs) on adjacent clouds. In this way overall distributed resources can be efficiently managed and delays can be reduced [1]. This MEC-enabled architecture that enables cloud computing capabilities at the edge of the access network is widely accepted by the literature in order to serve the increased requirements of the 5G services [2].

MEC implementation can also potentially leverage network function virtualisation (NFV) platforms, which eliminates the restrictions applied by the legacy LTE networks, e.g. the fixed placement of the network functionalities [3]. NFV enables the flexible and on-the-fly creation and placement of VNFs, satisfying the diverse application requirements and optimising the management of the heterogeneous (e.g. network, computational, and storage) resources. Hence, application and network functionalities are handled as VNFs and managed by an NFV orchestrator (NFVO) [4], and are
able to be executed within the various locations of a distributed system and, in our case, in the NFV-enabled MEC nodes.

In cloud environments, cloud providers propose their infrastructure, which is composed of a single or multiple data centers (DCs) containing computing and networking equipments. NFV, as an evolving technology, follows the similar concept and takes it to a higher level by leveraging virtualisation technology to virtualise network functions; yet, it has brought new challenges to network operators such as (i) finding the most appropriate placement of these VNFs into the MEC environment (ii) optimal resource allocation to these VNFs in NFV-enabled MEC nodes. Finding the best placement of a single VNF may not be as challenging as allocating resources (i.e. CPU, memory, storage, and bandwidth) for a set of VNFs in MEC nodes. Specifically, when a service function chain (SFC) consists of one VNF, it is a matter of finding a MEC node which has enough resources to minimise the costs. However, when an SFC comprises of multiple VNFs, the placement problem turns into even more difficult since VNFs are limited to some certain order as well as link and VNF delays. SFC [5] is a capability that uses software defined network (SDN) capabilities to create a service chain of connected network services (e.g. firewall, network address translation) and connect them in a virtual chain.

MEC nodes are usually distributed in different locations. The energy price for each MEC node varies depending on its location; hence, the operational cost (i.e. cost of running VNFs) is not the same. Moreover, from service providers’ (SPs’) perspective, they aim at provisioning service requests and increase their total revenue. This can only be accomplished by allocating resources for VNFs in MEC nodes such that the service request is provisioned at a reasonable cost to the end-users.

The combination of SDN, NFV, and MEC technologies are widely considered to be essential in the development of 5G wireless networks. MEC, together with SDN and NFV will play to address the challenges that 5G aims to undertake. We consider MEC nodes to be any NFV-enabled network nodes at which MEC services can be hosted with allocated VNFs. MEC involves placing computing power in new locations. This computation may involve network functionality and end-user applications. None of the studies on the integration of MEC and SDN explores the use of NFV to deploy specific VNFs to offer final user services, such as VoIP, video streaming, or web browsing. In [6], a recent placement scheme applicable to MEC in NFV environment has been proposed to minimise access latency and maximise service availability for ultra-reliable low-latency communications (uRLLC) services. The authors proposed a genetic algorithm and compared with CPLEX implementation. Their approach did not consider failure recovery in case of physical machine (PM) breakdowns, especially at the edge and for critical services, such as the connected car. Although the work proposed in [7] considered NFV resource allocation (NFV-RA) problem, it did not consider transmission latency. Also in the aforementioned latency-aware NFV-RA works, the hierarchical network resource allocation has not been considered yet. Hence, their solutions are not fit for the hierarchical network that contains both MEC nodes and DCs. Thus, these solutions may not be appropriate to implement in dynamic network resource allocation systems with a wide range of resource demands and services.

To benefit from the advantages of SDN, NFV, and MEC, VNFs need to be provisioned with sufficient resources in edge servers without impacting network quality of service (QoS). This paper proposes an SDN/NFV-enabled MEC model for optimal virtual network function placement and resource allocation (VNFPRA) with the goal to minimise both the overall deployment cost and resource cost. A single VNF might be enough to handle certain amount of requests before it becomes overloaded. But, when there is a sudden spike of requests, one VNF may not be enough to process all the received requests. In such case, we investigate the impact of replicating VNFs from neighbouring MEC nodes or from remote cloud with the help of centralised SDN controller. When receiving a VNF request, NFV-enabled MEC nodes either process the request and interact directly with the user if the requested VNF is available in it, if not, SDN controller decides whether the requested VNF should be forwarded to a neighbouring MEC node. Whenever a new VNF request from user equipment (UE) enters, the first NFV-enabled MEC node receives a description packet, which is forwarded to the SDN controller. This packet contains the description of the VNF, the bandwidth and the destination node that the UE requires. If the required VNF is not in the NFV-enabled MEC node, the controller finds the optimal route and resource allocation along the route such as bandwidth for the transmission of VNF.

The main contributions in this work are as follows: (i) We propose a novel SDN/NFV-enabled MEC infrastructure for low-cost MEC resource allocation framework; (ii) We formulate the VNFPRA problem as a mixed integer program (MIP) problem by taking into account the possibility of NFV-enabled MEC service instantiations and propose a genetic based heuristic solution which is theoretically proven to be sound and complete. Simulation results reveal that the proposed scheme approximates well with our optimal solution returned by Gurobi and also achieves reduction on overall cost compared to other methods.

The paper is organised as follows. Section 2 addresses the related work. Section 3 describes the proposed system model. Section 4 addresses the problem formulation. In Section 5, we present details of cost-aware VNFPRA for NFV-enabled MEC. Section 6 describes the proposed genetic algorithm for VNF migration (GA-VNFM). In Section 7 and 8, we show our simulation results and conclude the paper, respectively.

