symmetry

MDPI

*Article*

# Optimization of 5G/6G Telecommunication Infrastructure through an NFV-Based Element Management System

**Arunkumar Arulappan [1,*], Gunasekaran Raja [1], Kalpdrum Passi [2] and Aniket Mahanti [3]**

[1] Department of Computer Technology, Anna University, Chennai 600044, India; dr.r.gunasekaran@ieee.org
[2] School of Engineering and Computer Science, Laurentian University, Sudbury, ON P3E 2C6, Canada; kpassi@laurentian.ca
[3] School of Computer Science, University of Auckland, Auckland 1142, New Zealand; a.mahanti@auckland.ac.nz
* Correspondence: arunkumar@mitindia.edu

**Abstract:** Network Function Virtualization (NFV) is an enabling technology that brings together automated network service management and corresponding virtualized network functions that use an NFV Infrastructure (NFVI) framework. The Virtual Network Function Manager (VNFM) placement in a large-scale distributed NFV deployment is therefore a challenging task due to the potential negative impact on performance and operating expense cost. The VNFM assigns Virtual Network Functions (VNFs) and operates efficiently based on network demands with resilient performance through efficient placement techniques. The degradation in performance and a tremendous increase in capital expenditure and operating expenses indicated this chaotic problem. This article proposed a method for VNFM placement using information on the resources of each nodes' Element Manager (EM), which is an efficient method to assign VNFs to each node of element management systems. In addition, this paper proposed an Optimized Element Manager (OEM) method for looking at appropriate EMs for the placement of VNF through periodic information on available resources. It also overcomes challenges such as delays and variations in VNFs workload for edge computing and distributed cloud regions. The performance is measured based on computations performed on various optimization algorithms such as linear programming and tabu search algorithms. The advent of the new service provisioning model of BGP-EVPN for VXLAN is materialized by integrating VTS with OpenStack. The numerical analysis shows that the proposed OEM algorithm gives an optimal solution with an average gap of 8%.

**Keywords:** 5G/6G networks; telecom infrastructure; network automation; element management system; optimization algorithms



**Citation:** Arulappan, A.; Raja, G.; Passi, K.; Mahanti, A. Optimization of 5G/6G Telecommunication Infrastructure through an NFV-Based Element Management System. *Symmetry* **2022**, *14*, 978. https://doi.org/10.3390/sym14050978

Academic Editors: Boris Malomed and Dorian Popa

Received: 29 March 2022
Accepted: 5 May 2022
Published: 10 May 2022



## 1. Introduction

The telecommunication network carriers have started to move toward Network Function Virtualiztion (NFV) and Software-Defined Networks (SDN) technologies. The architectural design of NFV is meant to deliver new services more efficiently and scale the network effectively [1]. The NFV and 5G/6G network work together for future generation networks establishment by virtualizing the 5G/6G network infrastructure for effective management and optimization [2]. When 5G/6G-based network slicing is enabled for the establishment of a distributed cloud, that helps in creating flexible and programmable networks as per users' future demands. Applying the distributed cloud to an NFV technology enables 5G/6G virtualization, which separates a physical network into multiple virtual networks, which would support various telecom based packet–core radio access networks, leading to greater benefits in the resilience, fault-tolerance, and scalability of the network [3]. The Border Gateway Protocol (BGP) network routing mishap creates a social impact in articulating this kind of crypto-sensitive problem that has a symmetric role in publishing this future generation telecommunication network research article in the Symmetry section

of the *Journal of Computer and Engineering Science* that is best available for relevant areas of research and community forums. The research article is also best suited for symmetry, as it contains volumes on mathematical models, simulation with experimental outcomes, conclusions and results that best demonstrate the novelty of a scientific research article. *Symmetry Basel* played a major part in providing research publications in the computer networks section that range from first generation networks to sixth generation networks. There is a lot happening in the field of telecommunication network establishment and deployment. The symmetry extended its future plan of automated network management with the help of Artificial Intelligence (AI) and Machine Learning (ML) techniques. The future work of the proposed system mentioned in this article will inherit the AI/ML techniques such as reinforcement learning, deep queue learning, and neural networks for zero touch network automation.

The future 5G and 6G networks tend to offer a great flexibility to the network operators with regard to allocation of speed, coverage, and the capacity of various use cases [4]. In real time, the operators have to program their networks for the introduction of new services rather than manual configuration [5]. This adaptability is a key factor for the end users where there is variation in demands from providing a consistent state transition for storage-based distributed systems [6]. Globally, the data traffic has grown at a faster rate with the rapid deployment of 5G/6G networks. Although the service providers have already started to virtualize their packet core networks, the due benefits of 5G/6G is reaped with the horizontal breaking of NFV silos in a telco cloud to achieve the efficiency in deploying new services with mobile edge computing and network slicing features for end-to-end orchestration of services for 5G mobile system [7].

NFV decouples Network Functions (NFs) from generic hardware appliances and runs them as software Virtual Network Functions (VNFs) in Virtual Machines (VMs) [8]. The NFs are the building blocks positioned inside a network infrastructure that include interfaces. It enables virtualization, and VNFs are executed in Commercial-Off-The-Shelf (COTS) hardware servers within the data centers [9]. The VNFs workload management has created a great buzz in the field of telecommunication for various complex network functions with the availability of necessary optimization tools and frameworks [10]. The prior work discusses various VNF Manager (VNFM) placement techniques with an elastic search model for minimizing the Capital Expenditure (CapEx) and Operational Expenditure (OpEx) cost. A mixed integer linear programming model is used for solving the VNF service chain composition problem [11]. This problem is addressed using a non co-operative game theory model for the privacy preservation of VNF management in polynomial time and security of computing environments [12]. The game theory model focuses on resource distribution among the interaction between users demand and service availability. The Markov chain optimization method is used for the dynamic placement of VNF [13]. The focus is on VNFM placement that could minimize CapEx and OpEx without compromising on the performance requirements.

The advancement in scaling up faster central processing units and methods to achieve an efficient packet processing in software has spurred NFV [14]. During NFV deployment, an operations network sets up an NFV Infrastructure (NFVI), which best describes the hardware and software components on which virtual components are built. The VNFM is responsible for the Life Cycle Management (LCM) of virtual instances [15]. LCM refers a set of functions required to instantiate, maintain, and terminate a network service or a VNF. The purpose of VNFM is to collect the personal information of a virtualized resource from NFVI and VNF indicators from the Element Managers (EMs) or VNF instances [16]. The EM manages the VNFs in the NFV framework as represented in Figure 1. The problem in optimal placement of the VNFM component in a distributed environment is new, and it is described as a VNFM Placement Problem (VNFM-PP).

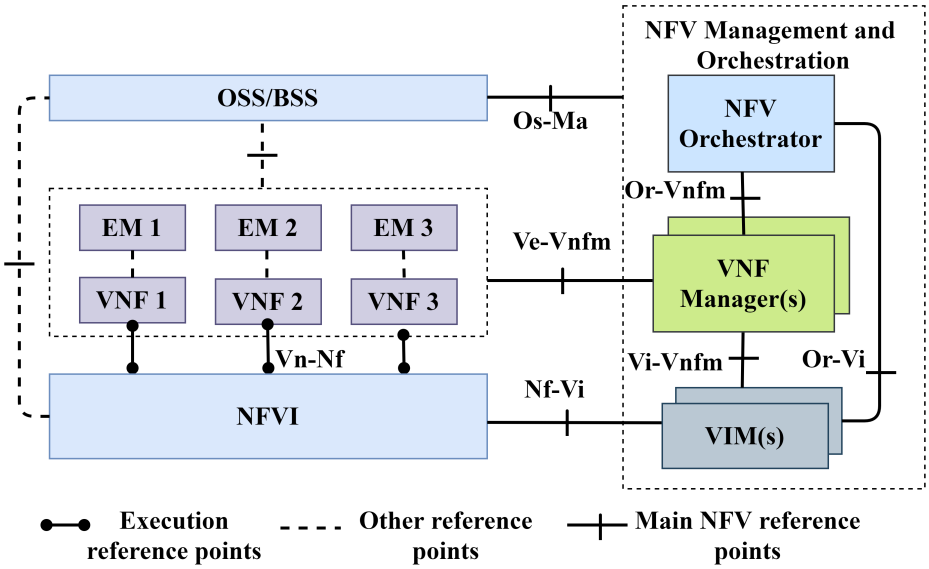


**Figure 1.** NFV architectural framework.

The telecom network management system-based VNFM placement is considered for several heuristic approaches in a distributed platform, and its evaluation gave better results. The European Telecommunication Standards Institute is a consortium of service providers meant for developing a suitable cloud-based platform (i.e., openstack cloud computing platform) to host VNFs (i.e., firewalls, routers, switches), and operators are trying to integrate NFV into openstack [17]. Development and operation is meant for automating the orchestration services and built tools for managing resources using Application Programming Interface (API). This results in a tremendous uplift of moving the infrastructure to the cloud data centers'. The NFVI Point-of-Presence (PoP) controls the distribution of software instances that result in an increase of NFVI performance with reduction of VNFM placement CapEx and OpEx. To deliver an optimized VNFM placement, the Virtual Topology System (VTS) increases resource prediction for Virtual Extensible Local Area Networks (VXLAN) overlay, provisioning a switch of the networking layer inside the data centers' programmable fabric [18]. The progression of VNFs in the cloud-optimized environment brings in attractive automation capabilities of cloud computing and Internet of Things platforms and sets a new standard within the telecommunications industry [19]. The on-demand scaling of VNF helps with the accommodation of hardware appliances whenever there is an increase in load. The major contributions of this article are as follows:

- The existing algorithms such as ILP, Greedy and TSA have not considered VNF characteristics based on the total volume of resources and have a major problem of poor efficiency in the VNFM placement. This paper proposes a node EM calculation method for assigning VNF with periodic information about resources and characteristics before assigning VNF.
- The formulation of the problem is termed as VNFM-PP, and it proposed an OEM algorithm for dynamic VNFM placement decisions. The placement uses Monte Carlo estimation for the proposed OEM algorithm instead of calculating the theoretical expectation as in existing static approaches.
- The proposed OEM algorithm supports a Border Gateway Protocol–Ethernet Virtual Private Network (BGP-EVPN) control plane system for provisioning VXLAN overlays for placing the VNFM component.

