Evolutionary Multi-objective Multicast Virtual Network Function Placement in NFV-SDN Networks

Luis G. Moré
Universidad Nacional de Asunción
Facultad Politécnica
Asunción, Paraguay
lmore@pol.edu.py

and

Carlos Cañete, Crishian Medina
Universidad Nacional de Asunción
Facultad Politécnica
Asunción, Paraguay
{*carlitoscanete97,cdmedina.py*}*@fpuna.edu.py*

and

José Colbes, Diego P. Pinto-Roa
Universidad Nacional de Asunción
Facultad Politécnica
Asunción, Paraguay
{*jcolbes,dpinto*}*@pol.una.py*

Abstract

Software-defined networking (SDN) and network functions virtualization (NFV) represent promising technologies for on-demand services that require the development of flexible multicast transmission mechanisms incorporating data processing functions at the network nodes. The multicast routing problem in NFV-SDN networks aims to compute multicast routing trees and deploy virtual network functions (VNFs) to satisfy traffic demands while optimizing resource utilization and ensuring equitable data transmission. Due to the inherent computational complexity and the presence of conflicting objective functions, this paper formulates multicast routing and VNF placement as a multi-objective optimization problem (MOP), specifically minimizing simultaneously the total resource cost, the maximum transmission delay variance, and the number of multicast request blocked. In this context, this study develops solutions based on Multi-objective Evolutionary Algorithms (MOEAs) that seek to compute the optimal set of non-dominated solutions that form the Pareto set. Simulations performed on test instances demonstrate the efficacy of the proposed methodologies, yielding efficient and non-dominated solutions when compared to a state-of-the-art single-objective approach.

Keywords: Multicast Routing, Software-Defined Networks, Network Functions Virtualization, Multi-objective Evolutionary Algorithms

1 Introduction

Packet transmission based on *Multicast Protocols* are designed aiming at distribution of data from a source to a group of destinations simultaneously. This is a very widely used technique for diverse applications on Internet, such as live video streaming, multimedia distribution, multi-player games, and Financial Data

Dissemination. Additionally, this approach is used in modern data centers, where *Multicast* is leveraged for tasks such as archive sharing, parallel distributed processing, and other critical tasks.

While *Multicast Protocols* have been available since many years ago [1], their widespread adoption in data centers has been hindered because of challenges about security, routing management, scalability, latency, and the complexity associated with routing trees [2].

To address these challenges and overcome limitations of traditional *IP Multicast Protocols*, *Software-Defined Networking* (SDN) [3] has emerged as a promising architectural paradigm. SDN decouples the control plane from the data plane, centralizing network management in programmable controllers. This architectural shift enables the use of standard application pgrogramming interfaces, providing a global view of network resources, and allowing rapid and efficient policy application [3].

Network operators typically deploy network functions tailored to specific traffic classes. In traditional deployments using middle-boxes, this process requires vendor-specific configuration and management procedures, including reconfiguration of switches, routers and network connectivity [4]. There is a need of decoupling service deployment from the underlying hardware infrastructure, thereby streamlining service planning and implementation. This is particularly advantageous given the prevailing reliance on static forwarding rules, which are inherently insensitive to dynamic user requirements and traffic patterns as seen in *multicast services*.

The implementation of multicast services in SDN environments requires careful consideration of various network services, including firewalls, load balancers, Network Address Translators (NAT), and others. These services, known as *middle-boxes* [5], are typically implemented using specialized network devices, requiring manual configurations. Services provided are usually strongly tied to specific middle-boxes, and accommodating middle boxes are increasingly challenging. Problems include manual configurations, error prone procedures and fail to effectively leverage the underlying infrastructure [3].

Network Functions Virtualization (NFV) [5] presents a compelling approach to address these problems. By enabling software implementation of network functions on commercial hardware, NFV achieves abstraction and consolidation of *Physical Network Functions* (PNF) into *Virtual Network Functions* (VNF). This approach leverages the capabilities of commercial-off-the-shelf (COTS) network devices [5], and helps reduce CAPital EXpenditure (CAPEX) and OPeration EXpenses (OPEX).

Multicast transmission within NFV-SDN enabled infrastructures offers a promising alternative to traditional multicast networks. This new approach addresses flexibility and scalability concerns by leveraging the efficient management capabilities provided by NFV and SDN. Nevertheless, this introduces new challenges, primarily related to computational complexity. The integration of multicast with SDN-NFV further complicates the problem, because of the additional requirement of VNF placement within the network. This combined problem is known as *Multicast Sessions Virtual Network Function Placement* (MSVNFP) [6].

Some challenges related to the MSVNFP problem are listed below:

- Latency Optimization: Multicast services often have stringent latency requirements. Placing VNFs to minimize end-to-end latency for all multicast receivers, is a critical challenge. This involves balancing the placement of VNFs closer to sources versus receivers, considering network congestion and processing delays.

- Resource Allocation and Management: Efficient allocation and management of network and computational resources (CPU, memory, storage) for VNFs is crucial. This includes dealing with resources contention, and ensuring fair resources distribution among multiple multicast sessions.

- Control Plane Overhead: The SDN control plane must manage the dynamic placement and configuration of VNFs, as well as the multicast forwarding paths. Minimizing the control plane overhead is crucial for ensuring network scalability and performance.

- Energy Efficiency: Optimizing VNF placement and resource allocation for energy efficiency is increasingly important, especially in large-scale deployments.