2 RELATED WORK

Recently, with the advent of SDN [8] and NFV [9], the concept of NFV-enabled MEC emerged whereby services can be hosted at any conventional network node that has virtualised resources (e.g. switches, routers, access points etc.). Within this new model, the on-demand network resource allocation in the form of VNFs becomes achievable. This has
the advantage of reducing investments in specialised hardware, while allowing flexible scaling of the capacity of the functions [10]. Additionally, routers and switches can use their unused computational resources to implement these VNFs [11].

NFV has become an essential technology for delivering network services to the end users. The network services accessed by users are becoming more dynamic and diverse with the rapid growth of user data traffic [12]. NFV can also be interconnected with other technologies such as internet of things (IoT), cloud computing, and MEC. It can also provide the necessary flexibility to offer diverse and dynamic network services through the service chaining (SC) model. SDN and NFV are complementary in nature. They have intersected objectives of substituting proprietary and closed network elements (switches) with commodity hardware that can be controlled by standardised software [13].

Existing works in the literature consider delay/latency perceived in the overall communication during the VNF placement and SC deployment. This overall communication delay is expressed as the propagation delay. In such scenarios, the delay is either considered explicitly in the objective function with the ultimate goal to be minimised [14] or it is taken as a constraint satisfying a maximum tolerable delay [15]. The delay minimisation has been studied in a multi-cloud environment. Authors in [16] considered only the distance between the cloud provider and the end user. In this case, the end user may be able to select among a set of available SPs to find the best offer between the latency perceived and deployment cost. On the other hand, the authors in [17] used a random cloud selection approach which combined with a predictive model with the aim to keep the delay below a threshold.

Existing techniques also consider load balancing features and deployment costs when placing VNFs and SCs. The goal is to increase the availability of the resources, providing essential scalability, and increasing the number of SCs in cloud infrastructure to maximise the revenue of the SP. In terms of load balancing there is a high interest in resource usage and path optimisation [18]. Deployment cost is usually expressed as the communication and computational resources that a SC needs, in order to be provisioned. Most of the existing procedures try to consolidate as many of the VNFs as possible in the same server [19], or rack in case of a data center. The authors in [20] proposed a microeconomic approach inspired from producer-consumer theory to model the interactions between virtual machines (VMs) and physical resources. Rankothge et al. in [21] provided a multi-objective genetic algorithm (GA)-inspired approach to predict dynamic resource utilisation and energy consumption in data centers. The results showed that the proposed placement algorithm not only reduces the memory and CPU usage, but also reduces the energy consumption of the data center as well.

Management and resource orchestration are essential for the NFV-MEC deployments. One of the first efforts to define a double-tier NFV-MEC architecture associated to the European Telecommunications Standards Institute (ETSI) NFV management and orchestration (MANO) specifications was proposed in [22] by improving the resource orchestration for MEC functions and IoT deployments (e.g. Cloud RAN). The MEC architecture and open research issues were also surveyed in [23] by emphasising the necessity to extend the NFV ecosystem at the edge and far edge scenarios to deal with 5G and IoT network deployments. Yi et al. in [24] addressed SFC provisioning problem and designed two meta-heuristic algorithms to make a trade-off between optimal network performance and time complexity. Luizelli et al. in [25] presented a meta-heuristic algorithm using variable neighbourhood search (VNS) to minimise allocated resources, while meeting network flow requirements.

The resources in MEC nodes are restricted compared to that in remote data centers [26]. By combining NFV and MEC technologies these resources can be utilised efficiently because network operators can assign resources for VNFs according to the MEC resource usage requirement. Cost reduction is one of the foremost goals of NFV. Most latency-aware NFV-RA methodologies emphasise on minimising cost without sacrificing service latency requirements. Authors in [27] developed a dynamic cost minimisation model by considering delay into the cost objective in order to jointly optimise three steps in NFV-RA, including VNFs-chain composition, VNFs-scheduling, and VNFs-forward graph embedding. In [28] a mixed integer linear programming (MILP) model with latency constraints is proposed for the cost minimisation in metro core networks. Another model is designed in [29] to minimise the same cost in the previous model to upgrade the solutions by adding virtualisation latency into constraints. A low latency VNF placement algorithm is designed in data center networks to reduce service latency [30]. In [31] a mixed integer quadratically constrained programming model is developed to minimise resource consumption for VNF placement while producing specific latency.

Only few works have addressed the problem of VNF placement in a mixed environment that includes edge and central clouds. Authors in [32][33] introduced an efficient content delivery network as a service platform and VNF placement and proposed optimisation strategies over an edge-central carrier cloud infrastructure by taking into account QoS requirements. Their main design goal is to provide an optimal tradeoff solution between the cost efficiency and quality of experience (QoE) to optimise resource utilisation for an appropriate set of VNFs. Usually, VNFs are placed either in physical machines within central cloud infrastructures only [34], or in MEC servers [35].

Although there are some methodologies for managing NFV and edge resources, there is still a chain of open issues and further gaps with regard to their well-organised integration and edge infrastructure management in NFV ecosystems. In order to address these gaps, we have taken a mixed environment which is consistent with current state-of-the-art SDN and NFV architectures. This paper addresses the interaction of the three technologies (SDN, NFV, and MEC) and builds a new framework for optimal VNF placement and resource allocation to minimise the overall deployment and resource cost. It also focuses on how to minimise the total number of VNF migrations and maximise the total utility of the MEC system.
3 | SYSTEM MODEL

The system model consists of a control plane and a data plane as shown in Figure 1. We assume that the network is made of \( N \) small base stations (BS) and each BS is attached with an NFV-enabled MEC node connected directly to each other that are capable of being controlled by an SDN controller. As in SDN-enabled technologies, only the centralised controller has the full view of the global network topology and it is a responsive agent, which will react based on the requests that it receives. We also assume that the SDN controller can communicate with the MANO layer of the NFV. As stated in [36], VNF manager is the critical component of the NFV architecture, also called MANO. In this system, we consider each MEC server has VNFs which are managed by a VNF manager located in the core network. VNF manager can start and stop VNFs based on local demand. By doing this, we can reduce deployment costs, power consumption as well as improve service readiness to operate a network component.