The rest of the article is organized as follows. Section 2 discusses the prior work. Section 3 presents the problem statement on telecommunication-based network management based on infrastructure VNFM placement. Section 4 presents the proposed OEM algorithm for VNFM placement. Section 5 presents the performance evaluation of the proposed algorithm. Section 6 summarizes and concludes the paper.

## 2. Related Work

In the NFV paradigm, traditional middleboxes are managed as single modules of software programmed to perform the role of a particular VNF [20]. This transformation helps to adapt and capitalize on specific goals such as speed and efficiency enhancement. The virtualization is meant to transform the traditional computing, making it efficient in managing workloads [21]. During network traffic, in applying the policy or providing network services to traffic, it needs to undergo a particular sequence of VNFs called a service chain [22]. This transformation is possible only through the consolidation of different VNFs in standard general-purpose computers (servers, storage devices, compute), which can be located either in data centers or network nodes [23]. In order to improve resource utilization, two main factors are considered: the initial thing is to consider the Fundamental Resource Overheads (FROs) incurred by instantiating VNFs in servers [24]. The existing work focuses on the VNF placement problem for mapping service function chaining of resources in cloud networks.

The FROs are considered, and past researchers mathematically formulated the resource optimization and guaranteed delay in the VNF placement problem, which is said to be NP-hard. To solve this problem, they devise a polynomial-time approach called Two-Phase Optimization Solution (TPOS) [25]. The releasing of redundant VNF instances with varying network load could save more server resources and improves resource utilization. A detailed analysis and evaluation of TPOS was conducted; for large-scale problems, it guarantees the delay requirements and outperforms benchmarks in terms of server resource consumption, activation and utilization. The optimal location determination of VNF and VM was determined, and they also focused on the VPS-CD problem as an MILP and proposed a cost-efficient VPS Scheme (CE-VPS) to minimize the cost of VNF placement and scheduling to deploy VNFs in a public cloud [26]. The MILP minimizes the cost computationally for large networks with the development of an efficient heuristic algorithm.

The ILP based Power-aware and Delay-constrained joint VNF placement and Routing (PD-VDR) problem is formulated [27]. The problem is addressed based on the consideration of VNF processing time, which changes dynamically based on network traffic. A Holu efficient heuristic algorithm is proposed for solving a PD-VDR problem using heuristic routing and employed a Delay Constrained Least Cost (DCLC) shortest path algorithm [28]. The problem of dynamic VNFM placement in large-scale distributed NFV system is discussed. A system is proposed with a generic ILP formulation that determines the number of VNFM placement at minimum [29]. They also proposed a TSA-based meta-heuristic technique to solve the placement problem of VNFM in both static and dynamic environments [30]. The focus is on the VNF placement problem in data center networks with guaranteed resource optimization and resource availability [31]. They proposed a novel Joint Path-VNF (JPV) backup model with affinity iterations on algorithms for the reducing the link consumption, and performance evaluation showed that JPV improves better with network availability [32].

The problem of service chain placement for the VNFs within the cloud is discussed for mixed ILP for VNF allocation and network management in cloud data centers. The proposed system comprises two new resource allocation algorithms based on genetic programming for the initial placement and scaling of VNFs to support traffic changes. The evaluation is carried out in data centers, with an efficient utilization for both computing resources and network configurations being managed concurrently in an openflow infrastructure. The problem of VNF placement and scheduling in a radio access network domain is discussed, as they proposed a slice scheduling mechanism with performance isolation between different slices, and numerical simulation shows that the proposed heuristic algorithm gave an optimal solution. In most of these related works, the coupling between different VNFM placement interface objectives has not been addressed.

Few of the existing works deal with NFV architecture in consideration of a VNF placement problem with multiple objectives such as network resource cost and license cost. However, they do not provide any effective solution to optimize the cost. The existing

scheme has not considered VNF characteristics based on the total volume of resources with poor efficiency during VNF placement. The VNF-specific characteristics indicate that the usage of more CPU, memory, and disk resources leads to inefficient placement of VNF. The existing algorithms such as greedy, TSA, ILP, dynamic programming, and Djikstars' are used for the optimal placement of VNFM. The proposed scheme is concerned with optimized methods for VNFM placement without wasting resources. The VNF is used to predict demands for various VNF resources, analyzing the correlations among those resources such as CPU, memory, and disk.

The optimized VNF placement performance can be increased using ILP as the number of nodes, VNFs to assign, complexity and computation increases with complexity. The demerit of the ILP technique is its extendability; it is not efficient as the size of the problem expands and cannot deal with algorithms that run exponentially. The consequent demerit of increasing complexity is that the performance deteriorates based on optimization algorithms used in computing environments. The issues in addressing the placement of VNFM rely either in data centers or servers inside the network. The issues are further discussed in two different ways: Firstly, the minimal placement of VNFs increases the forwarding cost and leads to traffic churns. Secondly, a maximum number of VNFs placement with a decrease in forwarding cost brings stability to the network. The proposed algorithm follows methods with optimization techniques that increases efficiency with VNFM placement. This paper introduces an Optimized Element Manager (OEM) for VNFM placement as a core technique in NFVI of the NFV framework.

The aim and key outcomes of this article are as follows:

- To transform the way network operators design, manage, and deploy their network infrastructure. Earlier, it required low latency for signal processing, wherein now, users require higher throughput for data traffic processing.
- The software instances of network functions and services are implemented in the form of VNFs. The placement of VNFM is managed according to their demand in the network, which dynamically changes with time.
- The cloud service providers should route their application flows during VNF placement, i.e., through a sequential ordering of VNFs specified in service function chaining.
- The distributed infrastructure helps to materialize degradation in performance and type of placement instantiated in line to a set of instances for large-scale environments.
- The chaining approach determines the path traversal of VNFs that are dynamically created, embraces a significant decrease in CapEx and OpEx for telecom operators, and makes necessary alterations in the way network functions are developed and deployed.

## 3. Problem Statement

In large-scale distributed environments, during VNF instance deployment, the VNFM maintains VNF instances without compromising its management function performance. However, VNFs are initiated on a demand basis and scaled elastically for effective maintenance. The number of VNFM adoption is based on VNF instances deployment for optimal performance [15]. A telecom network-based VNFM management of the NFV ecosystem challenge is addressed for enhancing the NFVI dynamism and efficiency. This function faces challenges such as delays and variations in VNFs workload. The significance in VNFM placement affects the overall system performance and operational cost. The various cost considerations are listed in a summary of key notations, as shown in Table 1:

In a telecom-based NFV ecosystem, the VNFM is a key component. The placement of VNFM becomes critical, as cost and performance optimization are needed for NFV systems, and so, the placement issue is labeled as VNFM-PP. Hence, we determine the placement function $c^* \epsilon \phi$ such that

$$P(c) = min_{c \in \phi} P(c)$$

where $P(c)$ is a criterion function [25]. The computational placement has been represented as placement $P(c) = P1, P2 \ldots Pk$; let $V(Cn, Cq)$ denote a set of virtual computes, and the

corresponding interconnection between these computes happens through overlays, which is expressed as $O(Cn, Cq)$. The hardware overlay provisioning that happens in L3 fabric using Spine-Leaf (SL) topology results in global error placement can be expressed as:

$$P(c) = \sum_{c_n, c_q} \in c\ min_{v(C_n, C_q)} d(O(C_n, C_q, V))$$

### *3.1. Capital Expenditure Cost (CapEx)*

Four different computational cost evaluation parameters have been considered for our CapEx, as shown below.