The controller is assumed to have sufficient resources to run all possible VNFs to satisfy user’s demand. By using appropriate cost function, the controller finds the best possible route of the incoming requests. It may happen that for the chosen route there is no feasible VNF placement found or there is no possible route to migrate VNFs. In such cases the SDN controller signals the NFV MANO layer to migrate the required VNF containers appropriately to the required NFV-enabled MEC nodes. The controller also pushes some flow rules for the request in the respective NFV-enabled MEC nodes. The rules are added in sequence starting from the destination node and the last rule is added at the source node in the path. After this, the incoming requests are switched across the data plane.

In order to effectively manage the network, there are two main decisions the controller needs to make: (i) VNF placement, i.e. how to allocate VNFs and assign optimal set of resources by considering the resource availability in MEC nodes. The cost incurs in deploying VNF services and delivering them to the end-users must be minimised for cost-efficiency, and (ii) Resource allocation, i.e. how the computational capabilities available at NFV-enabled MEC nodes are allocated to the VNFs. We use \( G(V, L) \) to denote the network with \( N \) users, where \( V = \{1, 2, \ldots, V\} \) is the set of NFV-enabled MEC nodes and \( L \) is the set of links between the NFV-enabled MEC nodes. In this paper, we assume that any end user can be within the coverage of one MEC node. Total demands that each VNF may satisfy are limited. For simplicity, initially we assume that one end user sends only one request to the MEC node and one VNF can handle \( K \) number of requests. Requests sent from users should be first received by the NFV-enabled MEC node \( j \), where \( j \in V \) and it interacts directly with the user if the requested VNF is available in it. But, when there is a sudden spike of requests and \( j \)-th MEC node cannot process these requests because of capacity limit, it can replicate required VNFs from the neighbouring MEC nodes or from remote cloud by VNF manager. Each MEC node \( j \) is attributed with a vector of available resource capacities \( W \) (e.g. CPU, Memory). We consider that each NFV-enabled MEC node \( j \), has maximum computing capacity \( C_j^{\max} \) and maximum storage capacity \( S_j^{\max} \), which is the maximum capacity to compute and store VNFs. We denote the set of VNFs of different types as \( F = \{1, 2, \ldots, F\} \). Each VNF \( i \in F \) on MEC node, if activated, consumes some physical resources (i.e. DRAM memory, CPU cycles) and has a predetermined resource requirement of computing capacity \( c_i \) (CPU cycles) and storage capacity \( s_i \) (memory) to process it.

We summarise the notation and symbols used in this paper in Table 1 unless otherwise specified.

4 | PROBLEM FORMULATIONS

Initially we migrate the VNFs from remote cloud and store them in distributed MEC nodes. This migration requires placement cost. After that we replicate the VNFs from neighbouring MEC nodes if needed rather than replicating from remote cloud. This replication may cause link usage cost (replication cost) to replicate VNFs from adjacent MEC nodes.

When a MEC node does not have the VNF application requested by some users, we should determine whether the application VNF should be replicated from another MEC node or not. Let \( l(j,k) \) denote the link between MEC node \( j \) and MEC node \( k \) and every link \( l \in L \) has capacity \( c_l \) and per unit link usage cost of \( \epsilon_{l,j} \). We denote by \( y_{j,k,l} \) the decision variable stating whether the link is used to replicate VNF \( i \) from NFV-enabled MEC node \( k \) to NFV-enabled MEC node \( j \). Bandwidth demand for replicating VNF \( i \) over the link \( l \) is...
TABLE 1 Summary of notations and symbols

| Notation | Representation of the symbol |
|----------|-------------------------------|
| V        | Set of NFV-enabled MEC nodes $V = \{1, 2, \ldots, \nu\}$ |
| L        | Set of links between the NFV-enabled MEC nodes |
| N        | Number of users |
| P        | Set of VNFs of different types $P = \{1, 2, \ldots, \nu\}$ |
| W        | Resource of MEC node (e.g. CPU, memory) |
| $C_j^{\text{max}}$ | Maximum computing capacity of MEC node $j$ |
| $S_j^{\text{max}}$ | Maximum storage capacity of MEC node $j$ |
| $i'$ | Predetermined computing resource requirement to process VNF $i$ |
| $i''$ | Predetermined storage resource requirement to process VNF $i$ |
| $a_{ji}$ | Allocation index if VNF $i$ is allocated in MEC node $j$ |
| $f_{ji \rightarrow k,i}$ | Decision variable if VNF $i$ is replicated over the link $l$ from MEC node $k$ to MEC node $j$ |
| $C_i'$ | Cost of placing VNF $i$ at NFV-enabled MEC node $j$ |
| $C_i''$ | Cost of resource $W$ (e.g. CPU, Memory) at MEC node $j$ |
| $u_{ji \rightarrow k,i}$ | Link usage cost for replicating VNF $i$ over link $l$ |
| $R_j'$ | Number of VNF requests for VNF $i$ from MEC node $j$ |
| $B_{ji \rightarrow k,i}$ | Bandwidth allocated to VNF $i$ over the link $l$ from MEC node $k$ to MEC node $j$ |
| $c_l$ | Link capacity |

denoted as $B_{ji \rightarrow k,i}$. We have

$$f_{ji \rightarrow k,i} = \begin{cases} 1, & \text{if VNF } i \text{ is replicated over the link } l; \\ 0, & \text{otherwise}. \end{cases}$$

Based on above assumptions, we address the VNF placement and resource allocation problem subject to various constraints such as the VNF requirement of each request and the resources available at each NFV-enabled MEC node to deploy the VNFs. We need to place VNFs at the NFV-enabled MEC node such that the overall resource cost is minimised. We consider three different types of costs incurred as follows:

### 4.1 VNF placement/allocation cost

This is the cost incurred if any VNF is placed/hosted in NFV-enabled MEC node. The optimal VNF placement involves determining how many VNF instances are needed to meet the end user's demand and on which servers they should be deployed. This cost can be defined as

$$\sum_{j \in P} \sum_{i \in V} a_{ji} C_i', \quad (1)$$

where $C_i'$ is the cost of placing VNF $i$ at NFV-enabled MEC node $j$. $a_{ji}$ is the allocation variable where

$$a_{ji} = \begin{cases} 1, & \text{if VNF } i \text{ is allocated in MEC node } j; \\ 0, & \text{otherwise}. \end{cases}$$

### 4.2 Resource/computation cost

This is the cost per utilised processor and memory of the NFV-enabled MEC node. This cost can be expressed as

$$\sum_{j \in V} \sum_{i \in P} a_{ji} R_j' C_i' W, \quad (2)$$

where $R_j'$ is the number of VNF requests for VNF $i$ from MEC node $j$, which is known or given. $C_i' W$ is the cost of resource $W$ (e.g. CPU, memory) at MEC node $j$.

### 4.3 Link usage/replication cost

This is the cost per utilised capacity of the virtual links. This can be defined as

$$\sum_{j \in V} \sum_{i \in P} \sum_{l \in \mathcal{L}} a_{ji} R_j' C_i' u_{ji \rightarrow k,i} B_{ji \rightarrow k,i}', \quad (3)$$

subject to

$$\sum_{j=1}^{\nu} a_{ji} \leq C_j^{\text{max}}, \quad \forall j \in V, \quad (4a)$$

$$\sum_{i=1}^{\nu} a_{ji} \leq S_j^{\text{max}}, \quad \forall j \in V, \quad (4b)$$

$$\sum_{i=1}^{\nu} a_{ji} \geq 1, \quad \forall i \in P, \quad (4d)$$

where $u_{ji \rightarrow k,i}'$ is the link usage cost for replicating VNF $i$ from MEC node $k$ to MEC node $j$ over the link $l$.

The objective function minimises the overall resource cost. These are the placement cost incurred if required VNF is placed/hosted in NFV-enabled MEC node, cost per utilised memory and CPU of the NFV-enabled MEC node, and the cost per utilised capacity of the links, which is defined as

$$\min_{a_{ji} \geq 0} \sum_{j \in V} \sum_{i \in P} a_{ji} C_i' W, \quad (4a)$$

subject to

$$\sum_{j=1}^{\nu} a_{ji} \leq C_j^{\text{max}}, \quad \forall j \in V, \quad (4a)$$

$$\sum_{i=1}^{\nu} a_{ji} \leq S_j^{\text{max}}, \quad \forall j \in V, \quad (4b)$$

$$\sum_{i=1}^{\nu} a_{ji} \geq 1, \quad \forall i \in P, \quad (4d)$$
5.1 Chromosome representation

In our problem, each solution of VNF placement is represented by a chromosome. A chromosome signifies a possible VNF placement solution. We implement a matrix representation for a chromosome. We use a binary encoding method to represent the assignment VNF placement solution. The placement matrix solution \( \{a_{ij}\} \) is a \( m \times n \) matrix if there are \( m \) VNFs and \( n \) NFV-enabled MEC nodes. If the \( i \)th VNF is deployed on the \( j \)th MEC node, the entry \( a_{ij} \) is set to 1 to guarantee that the VNF \( i \) is deployed only once on the MEC node \( j \) and other elements in the \( i \)th row must be equal to 0.

5.2 Population initialising

We generate initial population \( I_p \) which grows in each generation of genetic algorithm. First, we generate the accessible set \( \mathcal{S}_j \) for each VNF \( i \) in MEC node \( j \). \( \mathcal{S}_j \) represents the set of all MEC nodes from where VNF \( i \) can be migrated. Then, the MEC node calculates the accessible probability \( P_{\mathcal{S}_j} \) for each VNF \( i \) in MEC node \( j \), which is used to identify the target MEC node where the VNF can be migrated. MEC node calculates the accessible probability for each VNF \( i \), that represents how likely VNF can be migrated to another MEC node. First of all, we generate the accessible set for each VNF and the previous best solution to get the initial number of populations. Then we calculate the accessible probability for each VNF \( i \). If previous best solution is available then we use it as a seed with the accessible probability. The MEC node with higher accessible probability has higher
chances for getting selected. We calculate \( P^S_{i,k} \) by the following equation:

\[
P^S_{i,k} = \frac{\bar{j}^S_{i,k}}{\max_{j,k} \bar{j}^S_{i,k}},
\]

where \( \bar{j}^S_{i,k} \) is the variable which represents if VNF \( i \) can be migrated from MEC node \( j \) to MEC node \( k \).

### 5.3 Fitness function

The fitness function evaluates how well an individual matches the objective. We should obtain an optimal placement solution considering our main objective, minimising the overall resource cost. In order to be consistent with the final objective, the fitness of chromosome is calculated based on overall cost. The fitness function evaluates how a chromosome matches our main objective. In our Algorithm 1, fitness is defined as

\[
\text{fitness}(\alpha_i) = 1 / \text{objective function}, \tag{6}
\]

where \( \alpha_i \) is an individual in the population and the objective function is defined in (4). Considering the objective function in (4), we calculate the placement and resource cost of all the VNFs mapped on each MEC node according to the matrix placement solution. Then, we get the best fitness value for this solution. Moreover, the individuals with higher fitness value should be nominated as parents to generate new populations for obtaining better chromosomes for next generation.