#### 3.1.1. PoPs Site Components Cost $(C^{site}(t))$

The PoPs site components cost is calculated by representing the number of computing resources allocated for a data center and by calculating the cost of its size for every individual VM, as shown in Equation (1).

$$C^{site}(t) = \frac{n_d}{s} * \frac{c}{d} * n_s \tag{1}$$

**Table 1.** Summary of key notations.

| Symbols | Description |
|---|---|
| $n_d$ | Number of devices |
| $n_s$ | Number of sites |
| d | Device |
| s | Site |
| C | Cost |
| $C_{sd}$ | Cost of software development |
| $Cf_{ti}$ | Cost of first time installation |
| $sdn_c$ | SDN ontroller |
| $n_{sdn_c}$ | Number of SDN controllers |
| $O_{nd}$ | Number of devices |
| rs | Rackspace ($m^2$) |
| $y_r$ | Yearly rent |
| $Y_c$ | Yearly consumption |
| $d_{kw}$ | Device in kW |
| $C_{kw}$ | Cost in kW |
| Y | Year |
| $n_s$ | Number of shifts |
| w | wage ($m^2$) |
| hrs | Hours |
| S | Shift |
| $nh_f$ | Number of hardware failures |
| $d_f$ | Distance to failure (in km) |
| $fl_t$ | Failure location time |
| $ff_t$ | Time to fix the failure |
| $c_{hr}$ | Cost of hardware replacement |
| $n_{sf}$ | Number of software failures |
| $at_{sf}$ | Avg time to fix a software failure |
| $n_{cc}$ | Number of connections to be configured |
| $ct_{pc}$ | Configuration time per connection |
| $dt_{pc}$ | Documentation time per connection |
| $n_{cr}$ | Number of connections to be reconfigured |

3.1.2. SDN Components Cost $(C^{sdn}(t))$

The overall cost for performing an SDN computation is mentioned in Equation (2). It is incurred with the number of SDN controllers and the cost required for an individual SDN controller.

$$C^{sdn}(t) = n_{sdn_c} * \frac{c}{sdn_c} \tag{2}$$

3.1.3. Installation Cost $(C^{inst}(t))$

The installation cost includes physical hardware resources required for performing compute, storage and network operations in an infrastructure medium for setting up networking services. The cost of software development and first-time installation bind together to obtain an average cost, which gives the total cost spent on installation as shown in Equation (3).

$$C^{inst}(t)) = C_{sd} + Cf_{ti} \tag{3}$$

*3.2. Operational Expenditure Cost (OpEx)*

Four different operational cost component evaluating parameters have been considered to define our OpEx, which are described below.

3.2.1. Continuous Cost of Infrastructure $(C^{infra}(t))$

The continuous cost of infrastructure is exemplified in managing the infrastructural components such as compute, storage and network by inhibiting several cost parameters such as the controller, network devices and rackspace as represented in Equation (4).

$$C^{infra}(t) = O_{nd} * rs(m^2) * \frac{y_r}{rs(m^2)} + n_d * \frac{Y_c}{d_{kw}} * \frac{c_{kw}}{Y_r} \tag{4}$$

3.2.2. Cost of Maintenance and Repair $(C^{main}(t))$

The cost of maintenance and repair characterizes several parameters for hardware failures as shown in Equation (5) that includes distance, time, wage, software and hardware replacement and time taken to fix these parameters.

$$C^{main}(t) = n_s * \frac{\frac{hr}{s}}{y_r} * \frac{w}{hr} + nh_f * \left[d_f * \frac{c}{km} + fl_t * \frac{w}{hr} + ff_t * \frac{w}{hr} + c_{hr}\right] + n_{sf} * \left[at_{sf}\frac{w}{hr}\right] + \frac{n_{cc}}{yr} \tag{5}$$

3.2.3. Cost of Service Provisioning $(C^{serprov}(t))$

The service provisioning cost is exemplified by calculating the time taken for a service to be reconfigured in a particular year and the service connection establishment time being represented in hours as shown in Equation (6).

$$C^{serprov}(t) = \left[ct_{pc} * \frac{w}{hr} + dt_{pc} * \frac{w}{hr}\right] + \frac{n_{cr}}{yr} \tag{6}$$

3.2.4. Cost of Service Management $(C^{sermgnt}(t))$

The cost of service management characterizes the configuration and management time in Equation (7). It is being represented in hours required for establishment of the service and managing of resources.

$$C^{sermgnt}(t) = ct_{pc} * \frac{w}{hr} + dt_{pc} * \frac{w}{hr} \tag{7}$$

The sum of the weighted cost of $CapEx$ and $OpEx$ is expressed in Equation (8), where $Nc$ is the number of PoPs per km$^2$. The total number of PoPs site cost per km$^2$, where $CapEx$ and $OpEx$ are the cost per data center, and $i$ is the interest rate defined for $y$ years:

$$\epsilon_{Total} = \frac{Nc}{\text{km}^2}\left(CapEx\frac{i(1+i)^y}{(1+i)^y-1} + OpEx\right) \tag{8}$$

The software-based VNF instances spun across geographically distributed areas using NFVI-PoPs that result in an increase of NFVI performance and thereby reducing the CapEx and OpEx costs during VNFM placement. The EM helps in the functional management of VNFs using proprietary interfaces. The placement function is not handled by a single component in an NFV framework. The communication between functional blocks (NFVI, VNFM, VIM) happens through VNF instances. The full architectural framework depicts both the functional blocks and the main reference points between these blocks. Hence, for the optimal VNFM placement, VTS increases prediction of the resource requirements of each component for VXLAN overlay management and provisioning system for data center networks. We proposed an algorithm termed as OEM, which will be foreseen as a contending model in comparison to Greedy, ILP and TSA. Existing algorithms have the potential to resolve the scaling of VNFs based on traffic changes. It is traditionally used to optimize VNFM placement based on VNF allocation and network management in cloud data centers.

## 4. Proposed OEM Approach

This article proposed a method for VNFM placement using information about the resources of each nodes' EM, which is an efficient method for assigning VNFs to each node of an element management system. In addition, an Optimized Element Management (OEM) algorithm is devised for looking at appropriate EM for the placement of VNF through periodic information about resources. The proposed OEM Algorithm 1 accounts for the placement of VNFM in distributed NFV environments. The proposed algorithm and its activities are listed as follows: (1) Mapping the infrastructure components to their respective VNFs software switches in L3 fabric and Virtualized software VXLAN overlay provisioning of the BGP-EVPN component using RR. (2) For Cloud-based deployment, the OpenStack open-source cloud platform can impose the NFV-Cloud-based data center networking to provide customized services. (3) East–west traffic shaping using neutron networking component. With these approaches, the proposed OEM reduces the overall computational cost to control the CapEx and OpEx involved in VNFM placement.

### *4.1. OEM Algorithm for Mapping Compute and VXLAN Overlay Provisioning of BGP-EVPN*

The integration of Compute Components C1, C2, and C3 with conditional expectations are calculated using the Monte Carlo method for our proposed OEM algorithm. Our OEM framework is examined using exponential distribution. In an OpenStack project, Virtual Machines VM1, VM2, and VM3 are associated with the respective Switches S1, S2, and S3, incurring VLAN_ID and VNI_ID to it. The VMs are independent random variables that follow an exponential distributions with terms $\phi_1$, $\phi_2$, and $\phi_3$, respectively. Based on our proposed OEM algorithm, we simulate $(vnf1_1^0, j = 1, ..., m)$ and $(vnf2_1^0, j = 1, ..., m)$ based on

$$f_{vm1}(vm_1) = \phi_1^0 e^{-\phi_1^0 vm_1}$$

$$f_{vm2}(vm_2) = \phi_2^0 e^{-\phi_2^0 vm_2}.$$

The revised computational cost for evaluating OEM (BGP-EVPN) algorithm operations are computed in Equation (9) for multiple EMs with '$n$' number of computations $C_n$ performed on '$n$' number of VMs.

$$E\left((C_n(vm_n))|X = x_i\right) = \frac{\sum_{j=1}^{m}(-vm_{n_1}^{0})\phi_3^0 e^{-\phi_3^0(x_i - vm_{n_1}^{0} j - vm_{n_2}^{0} j)}}{\sum_{j=n}^{m}\phi_3^0 e^{-\phi_3^0(x_i - vm_{n_1}^{0} j - vm_{n_2}^{0} j)}} \tag{9}$$

**Algorithm 1** OEM Algorithm

**Input** The set 'N' of Compute Components (C), Virtual Network Functions (VNFs), Switches (S), VLAN_ID, VNI_ID

**Output** Mapping Computes in 'N' that are assigned to multiple switches in L3 Fabric

```
Set end ← false and field ← ϕ;
while !end and C(n)≠ ϕ do
  msg ← setting up the system from Compute node 'N'
  if msg = "out-of-sync commit" then
    n ← enter the DHCP server IP address;
    if S_c^t ≤ S_c^n, ∀p ∈ ρ then
      n ← enter the Any Cast Gateway Mac address;
      Accept: disable synchronize config feature in network inventory
      d:S_c^n = S_c^n − S_c^t ∀ p ∈ ρ
      VM1 is assigned to component (C1, S1 (VLAN 100), VNI 5000)
    else if S_c^t ≤ S_c^n − S_c^t − |field|^ρ and c ∈ C(n) then
      for each n' with n ∞ d, n' do do
        Sends a rejected message to n' then
        VM2 is assigned to components (C2, S2 (VLAN 200), VNI 5000)
        Reject: enter default range for VLAN pools in the start and end fields
        Update resources of d:
        S_c^t = S_c^n + S_c^t ∀ p ∈ ρ
      end for
    else
      Sends a rejected message to components C_n;
    end if
  end if
  if msg = "stop" then
    end ← true;
  end if
end while
```