Algorithm 1 solves our optimisation problem by generating a group of random chromosomes \( a_a, a_b, a_c, \) and \( a_d \) and evaluate each chromosome by (4)–(4f). Two best chromosomes \( a_a \) and \( a_b \) whose resource cost in (4) is lower are selected to produce another group of random chromosomes by mutation and recombination. This technique lasts until the acquired result reaches convergence or until the number of generations reaches the maximum number of generations.

### 6 GENETIC ALGORITHM FOR VNF MIGRATION (GA-VNFM)

Next key challenge is to find an optimal MEC node for VNF migration (VNFM). The goal of VNFM is to minimise the total number of VNF migrations and to maximise the total utility of the MEC system.

In VNFM migration, we have to assign the VNFs to new MEC nodes so that the total utility of the MEC system is maximised. This is a NP-hard problem and thus, there is no algorithm that can provide an assured optimal solution in polynomial time. For this reason, we propose a GA-VNFM that can ensure near-optimal solution by reducing the total number of VNF migrations. The proposed Algorithm 2 allows us to identify the candidate MEC node for VNFM. The SDN controller makes decision on possible MEC nodes based on the load of the target MEC node. If the load of MEC node is not considered, then the total number of VNF migration will be increased. In the following subsection we present different parts of GA-VNFM.

#### 6.1 Chromosome representation

In our problem, each solution of VNFM is represented by a set of chromosomes denoted as \( C = \{C_1, C_2, \ldots, C_N\} \) of length \( N \). Each bit in a chromosome represents a VNF and the value of this bit is used to identify the target MEC node where the VNF
can be migrated. Next, we use Algorithm 2 to generate the initial population and then we have to select two individuals for the crossover after the calculation of fitness value. After selecting two individuals, we perform mutation operation on the new offspring. Then, we replace the individuals i.e. \( y_a \) and \( y_b \) with the lowest fitness by the offsprings \( y_a \) and \( y_b \) with the highest fitness value.

### 6.2 Fitness function

For VNF migration in our model we can use the following fitness function to evaluate all the individuals in a population:

\[
F_n = \sum_{i \in P} U_{n,i}^j. \tag{7}
\]

Here, \( F_n \) is the fitness value for \( n \)th individual and \( U_{n,i}^j \) is the utility of VNF \( i \) in MEC node \( j \), which is defined by the following equation:

\[
U_{n,i}^j = P_{n,i}^j + T_{n,i}^j + L_{n,i}^j, \tag{8}
\]

where \( P_{n,i}^j \) is the minimum normalised task processing time of VNF \( i \) for the \( n \)-th individual when VNF \( i \) is not migrated from MEC node \( j \) to MEC node \( k \). When VNF \( i \) is not migrated, it means the selected MEC node doesn’t have the required VNF to migrate. In such case the SDN controller signals the NFV MANO layer to migrate the required VNF appropriately to the required NFV-enabled MEC node. We calculate \( P_{n,i}^j \) by this equation

\[
P_{n,i}^j = \min_{k \in S_n^j} \left\{ \frac{P_{i,k}^j}{\max_{k \in S_n^j} P_{i,k}^j} \right\}, \tag{9}
\]

where \( S_n^j \) is the accessible set for each VNF \( i \) in MEC node \( j \) and \( I_y \) is the set of individuals.

Moreover, \( T_{n,i}^j \) is the minimum normalised transfer time of VNF \( i \) for the \( n \)th individual from MEC node \( j \) to \( k \). Using the below equation we calculate minimum VNF transfer time as

\[
T_{n,i}^j = \min_{k \in S_n^j} \left\{ \frac{T_{n,i,k}^j}{\max_{k \in S_n^j} T_{n,i,k}^j} \right\}, \tag{10}
\]

where \( L_{n,i}^j \) is the minimum normalised VNF load association degree between MEC node \( j \) to \( k \), which helps the system to predict the load of the MEC node. Minimum normalised VNF load association can be calculated as

\[
L_{n,i}^j = \min_{k \in S_n^j} \left\{ \frac{L_{n,i,k}^j}{\max_{k \in S_n^j} L_{n,i,k}^j} \right\}. \tag{11}
\]

### 6.3 Genetic operators

Once fitness value for each individual is calculated, next we have to choose two individuals for the crossover operation. At the start of the selection process, the selection probability \( P_n^j \) for each individual \( n \) is calculated using this equation:

\[
P_n^j = \frac{F_n}{\sum_{i \in I_y} F_i}. \tag{12}
\]

Here, an individual with greater \( P_n^j \) has greater chance to be selected. Next, a mutation operation is performed on the new offspring with the mutation probability \( P_m^j \) and then we reduce the offspring which are not compatible with VNF’s accessible set. To retain the stochastic property of VNFPRA problem, we replace an individual from \( I_y \) with the offsprings such that the offspring fitness values are higher than the individual fitness values. The whole procedure is listed in Algorithm 2.

### 7 PERFORMANCE EVALUATION

In this section, we assess the performance of our proposed methods for VNFPRA. We develop a simulation model using Python programs to evaluate the performance of the proposed algorithms. Each MEC node is randomly deployed in the network. The system initially generates the target users, each of them is randomly distributed in the coverage area of different
TABLE 2  Simulation parameters

| Parameters | Values |
|------------|--------|
| Number of MEC nodes, $V$ | 5 |
| Number of edges, $L$ | $[5, 10]$ |
| Number of users, $N$ | 21 |
| Number of VNFs, $F$ | 75 |
| Computing capacity of each MEC node, $c_i$ | 64 units |
| Storage capacity of each MEC node, $s_i$ | 64 Gb |
| Maximum requests can be handled by a VNF, $R$ | 50 |
| Edge capacity, $c_{ij}$ | 128 Mb |

NFV-enabled MEC nodes. We use 5 MEC nodes, each having $[1.5–3.0]$ GHz processor and $[8–64]$ Gb memory. Each MEC supports 5 UEs and MEC nodes are interconnected with each other as shown is Figure 3. The simulation settings are shown in Table 2.

In this work, first of all we compare the proposed MIP model, implemented using the Gurobi optimizer [40], with the proposed GA-VNFPR to show how close our solutions are to the optimal values returned by MIP. In particular, we used the Gurobi with PyCharm to execute the MIP algorithm. Gurobi is an optimisation solver that is available for a number of programming languages. The Gurobi distribution includes a Python interpreter and a basic set of Python modules and these are sufficient for building and running optimisation models. Each PyCharm project also have a virtual environment (venv) associated with it. The gurobipy package is available to the Python interpreter of that venv. The parameters used in genetic algorithm are listed in Table 3. Furthermore, the proposed heuristic based GA-VNFPR algorithm is studied in detail to achieve the near optimal placement and cost effectiveness for VNFs. Regarding MIP approaches, they are trying to go through every possible solution to extract the optimal one but they are characterized by an increased complexity for large scale experiments. In contrast to MIP, the genetic algorithm is very simple and easy to implement by simply translating the pseudocode into a Python code. For optimal solutions, we consider the VNF placement with/without dynamic virtual link bandwidth allocation. It is not surprising that MEC nodes running a higher number of virtual functions may experience larger traffic loads, causing congestion on their outgoing virtual links, which, in turn, become bottlenecks and ultimately affect the transmission delays along those links. The reality is that not all virtual links should be allocated for the same share of the underlying physical capacity. Rather, a more (traffic-aware) proportional allocation of bandwidth is suitable. The obtained results assert that a traffic-aware link bandwidth allocation in GA-VNFPR is able to reduce the overall cost. Indeed, this is achieved by allocating more bandwidth for the links which need it most, i.e. for those MEC nodes with higher loads. Such allocation is able to reduce the transmission delays: higher capacity results in lower costs. Furthermore, to better illustrate the efficiency of our proposed approaches, we compare them against the following baseline algorithms:

(1) Resource utilisation-single objective evolutionary algorithm (RU-SOEA):

The RU-SOEA is an evolutionary algorithm designed to minimise the size of the robot body and reducing the number of collision between the robot and the obstacle during the climbing motion presented in [38]. We use RU-SOEA approach to compare it with our proposed approach by designing fitness function in such a way that the lesser is the fitness score, the best the placement solution is. Once fitness score is obtained, it will be compared to the fittest placement fitness score. If current fitness score is better than the fittest score, current solution will be kept as the fittest one. The simulation will be repeated until the maximum number of generation is reached. In our simulations GA-VNFPR and RU-SOEA both methods are able to produce optimised results with the capability to performing better placement solutions. However, one of the solutions from GA-VNFPR is outperforming the fittest solution from RU-SOEA. The reason is in RU-SOEA each evolutionary run generates single solution, the second solution will only be generated after the first solution is obtained; these processes will be repeated for obtaining other solutions. While the advantage of GA-VNFPR is its capability in generating a complete set of optimal solutions in a single run with provides users a choice of best placement solutions.
for trade-off between different MEC nodes. As the number of VNFs increases, the RU-SOEA become unable to provide a comprehensive set of solutions over multiple dimensions.

(2) Genetic non-bandwidth link allocation algorithm (GA-NBA):
This algorithm considers the VNF placement without dynamic virtual link bandwidth allocation presented in [39].

(3) Random-fit placement algorithm (RFPA):
Based on a given number of VNFs, RFPA places each VNF at MEC node selected uniformly at random.

(4) First-fit placement algorithm (FFPA):
This algorithm places each VNF at the first available MEC node that has enough capacity to host it.

(5) Gurobi optimizer:
This is an exact algorithm [40] which derives the optimal solution to (4).

We use Python programming to verify the effectiveness of the proposed algorithm. In this section, a comparison of the two approaches is presented via simulation. Figure 4 presents a comparison of the objective function values obtained for our proposed GA-VNFPPRA and optimal solution returned by Gurobi. MIP algorithm is used as a benchmark since it always has the exact optimal solution but is not implementable for large-scale systems (e.g., MIP needs more execution time than GA). When the number of iterations increases, both the objective value of the proposed heuristic GA-VNFPPRA and the Gurobi are gradually decreased and eventually tend to converge. Since we are solving a minimisation problem, lower values for a specific placement and resource allocation combination mean better performance. Our proposed GA-VNFPPRA is shown to approximate well with the optimal solutions provided by the Gurobi. However, after 2500 iterations, the proposed GA-VNFPPRA converges to a local optimum and no further improvements can be achieved while Gurobi converges after 4000 iterations, this is because it increases the particular solution computational time for large scale experimentations and thus we can only use it as a benchmark approach or an offline solution.

In Figure 5, we evaluate the performance of the proposed scheme and four existing approaches to find VNF placement cost. It depicts the cost per VNF placement with increasing number of users. It shows that our proposed GA-VNFPPRA strategy is beneficial in terms of VNF placement cost. The proposed algorithm finds the maximum number of VNFs that can be placed in MEC nodes and achieve the minimum cost. Here, we can also see how the cost of the random allocation spikes in comparison with the GA-VNFPPRA and other baseline algorithms.