### *4.2. Integrating VTS Use Case with OpenStack*

The Virtual Topology System (VTS) is a use case designed for provisioning and managing the overlay networks automatically. It is suitable for standard-based protocols such as BGP-EVPN and VXLAN for automated overlay provisioning in fabric automation to reduce the complexity in data center operations and increase the agility of multi-cloud environments. In OpenStack, we have projects, and once we create a project in OpenStack, that information is captured via a plugin API and VTS recognizes it. Project A translates into a Tenant A within VTS, and the VTS created Tenant A as a canvas. So, within the OpenStack project A, we have created two networks, as shown in Figure 2. The moment we create the network in the OpenStack project, that information is sent to VTS by Rest API, and then VTS immediately assigns resources from its pool of resources automatically; i.e., Network N1 is assigned with VNI 5000, and Network N2 is assigned with VNI 5001. This assignment is done automatically, it is seamless and there is no user interaction required. Until this point, we are not provisioning anything in the switches. When we place a VNF and attach it to the network, nova performs network computation, and it identifies that

VNF1 should now be placed on component C1; at this time, the port attachment will be propagated to VTS. It now knows that VNF1 is placed in C1; it looks into its topology database and identifies that C1 is connected to S1 and Ethernet port E1/1. So, it automatically start provisioning more resources and consequently identifies that VNF1 is on C1, which is connected to S1 and it assigns VLAN 100 on S1 and associates it with VNI 5000 that is assigned to that network. VTS uses Nexus API that will provision the VNI on the switch S1 and also provisions VLAN 100 on Ethernet E1/1 in the southbound direction.

The sequential activity of VXLAN overlay provisioning for VTS integration with OpenStack as mentioned in Figure 3 is as follows: If VNF1 from C1 reaches VTS, and VTS provides VLAN 100 as feedback to the NA, and NA provisions VLAN 100 on the OvS. The OvS has to know to which VLAN the VNI belongs, so the NA reaches out to VTS for the VLAN information. The service mapping for VLAN is configured to VLAN 100 and switch s1 via VNI 5000 in the VXLAN segment. Whenever we provision the second VNF2 on the same network, the port attachment will be propagated to VTS, and it assumes that nova places the VNF in C2. The VXLAN segment is set up between the two switches, and the peering information from RR and the BGP peer is set up automatically. There is no manual intervention required from the user to set up the control plane to manage the host, and the reliability within the fabric could be achieved seamlessly.

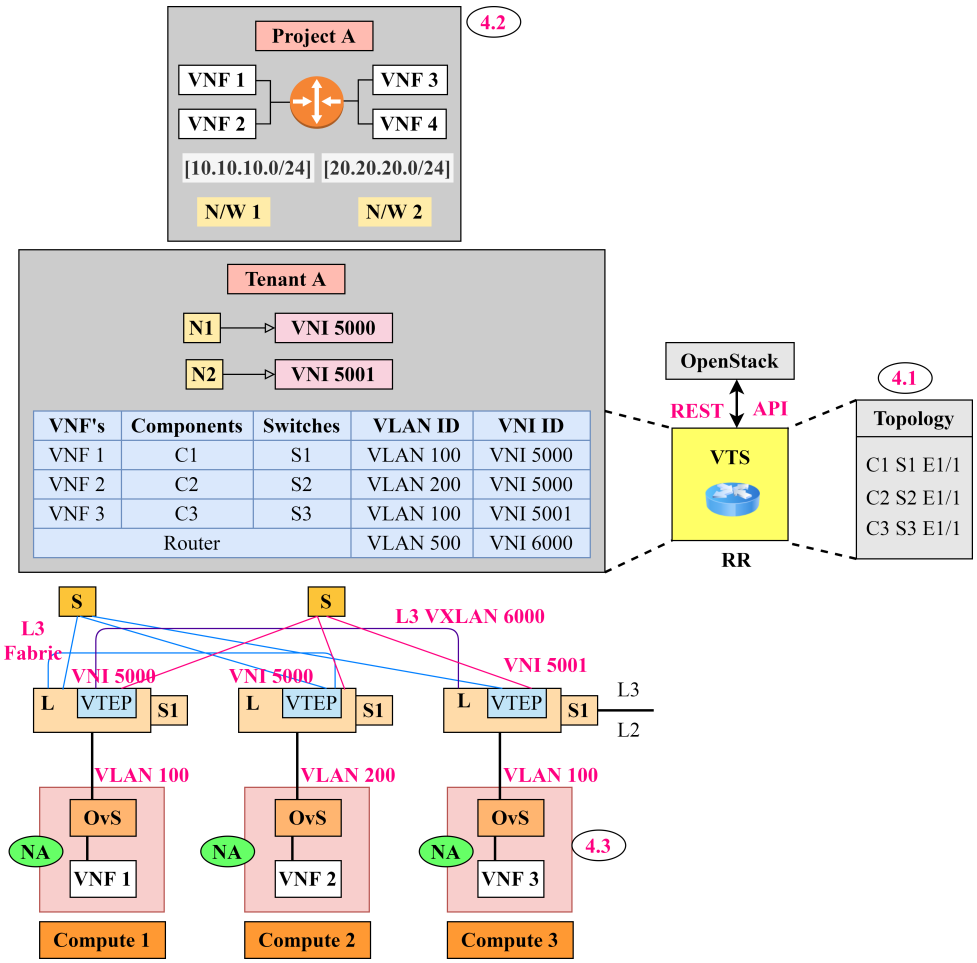


**Figure 2.** OpenStack Integration with Virtual Topology System.

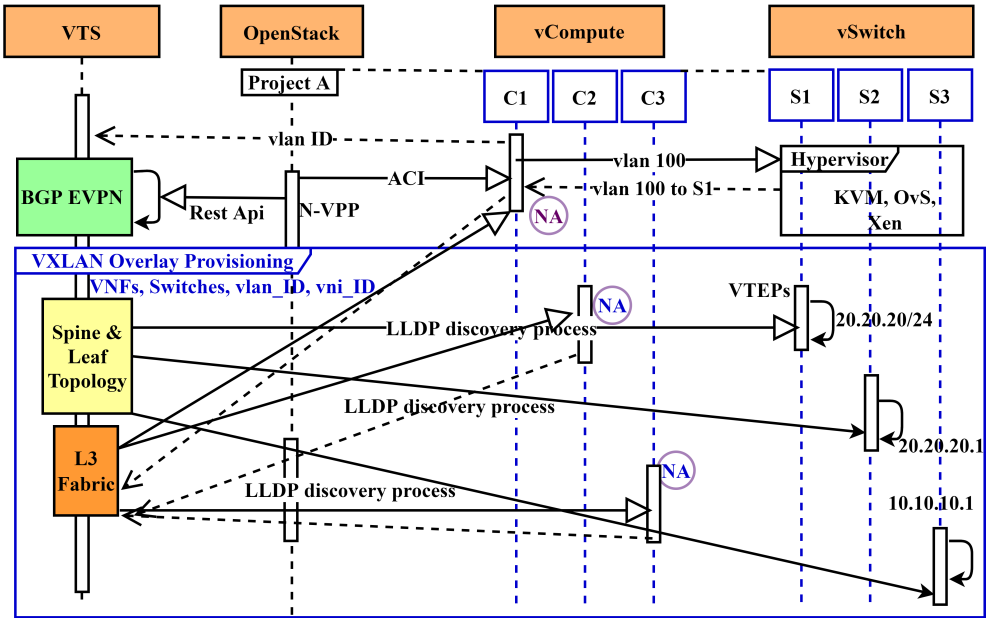


**Figure 3.** Sequence activity for VXLAN Overlay Provisioning.

In OpenStack, we can go ahead to provision a router and attach its interfaces to the two networks for VXLAN VNF communication, as represented in Figure 4. This information is propagated to VTS via plugins and VTS now goes and provisions a distributed L2 and L3 gateway on every switch that is part of the two networks. Moreover, VTS goes over to the provision switched virtual interface, Any cast Gateway MAC_ADDR, Gateway IP_ADDR on every switch. In effect, we are setting up an L3 VXLAN segment that spans across all these switches with a new VNI 6000, and it has another VLAN ID 500 that is auto provisioned as an L3 VXLAN segment, so the network can talk to each other. Since VM3 belongs to the [20.20.20/24] network, in the VTEP, [20.20.20.1] is accessible to VM3 locally. At the same time, VM1 and VM2 belong to network1, the gateway attached to gate1 [10.10.10.1] and the same gateway will also be attained at Switch2. By this, any VM at Net1 that needs to communicate with VM3 in Net2 will now be able to communicate across the VXLAN segment between the two networks.

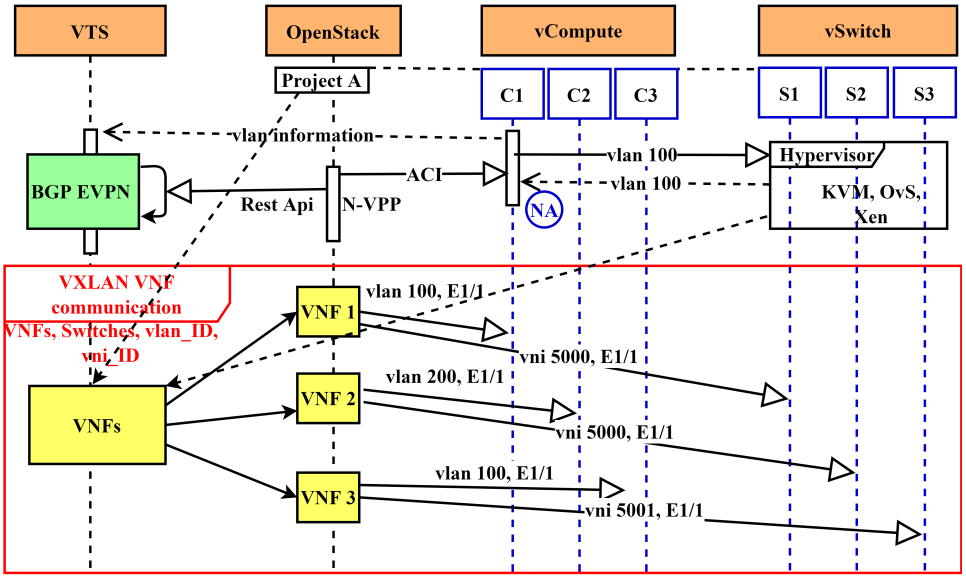


**Figure 4.** Sequence activity for VXLAN VNF Communication.

### *4.3. Management of Traffic Traversal in OpenStack Neutron Network*

The OpenStack neutron API, a virtual Network Interface Card (NIC) for providing network as a service (i.e., between interface devices) uses three main networks, as mentioned in Figure 5. To start with the Management network, it takes care of managing communication between the nodes, and the communication happens via Ethernet. In contrast, the provider network takes care of external communication from the instance, and it uses a Linux bridge+VLAN mechanism and lastly self-service or VMnet, which take care of the tenant network by using Linux bridge+VXLAN. The Linux network consists of multiple VMs, and KVM is used as a hypervisor, and these VMs are connected via a Linux bridge with several switches such as firewall, VLAN, and VXLAN. VLAN is meant for a provider network that uses a public IP for communicating with the external sources. In contrast, VXLAN is intended for a self-service network that uses private IP for internal communications.

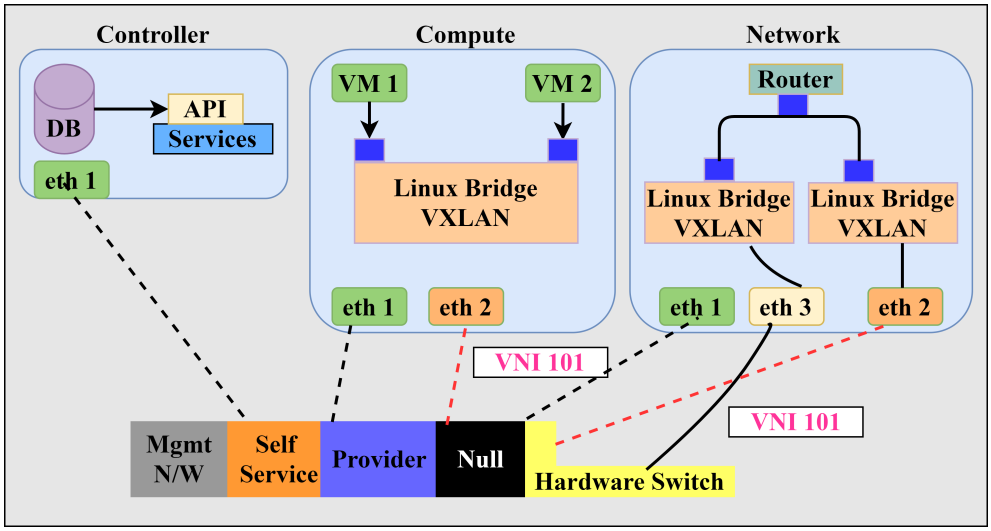


**Figure 5.** Open Networking Architecture for NFV Infrastructure.

## 5. Performance Evaluation

This section evaluates the OEM algorithm performance. The result obtained after evaluation is compared to first fit greedy algorithms such as TSA and ILP. NFV performance management involves the process of monitoring the NFV performance. A critical challenge in monitoring is to ensure that network performance remains as good as a purpose-built hardware. Two of the most common performance management tools are network performance management and application performance management. An initial obstacle occurs with a visibility gap during NFV deployment as the network management tool runs operations in a periodic manner, where the tools provide visibility to all of these interdependencies. The next challenge is in adapting NFV to the network performance tools that start interacting with the digital infrastructure and orchestration services. The carrier network design choices have a significant impact on the network performance and cost. Critical design choices are categorized into four areas based on the functional design of NFVI architecture; it can be listed as the packet-forwarding method, data-center design, hardware choices, and software efficiency.

The proposed OEM algorithm is implemented with a Monte Carlo estimation model and solved using CPLEX Optimization Studio V12.6.3. These greedy algorithms assess the VNF dynamic sequence on VNFM placement operations. The CPLEX is constructed as a baseline in order for the evaluation of the proposed OEM algorithm for large-scale deployment. The TSA and ILP algorithms iterate on a set of VNF and its instances. During the initial iteration process, the VNF instance is assigned to first VNFM with its adequate capacity requirements. The NFVI-PoPs changes the constraints in a capacity that satisfies with VNF-PP constraints.

### *5.1. Simulation Tool*

The simulation for evaluating OEM is carried out in connecting to the servers of Cloudlab.us, where eight servers are connected together as a chain in a network service function testbed. The data plane development kit is preliminary for performing any kind of traffic steering operation. OpenNetVM Speed Tester NF is a tool which evaluates realistic traffic loads in the management layer with machines of eight 10 Gbps NIC ports. Our goal in this setup is to send traffic from nodes 2 to 10. The bridging between the initiator node1 to executor node11 happens using NIC ports. So, the intermediate nodes from 2 to 10 will act as transparent middle boxes, which is useful for chaining servers. When the web traffic is initiated from node1, a chain of activities happens, which helps in traffic steering activities until before the traffic reaches the descendant node12. Speed tester NFs can automatically balance the traffic across two NFs, which thereby results in the consistent delivery of network services and improves the performance of up to 68 Gbps in comparison to major network speed testers such as Iperf, PRTG network monitor and Wireshark.

### *5.2. OpenStack Testbed Experimental Setup*

Our testbed uses the private cloud that runs on the OpenStack opensource cloud computing platform. The deployment of VMs is carried out using Red Hat Enterprise Linux 7, and the whole experiment follows OpenStack Ocata release. Initially, we performed VLAN (switch) configuration by configuring VLAN elements (VLAN_ID, VLAN_Name) and later assigned IP to VLAN. For our experiment, we used rackserver in which rack 1 is meant for nova compute services. Rack 2 consists of neutron networking controller services. Finally, rack 3 is used for storage services (object swift, block cinder). It uses OVS as a hypervisor agent which acts as a software switch for managing the VMs. A summary list of simulation parameters and values are displayed in Table 2. The physical interconnection between OpenStack use case happens via a hardware switch that has management, self-service and provider networks. The OpenStack experiment results in minimal production of overlay traffic for self-service networks that best route the traffic via the management network instead of a dedicated network.

**Table 2.** OpenStack Testbed Simulation Summary.

| **Parameters** | | **Values** | |
|---|---|---|---|
| **Rack Servers** | **Processor** | **Memory** | **Storge** |
| Rack 1 (Nova Compute) | 1 | 2 GB | 10 GB |
| Rack 2 (Neutron Controller) | 1 | 4 GB | 5 GB |
| Rack 3 (Storage) | 1 | 32 GB | 100 GB |

### *5.3. Performance Analysis*

The optimality is achieved by calculating the total incurred cost in our proposed OEM algorithm with comparison of the two existing ILP and TSA algorithms. The performance is measured based on cost metrics for obtaining the total incurred cost within 10 time slots, as shown in Figure 6. In Figure 6a, the ILP algorithm tends to consolidate the node utilization of resources and reduce the active nodes, as the operational cost values tend to be lowest. In consideration from time slots 7 to 9, the OEM operational cost is lower in comparison to ILP as the cost of ILP increases. In Figure 6b, the analysis of traffic cost clearly shows that TSA incurs the lowest traffic cost and it involves lot of active nodes. Although TSA looks not so promising at first comparison, it obtains better results with traffic cost comparison. In Figure 6c, the capital cost comparison proves that our proposed OEM algorithm obtains the lowest CapEx with a good marginal ratio with the changed workload compared with ILP and TSA. In Figure 6d, in 10 time slots, OEM reduces 21.3% and 10.4% of the total cost been compared with ILP and TSA algorithms, respectively. Lastly, OEM demonstrated the optimal way of procuring the network traffic cost, CapEx cost and OpEx cost. The prefetched result signifies that our proposed algorithm is efficient in element

management with NFV-based VNFM placement with better optimality in comparison to ILP and TSA algorithms.