In Figure 6, we applied the proposed algorithm to find the minimum computation cost. We can see that our proposed heuristic approach has an adverse effect in terms of computation cost and it performs better than RU-SOEA and other baseline approaches. The GA-VNFPPRA performs well as number of users grows by allocating optimal computing resources to VNFs and link bandwidth allocation at much lower cost. This is because in GA-VNFPPRA, we provide a heuristic approach
to organise the distribution of VNFs and to find suitable MEC nodes with maximum resources to run on top of it with the cheapest cost. If the MEC node has enough resources to host the VNFs, then the program automatically search to another valid and available MEC node to run these VNFs in it. Figure 7 demonstrates that our approach for VNFPRA consist of finding the most appropriate MEC nodes, determine the shortest paths to both the selected content nodes and the end-user with the aim of minimising link usage cost. As our proposed framework has more than one path between most of the (source, destination) node pairs, and these paths mostly use different links, the GA-VNFPRA tries to maximise the number of used links, so that the system is encouraged to use different paths to route traffic between each (source, destination) node pair. This results in distributing traffic over different paths, and reduces the average link utilisation and minimises the link usage cost. It can be seen that with the increasing number of users our proposed GA-VNFPRA gives better results compare to baseline approaches.

In Figure 8, we applied the proposed algorithm to evaluate the objective function and overall cost. In the figure, the overall cost increases with increasing number of users. RFPA performs worse than GA-VNFPRA and RU-SOEA, because there is a greater probability for RFPA to choose nodes with less remaining resources to serve VNFs. Since the GA-VNFPRA considers to reduce the resource consumption and adopt an adjustment process to cope with constraint violations, the advantage of GA-VNFPRA becomes more and more obvious with the increasing number of users.

In Figure 9, we evaluate the total number of MEC nodes required with increasing number of requests from users. The number of MEC nodes needed to facilitate the incoming requests increase rapidly for GA-NBA, RFPA and FFPA while GA-VNFPRA algorithm requires minimum MEC nodes to process requests and also it approximates well with the optimal solution. This is because GA-VNFPRA takes account of the resources of MEC nodes, load of the server, and the load of the links during the selection of VNFs to be placed. RU-SOEA also effectively distributes the incoming requests among available MEC nodes. RFPA doesn’t consider the best MEC nodes with efficient available resources, which means the allocation of CPU and memory resource is not optimal. When there are too many VNF requests from users, the optimal allocation of resources in MEC nodes for VNFs will become more important.

In Figure 10, we study the performance of VNF migration for increasing number of requests using our proposed GA-VNFM. GA-VNFM requires minimum VNF migrations while in baseline approaches, the numbers of VNF migrations rapidly increase with the number of requests from users because the baseline algorithms do not consider the load of target MEC node. Compared to these approaches the number of VNF migration in our proposed GA-VNFM steadily goes to a stable state. This is because it always selects the appropriate MEC node for each VNF.

In Figure 11, we measure the average execution time overhead added by the additional VNFs in MEC nodes. We ran a series of experiments to measure how the additional deployment of VNFs affect the CPU, memory, and network.
performance. GA-VNFPRRA algorithm can decide MEC node and network allocations for hundreds of VNFs based on the CPU core cost, bandwidth cost and the remaining resource of MEC nodes. Nodes that incur less cost and have more remaining resources are more possible to be chosen. Results show that time needed to solve the VNFPRA problem increases dramatically for GA-NBA when dealing with large number of VNFs. On the other hand, GA-VNFPRRA shows better scalability and seems more convenient for real network scenarios since it provides a near optimal solution within a reasonable time. The initial solution provided by genetic approach can impact the overall cost. The GA-VNFPRRA solution provides relatively smaller execution-time as compared to the baseline methods.

8 | CONCLUSION

A framework based on SDN and NFV MANO architecture is presented for deploying VNFs in distributed NFV-enabled MEC nodes. Furthermore, a VNFPRA problem is formulated that requires meeting the optimal placement requirements of VNFs in SDN/NFV-enabled MEC nodes to reduce the deployment and resource cost. This paper proposes a solution for this problem based on two algorithms (i) an optimal solution formulated as a MIP problem and (ii) a genetic based heuristic algorithm. We also proposed a GA-VNFPRRA algorithm to minimize the total number of VNF migrations. Results show the higher performance of the proposed methods in comparison with four existing algorithms in literature. The results also validate the fact that a coordinated placement of VNFs in SDN, NFV, and MEC can satisfy the objectives of overall reduced cost.