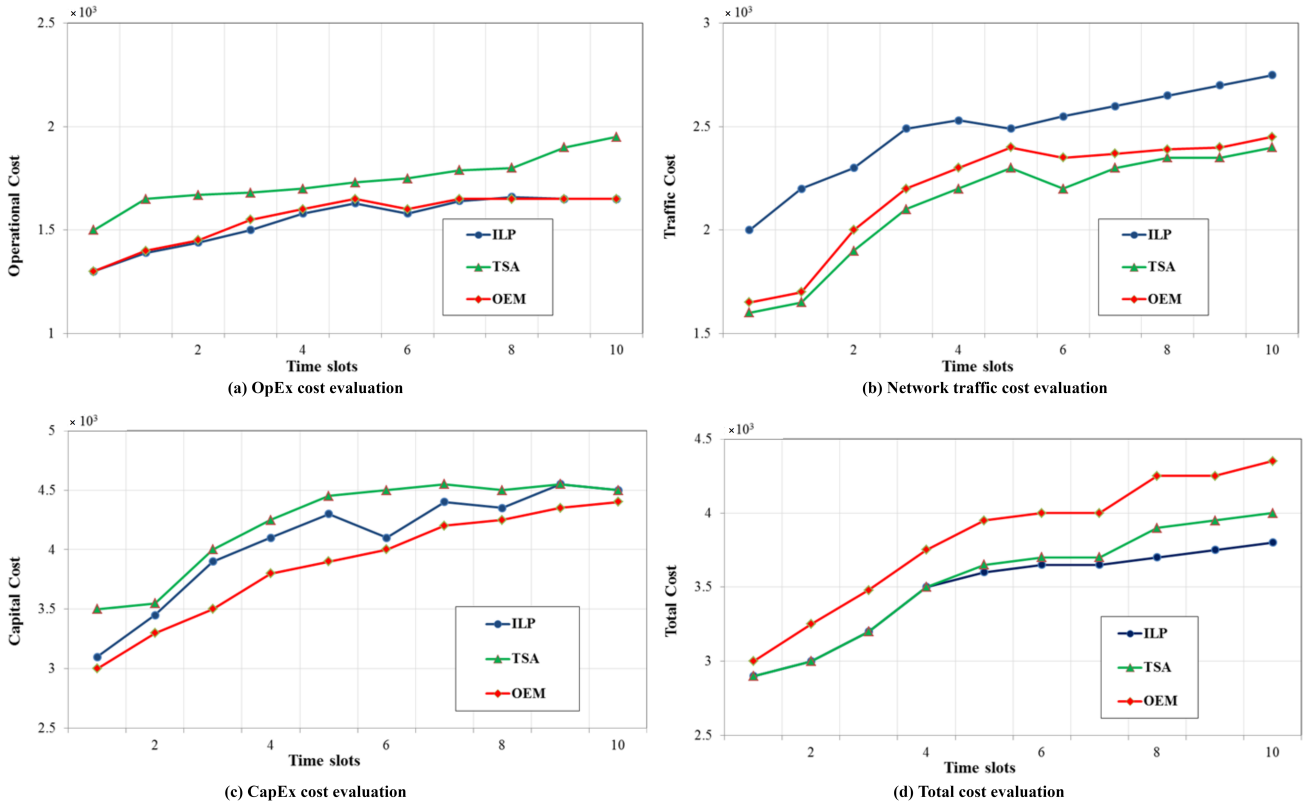


**Figure 6.** Evaluation of the total incurred cost within 10 time slots (**a**) OpEx cost evaluation, (**b**) Network traffic cost evaluation, (**c**) CapEx cost evaluation and (**d**) Total cost evaluation

### *5.4. Algorithmic Performance Analysis*

The test bed for our experimental setup is composed of a server with an Intel Core Xeon processor with 128 GB variant memory. The total execution time of the OEM algorithm in comparison to the baseline CPLEX optimizer is represented in Table 3. The complexity of the proposed OEM algorithm time complexity is O(nlogn), as it involves Monte Carlo estimation as the placement problem is said to be NP-Hard. A better solution for the VNFM placement problem is carried out as VNFM can either proactively or reactively respond to VNF status updates in the network and perform management operations in comparison to other optimization algorithms such as ILP and TSA based on the delay in the performance of the modeling network. The performance evaluation is measured by evaluating the quality of the solution derived from comparison with the existing Greedy, ILP and TSA algorithms based on CPLEX. The OEM algorithm analysis with $\epsilon = 1$ and $\epsilon = 20$ clearly shows that the OpEx cost corresponds to optimality with an average that remains smaller than 8% for 8, 16 and 24 PoPs. The result shows that our proposed OEM algorithm is faster for many of the dynamic VNF class parameters where the total execution time is greater than one second.

**Table 3.** Average execution time.

| Experiment Parameters | | | Execution Time (s) | |
|---|---|---|---|---|
| **NFVI-PoPs** | **VNFs** | $\epsilon$ | **CPLEX** | **OEM Algorithm** |
| 8 | 100 | 1 | 2.9 | 0.37 |
| 8 | 100 | 20 | 0.25 | 0.29 |
| 8 | 200 | 1 | 3.10 | 0.77 |
| 8 | 200 | 20 | 4.22 | 0.49 |
| 8 | 500 | 1 | 448.30 | 1.72 |
| 8 | 500 | 20 | 61.20 | 1.54 |
| 16 | 500 | 1 | 44286 | 2.34 |
| 16 | 500 | 20 | 61.80 | 1.32 |
| 24 | 500 | 1 | ∞ | 1.66 |
| 24 | 500 | 20 | ∞ | 1.37 |

### *5.5. Bandwidth Optimization in EMs*

The optimization in the placement of VNFM is addressed in our proposed OEM algorithm with comparison to existing ILP, and TSA algorithms. The performance is measured based on the criterion function, as it includes two metrics: the compute resource cost and LCM cost. The bandwidth consumption of LCM is bigger than the $\epsilon$ value, and the placement is favored in minimizing the LCM cost, as represented in Table 4.

**Table 4.** Average execution time.

| Parameter | VNF C1 | VNF C2 |
|---|---|---|
| VNF Indicators | 25 | 10 |
| Performance Metrics | 50 | 20 |
| Max delay b/w VNFM, EM | 25 ms | 50 ms |
| Max delay b/w VNFM & NFVO | 60 ms | 70 ms |
| Traffic collection of VNFM | 20 ms | 50 ms |
| B/W used b/w VNFM & NFVO | 0.32 MB/h | 0.12 MB/h |
| B/W used b/w VNFM & VIM | 1.18 MB/h | 0.56 MB/h |
| B/W used b/w VNFM & VNF/EM | 1.54 MB/h | 0.42 MB/h |

The uniformity in bandwidth cost over communication links in NFVI-PoPs translates the additional placement of VNFM on NFVI for managing VNF instances. This placement method reduces the overall traffic on the communication links and will be circulated to NFVI-PoP without incurring any cost. The evaluation in Figure 7 describes the number of VNFM placed on NFVI, in relation to the number of VNF instances for 8, 16 and 24 NFVI-PoPs with consideration to $\epsilon = 1$ and $\epsilon = 20$. There is a steady rise in the number of VNFM, if $\epsilon = 20$ is being compared to $\epsilon = 1$. Digging deeper, we observed that the additional VNFMs allocated are of compute C1 type. In our evaluation, the VNF instances with compute C1 utilizes more bandwidth in comparison to C2 VNF instances. Overall, our OEM algorithm significantly outperforms Greedy, ILP, and TSA with average gaps that are smaller than 8%.

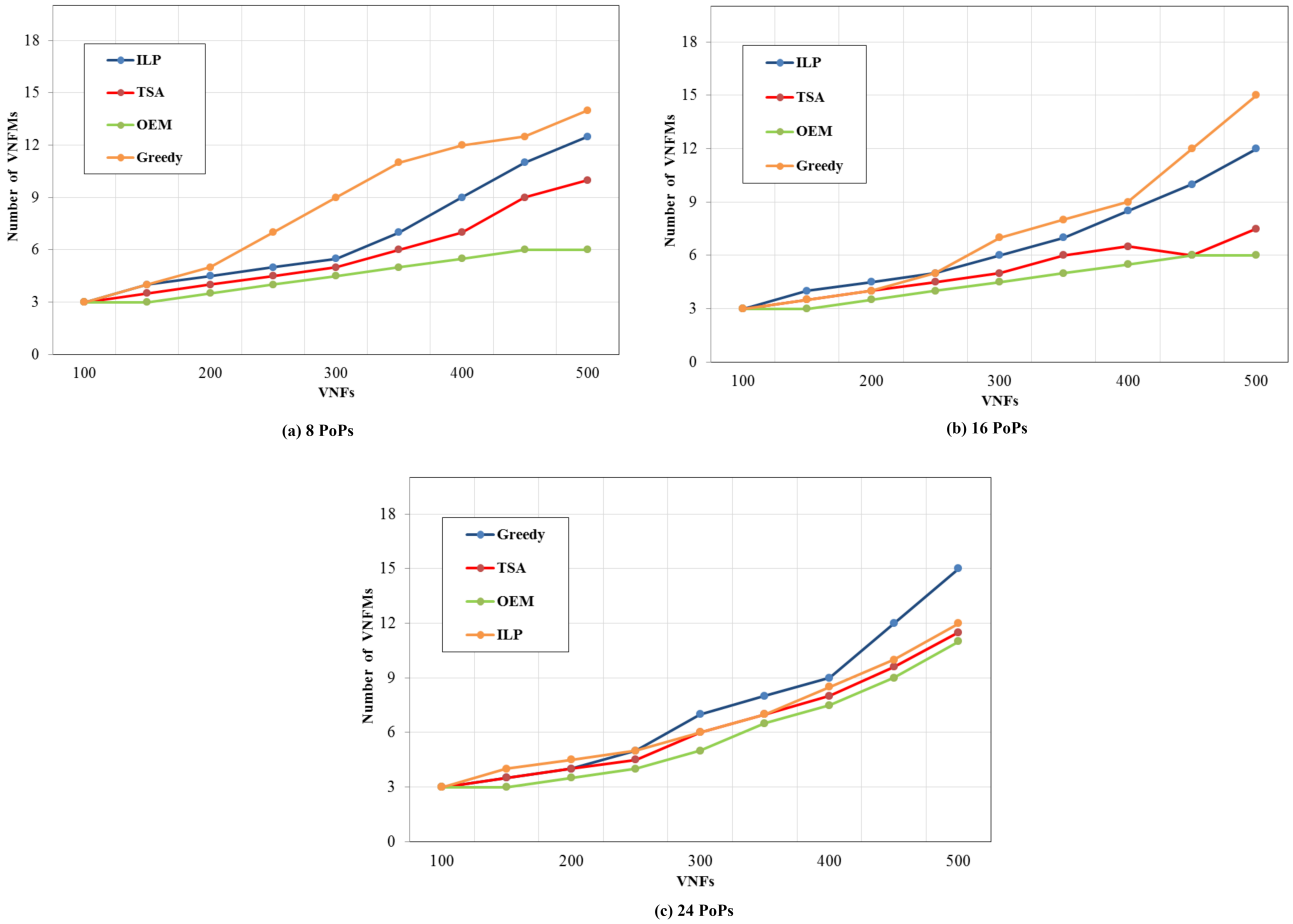


**Figure 7.** Number of VNFMs placed in the NFVI with respect to the number of VNF instances: (**a**) 8 PoPs, (**b**) 16 PoPs and (**c**) 24 PoPs with $\epsilon = 1$ and $\epsilon = 20$.

### *5.6. Distributed Large-Scale Deployment*

Figure 8 represents the total operational cost for the OEM, ILP, TSA and Greedy solutions in consideration of both VNFM-based generic and specific architecture options for $\epsilon = 1$ and $\epsilon = 20$, respectively. In the generic approach, the two distinct sets of VNFMs are placed for managing compute classes C1 and C2. The utilization rate is relatively small, as the compute classes for the number of VNF instances leave many VNFMs with unutilized capacity, leading to resource wastage. Whenever a generic VNFM is adopted, the VNF instances are managed by a single set of VNFMs. This indeed results in maximizing the VNFMs capacity utilization and placement of fewer VNFMs. For the dynamic VNFM placement, the over-provisioning of the VNF instances problem is solved with 225 VNF instances for every VNF class. The figure reports the PoPs locations as per the VNFM architectural options. However, we use the same number of VNF instances to be placed on NFVI for 8, 16, and 24 PoPs. The differences between these PoPs are already experimented with results of average execution time as shown in Table 3. The dynamic placement involves the sum of VNFM compute resource cost and LCM cost. The performance gain in dynamic VNFM placement in a distributed environment is calculated based on the comparison of numerical results between OEM and comparative algorithms with dynamic placement scenarios for 8, 16, and 24 PoPs. The numerical results confirm that the quality of the solutions obtained from the OEM algorithm is significantly higher than other algorithms by 36.34% to 164.8%. The superiority of dynamic VNFM placement emerges by its ability to adapt to the system based on its workloads.

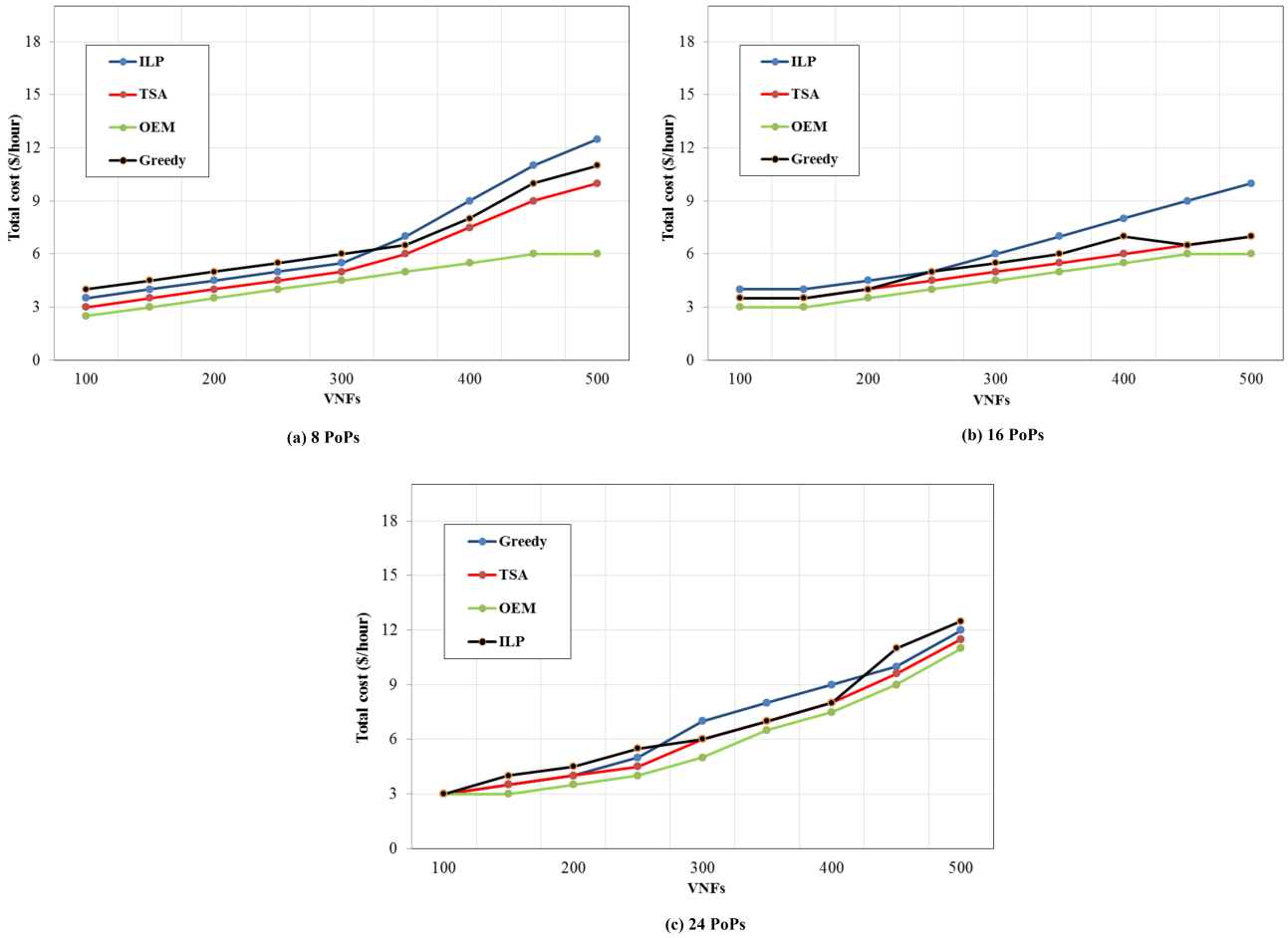


**Figure 8.** Total cost for OEM, ILP, TSA, and Greedy solutions for: (**a**) 8 PoPs, (**b**) 16 PoPs and (**c**) 24 PoPs with $\epsilon = 1$ and $\epsilon = 20$ for the VNFM architectural options: generic and VNF-specific.

## 6. Conclusions and Future Work

This article formulates the telecommunication network-based NFV element management system for edge computing and distributed cloud NFV environments. There is an adverse degradation in performance and persistent rise in CapEx and OpEx when managing an NFV ecosystem. To overcome the above-mentioned issue, we have proposed an OEM algorithm for BGP-EVPN control plane-based VXLAN overlay provisioning for the VNFM placement on NFVI-PoPs. The advent of the new service provisioning BGP-EVPN model for VXLAN is materialized by integrating VTS with OpenStack. The numerical analysis shows that the OEM algorithm gives high-quality solutions, with an average gap of 8% in obtaining an optimal solution. The NFVI-PoP-based VNFM placement decisions show a rapid decrease in overall operational cost. Finally, the dynamic VNFM placement results in adapting the decisions to changes in the NFVI that lead to cost reduction with respect to dynamic placement techniques. In the future, the proposed OEM optimization technique for VNFM placement in the NFV framework is carried out for real-time cloud-native applications with the help of Artificial Intelligence and Machine Learning techniques for VNFM placement, and its workload allocation in an autonomous way will help in automating the NFV Infrastructure through zero touch operations.