REFERENCES

1. ETSI, ISGMEC: Mobile-edge computing introductory white paper. (2014). https://portal.etsi.org/Portals/0/TBpages/MEC/Docs/Mobile-edge_Computing_-_Introductory_Technical_White_Paper_V1%2018-09-14.pdf
2. Taleb, T., et al.: On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration. IEEE Communications Surveys & Tutorials 19(3), 1657–1681 (2017)
3. ETSI NFV: Network functions virtualisation (NFV): management and orchestration. GS NFV-MAN 001 (2014). https://ieeexplore.ieee.org/abstract/document/8350296. Accessed 26 Apr 2018
4. Gonzalez, A.J., et al.: Dependability of the NFV orchestrator: State of the art and research challenges. IEEE Communications Surveys & Tutorials 20(4), 3307–3329 (2018)
5. Ghaznavi, M., et al.: Service function chaining simplified. arXiv:1601.00751 (2016)
6. Yala, L., Frangoudis, P.A., Ksentini, A.: Latency and availability driven VNF placement in a MEC-NFV environment. In: IEEE Global Communications Conference, pp. 1–7. IEEE, Piscataway, NJ (2018)
7. Askari, L., et al.: Virtual-network function placement for dynamic service chaining in metro-area networks. In: 2018 International Conference on Optical Network Design and Modeling, pp. 136–141. IEEE, Piscataway, NJ (2018)
8. Askari, L., et al.: Towards NFV-based multimedia delivery. In: Proceedings of 2015 IFIP/IEEE International Symposium on Integrated Network Management, pp. 738–741. IEEE, Piscataway, NJ (2015)
9. Bu, C., et al.: SDNFV-based dynamic network function deployment: Model and mechanism. IEEE Commun. Mag. 53(2), 90–97 (2015)
10. Leivadeas, A., et al.: Dynamic virtualized network function allocation in a multi-cloud environment. In: Proceedings of IEEE 21st Int. Conf. on Telecommunications, pp. 1–5. IEEE, Piscataway, NJ (2016)
17. Bhamare, D., et al.: Optimal virtual network function placement in multi-core service function chaining architecture. Elsevier Journal on Computer Communications 102(1), 1–16 (2017)
18. Lee, S., et al.: resource management in service chaining. Internet Engineering Task Force. https://tools.ietf.org/html/draft-lee-nfvg-resource-management-service-chain-01 (2016). Accessed 9 Mar 2015
19. Addis, B., et al.: Virtual network functions placement and routing optimization. In: Proceedings of IEEE Int. Conf. on Cloud Networking, pp. 171–177. IEEE, Piscataway, NJ (2015)
20. Mohammadi, A., Rezvani, M.H.: Optimization of virtual machines placement based on microeconomics theory in cloud network. In: Proceedings of 4th IEEE International Conference on Knowledge- Based Engineering and Innovation, pp. 299–303. IEEE, Piscataway, NJ (2017)
21. Rankotiya, W., et al.: Optimizing resource allocation for virtualized network functions in a cloud center using genetic algorithms. IEEE Transactions on Network and Service Management 14(2), 343–356 (2017)
22. Sciancalepore, V., et al.: A double-tier MEC-NFV architecture: Design and optimization. In: Proceedings of 2016 IEEE Conference on Standards for Communications and Networking, pp. 1–6. IEEE, Piscataway, NJ (2016)
23. Mach, P., Becvar, Z.: Mobile edge computing: A survey on architecture and computation offloading. IEEE Communications Surveys Tutorials 19(3), 1628–1656 (2017)
24. Yi, B., Wang, X., Huang, M.: Design and evaluation of schemes or provisioning service function chain with function scalability. J. Netw. Comput. Appl. 93(1), 197–214 (2017)
25. Luizelli, M.C., et al.: A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining. Comput. Commun. 102(1), 67–77 (2017)
26. Zhao, T., et al.: Pricing policy and computational resource provisioning for delay-aware mobile edge computing. In: Proceedings of 2016 IEEE/CIC International Conference on Communications in China, pp. 1–6. IEEE, Piscataway, NJ (2016)
27. Wang, L., et al.: Joint optimization of service function chaining and resource allocation in network function virtualization. IEEE Access 4, 8084–8094 (2016)
28. Baumgartner, A., Reddy, V.S., Bauschert, T.: Combined virtual mobile core network function placement and topology optimization with latency bounds. In: Proceedings of 2015 Fourth European Workshop on Software Defined Networks, pp. 97–102. IEEE, Piscataway, NJ (2015)
29. Oljira, D.B., et al.: A model for QoS-aware VNF placement and provisioning. In: Proceedings of 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks, pp. 1-7. IEEE, Piscataway, NJ (2017)
30. Cho, D., et al.: Virtual network function placement: towards minimizing network latency and lead time. In: Proceedings of 2017 IEEE International Conference on Cloud Computing Technology and Science, pp. 90–97. IEEE, Piscataway, NJ (2017)
31. Alleg, A., et al.: Delay aware VNF placement and chaining based on a flexible resource allocation approach. In: Proceedings of 2017 13th International Conference on Network and Service Management, pp. 1–7. IEEE, Piscataway, NJ (2017)
32. Benkacem, L., et al.: Optimal VNFs placement in CDN slicing over multi-cloud environment. IEEE J. Sel. Areas Commun. 36(3), 616–627 (2018)
33. Jemaa, F.B., Pujolle, G., Pariente, M.: QoS-aware VNF placement optimization in edge-central carrier cloud architecture. In: Proc. IEEE Global Communications Conference, pp. 1–7. IEEE, Piscataway, NJ (2016)
34. Bari, F., et al.: Orchestrating virtualized network functions. IEEE Transactions on Network and Service Management 15(4), 725–739 (2016)
35. Katsalis, K., et al.: SLA driven VM scheduling in mobile edge computing. In: Proc. 9th IEEE International Conference on Cloud Computing, pp. 750–757. IEEE, Piscataway, NJ (2016)
36. Hawilo, H., et al.: NFV: State of the art, challenges, and implementation in next generation mobile networks (vEPC). IEEE Netw. 28(6), 18–26 (2014)
37. Herrera, J.G., Botero, J.F.: Resource allocation in NFV: A comprehensive survey. IEEE Transactions on Network and Service Management 13(3), 518–532 (2016)
38. Lim, S.H., Teo, J.: Multi-objective vs. single-objective evolutionary algorithms for hybrid mobile robot optimization. In: Proc. IEEE International Symposium on Robotics and Manufacturing Automation, pp. 230–235. IEEE, Piscataway, NJ (2014)
39. Qu, L., Assi, C., Shaban, K.: Delay-aware scheduling and resource optimization with network function virtualization, IEEE Transactions on Communications 64(9), 3746–3758 (2016)
40. Gurobi: Gurobi optimizer. http://www.gurobi.com (2014)

How to cite this article: Kiran, N., et al.: Optimising resource allocation for virtual network functions in SDN/NFV-enabled MEC networks. IET Commun. 1–13 (2021). https://doi.org/10.1049/cmu2.12183