**Author Contributions:** Conceptualization, A.A. and G.R.; methodology, A.A. and G.R.; software, K.P. and A.M.; validation, K.P. and A.M.; formal analysis, A.A. and G.R.; investigation, K.P. and A.M.; resources, A.A. and G.R.; data curation, K.P. and A.M.; writing—original draft preparation, A.A. and G.R.; writing—review and editing, K.P. and A.M.; visualization, K.P. and A.M.; supervision, A.A and G.R.; project administration, A.A., G.R., K.P. and A.M.; funding acquisition, K.P. and A.M. All authors have read and agreed to the published version of the manuscript.

**Funding:** Not applicable.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Arzo, S.T.; Bassoli, R.; Granelli, F.; Fitzek, F.H. Study of virtual network function placement in 5G cloud radio access network. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 2242–2259. [CrossRef]
2. Varasteh, A.; Madiwalar, B.; Van Bemten, A.; Kellerer, W.; Mas-Machuca, C. Holu: Power-Aware and Delay-Constrained VNF Placement and Chaining. *IEEE Trans. Netw. Serv. Manag.* **2021**, *18*, 1524–1539. [CrossRef]
3. Németh, B.; Molner, N.; Martinperez, J.; Bernardos, C.J.; De la Oliva, A.; Sonkoly, B. Delay and reliability-constrained VNF placement on mobile and volatile 5G infrastructure. *IEEE Trans. Mob. Comput.* **2021**, *14*, 864–869. [CrossRef]
4. Bunyakitanon, M.; Vasilakos, X.; Nejabati, R.; Simeonidou, D. End-to-end performance-based autonomous VNF placement with adopted reinforcement learning. *IEEE Trans. Cogn. Commun. Netw.* **2021**, *6*, 534–547. [CrossRef]
5. Talluri, L.S.R.K.; Thirumalaisamy, R.; Kota, R.; Sadi, R.P.R.; KC, U.; Naha, R.K.; Mahanti, A. Providing consistent state to distributed storage system. *Computers* **2021**, *10*, 23. [CrossRef]
6. Li, F.; Wang, D. 5G Network Data Migration Service Based on Edge Computing. *Symmetry* **2021**, *13*, 2134. [CrossRef]
7. Dangi, R.; Lalwani, P.; Choudhary, G.; You, I.; Pau, G. Study and investigation on 5G technology: A systematic review. *Sensors* **2021**, *22*, 26. [CrossRef]
8. Tan, M.; Li, G.H.; Yu, H.; Wei, M.; Hu, J.W. Research on Fault Prediction Model Based on 5G Data Center. *J. Internet Technol.* **2021**, *22*, 53–59.
9. Gao, T.; Li, X.; Wu, Y.; Zou, W.; Huang, W.S.; Tornatore, M.; Mukherjee, B. Cost-efficient VNF placement and scheduling in public cloud networks. *IEEE Trans. Commun.* **2021**, *68*, 4946–4959. [CrossRef]
10. Xu, Z.; Zhang, Z.; Lui, J.C.; Liang, W.; Xia, Q.; Zhou, P.; Wu, G. Affinity-Aware VNF Placement in Mobile Edge Clouds via Leveraging GPUs. *IEEE Trans. Comput.* **2021**, *48*, 946–959. [CrossRef]
11. Ochoa-Aday, L.; Cervello-Pastor, C.; Fernandez-Fernandez, A.; Grosso, P. An online algorithm for dynamic NFV placement in cloud-based autonomous response networks. *Symmetry* **2018**, *10*, 163. [CrossRef]
12. Li, D.; Hong, P.; Xue, K.; Pe, J. Virtual network function placement and resource optimization in NFV and edge computing enabled networks. *Comput. Netw.* **2019**, *152*, 12–24. [CrossRef]
13. Patwary, A.A.N.; Naha, R.K.; Garg, S.; Battula, S.K.; Patwary, M.A.K.; Aghasian, E.; Gong, M. Towards secure fog computing: A survey on trust management, privacy, authentication, threats and access control. *Electronics* **2021**, *10*, 1171. [CrossRef]
14. Otokura, M.; Leibnitz, K.; Koizumi, Y.; Kominami, D.; Shimokawa, T.; Murata, M. Evolvable virtual network function placement method: Mechanism and performance evaluation. *IEEE Trans. Netw. Serv. Manag.* **2019**, *16*, 27–40. [CrossRef]
15. Pei, J.; Hong, P.; Xue, K.; Li, D. Efficiently embedding service function chains with dynamic virtual network function placement in geo-distributed cloud system. *IEEE Trans. Parallel Distrib. Syst.* **2018**, *30*, 2179–2192. [CrossRef]
16. Pei, J.; Hong, P.; Xue, K.; Li, D.; Wei, D.S.; Wu, F. Two-phase virtual network function selection and chaining algorithm based on deep learning in SDN/NFV-enabled networks. *IEEE J. Sel. Areas Commun.* **2020**, *38*, 1102–1117. [CrossRef]
17. Chegini, H.; Naha, R.K.; Mahanti, A.; Thulasiraman, P. Process automation in an IoT–fog–cloud ecosystem: A survey and taxonomy. *Internet Things* **2021**, *2*, 92–118. [CrossRef]
18. Yue, Y.; Cheng, B.; Liu, X. Resource Optimization and Delay-aware Virtual Network Function Placement for Mapping SFCRequests in NFV-enabled Networks. In Proceedings of the IEEE 13th International Conference on Cloud Computing, Beijing, China, 19–23 October 2020; Volume 13, pp. 267–274.
19. Tong, M.; Wang, X.; Li, S.; Peng, L. Joint Offloading Decision and Resource Allocation in Mobile Edge Computing-Enabled Satellite-Terrestrial Network. *Symmetry* **2022**, *14*, 564. [CrossRef]
20. Witanto, E.N.; Oktian, Y.E.; Lee, S.G. Toward Data Integrity Architecture for Cloud-Based AI Systems. *Symmetry* **2022**, *14*, 273. [CrossRef]
21. Alsaih, M.A.; Latip, R.; Abdullah, A.; Subramaniam, S.K.; Ali Alezabi, K. Dynamic job scheduling strategy using jobs characteristics in cloud computing. *Symmetry* **2020**, *12*, 1638. [CrossRef]

22. Sun, J.; Huang, G.; Sun, G.; Yu, H.; Sangaiah, A.K.; Chang, V. A Q-learning-based approach for deploying dynamic service function chains. *Symmetry* **2018**, *10*, 646. [CrossRef]
23. Duan, K.; Fong, S.; Siu, S.W.; Song, W.; Guan, S.S.U. Adaptive incremental genetic algorithm for task scheduling in cloud environments. *Symmetry* **2018**, *10*, 168. [CrossRef]
24. Lebdeh, M.A.; Naboulsi, D.; Glitho, R.; Tchouati, C. W. On the placement of VNF managers in large-scale and distributed NFV systems. *IEEE Trans. Netw. Serv. Manag.* **2017**, *14*, 875–889. [CrossRef]
25. Cho, D.; Taheri, J.; Zomaya, A.Y.; Wang, L. Virtual network function placement: Towards minimizing network latency and leadtime. *IEEE Int. Conf. Cloud Comput. Technol. Sci.* **2017**, *4*, 90–97.
26. Ahvar, S.; Phyu, H.P.; Buddhacharya, S.M.; Ahvar, E.; Crespi, N.; Glitho, R. CCVP: Cost-efficient centrality-based VNF placement and chaining algorithm for network service provisioning. In Proceedings of the IEEE Conference on Network Softwarization, Bologna, Italy, 3–7 July 2017; Volume 11, pp. 1–9.
27. Khebbache, S.; Hadji, M.; Zeghlache, D. Scalable and cost-efficient algorithms for VNF chaining and placement problem. In Proceedings of the 20th Conference on Innovations in Clouds, Internet and Networks, Paris, France, 7–9 March 2017; Volume 18, pp. 92–99.
28. Pham, C.; Tran, N.H.; Ren, S.; Saad, W.; Hong, C.S. Traffic-aware and energy-efficient vNF placement for service chaining: Joint sampling and matching approach. *IEEE Trans. Serv. Comput.* **2017**, *13*, 172–185. [CrossRef]
29. Chegini, H.; Mahanti, A. A framework of automation on context-aware internet of things (IoT) systems. In Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing Companion, Auckland, New Zealand, 2–5 December 2019; pp. 157–162.
30. Khan, M.; Alhumaima, R.S.; Al-Raweshidy, H.S. QoS-aware dynamic RRH allocation in a self-optimized cloud radio access network with RRH proximity constraint. *IEEE Trans. Netw. Serv. Manag.* **2017**, *14*, 730–744. [CrossRef]
31. Liu, J.; Lu, W.; Zhou, F.; Lu, P.; Zhu, Z. On dynamic service function chain deployment and readjustment. *IEEE Trans. Netw. Serv. Manag.* **2017**, *14*, 543–553. [CrossRef]
32. Nguyen, V.G.; Brunstrom, A.; Grinnemo, K.J.; Taheri, J. SDN/NFV-based mobile packet core network architectures: A survey. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1567–1602. [CrossRef]