In a recent study [7], MSVNFP problem has been addressed considering a NFV-SDN context, incorporating diverse configurations and capacity constraints. However, the prior work has addressed MSVNFP as a single-objective optimization problem, focusing solely on cost minimization. To achieve a more realistic model of real-world deployments, a multi-objective approach in MSVNFP must be considered, incorporating relevant objective functions.

Multi-Objective Evolutionary Algorithms, (MOEAs) [8] present a valuable approach for addressing multi-objective optimization problems, such as MSVNFP. In this context, this study proposes a two-fold contribution which were not proposed in the literature:

- The formal definition of the MSVNFP as a multi-objective optimization problem considers objective functions: the total operative cost, fairness of traffic delay, and the number of blocked multicast requests.
- The application of MOEAs considering the multi-objective formulation of the MSVNFP problem.

This manuscript is organized using the following structure: Section 2, presents a review of pertinent literature related to MSVNFP problem, identifying the limitations and challenges present in the existing approaches. Section 3 provides a general description of *Multi-objective Optimization Problems* (MOP) and MOEAs. Section 4, provides a detailed formulation of MSVNFP as a MOP, including defined variables, constraints and objective functions. Section 5 describes the algorithms employed to efficiently address the problem with heterogeneous VNF configurations, focusing on the application of MOEAs. Section 6 presents a performance evaluation of proposed algorithms using numerical experimentation, comparing their efficiency and efficacy against the existent solution. Finally section 7 summarizes the conclusions of this study, discussing contributions, limitations and future research topics in this domain.

2 Related Works

Given that *multicast* within SDN is a NP-complete problem [9], numerous mechanisms and algorithms have been proposed. In [2], the *Avalanche Routing Algorithm* (AvRA) was developed to minimize the routing tree size generated for any given *multicast* group. Notably, within tipical data center topologies such as Tree and FatTree, AvRA reduces to an optimal routing algorithm, effectively transforming the problem into the well-known Steiner Tree problem.

The Avalanche framework also implements an Openflow controller module which leverages path diversity for efficient resource utilization. This module enforces admission control, constructs near-optimal *multicast* trees, facilitates *multicast* compatibility with basic SDN switches, and dinamically adapts to new group members by identifying the nearest intersection point within the existing tree. Aditionally, ATHENA [10] presents another SDN-based multicast platform designed to allow *multicast protocol* utilization for inter-application group communtication within data centers. The ATHENA implementation offers guaranteed accountability, *statelessness* operation and compatibility with the TCP protocol.

Huang et al. [11] designed and implemented a routing system leveraging SDN and sFlow technologies. This routing system's functionality is integrated in the SDN controller layer.

In [12] the scalability of Group Tables within large SDN deployments is considered, particularly concerning the problem to exceed *Ternary Content-Addressable Memory* (TCAM) capacity due to numerous *multicast* trees. To mitigate this issue, a *Branch forwarding* technique was proposed. This technique optimizes storage by maintaining *multicast* forwarding entries exclusively at *branch* nodes, instead of using every node within each *multicast* tree.

In [13] a large-scale industrial IoT *multicast* architecture is presented. In [14] is proposed a locality-aware *multicast* approach (LAMA) for constructing shared SDN trees, where each shared tree serves several *multicast* groups.

Tao et al. [15] considered the *transmission delay* as a critical performance metric. UP-Jia, a novel distributed heuristic *multicast* algorithm is developed based upon the existing Jia algorithm and incorporating Quality of Service (QoS) considerations. Up-Jia construct *multicast* trees with reduced cost while adhering to specified *transmission delay* constraints. Furthermore, this research addresses the determination of shortest paths between all source and destination node pairs. In order to minimize *delay*, the methodology iteratively computes route paths from each source to all other destination nodes, rather than focusing on single source-destination node pairs. A subsequent path recalculation is performed to identify optimal routes, following by cycle deletion, while considering accumulated *delay* from the source to each destination.

The *Delay Constraint Least Cost Problem* (DCLC), which focuses on finding minimum-cost solution under *delay* constraints, is addressed in [16]. Considering the NP-Completeness of the DCLC problem [17], the authors proposed an approximation algorithm employing a teaching-learning-based approach (TLBO).

In [16] the problem of Quality of Service (QoS) provisioning for multimedia applications is investigated. The proposed algorithmic approach assigns link costs based on link utilization. Specifically, higher link utilization results in higher assigned link cost. Consequently, when the Teaching-Learning-Based Optimization (TLBO) algorithm is applied to solve the Delay Constrained Least Cost (DCLC) problem, it selects paths traversing less congested links. In [18], the construction of *multicast* trees was investigated using Dijkstra's algorithm, with a comparative analysis against Tabu Search. The study focused on evaluating the trade-off between solution quality and computation time for each approach.

Shen et al. [19] proposed a novel reliable *multicast* tree construction method for SDN, called *Recovery-aware Steiner Tree* (RST). Given a source node, a set of destination nodes within a *multicast* group, a set

of candidate recovery nodes within the network, and a non-negative integer r, the RST problem aims to identify a *multicast* tree that fulfills two criteria: (1) connectivity between the source and all destination nodes, and (2) inclusion of at most r recovery nodes as intermediate nodes in the tree, thereby providing resilience against local packet losses. The objective of RST problem is to minimize both the total cost of the tree, satisfying connectivity criterion (1), and the associated recovery cost for the resulting tree. Considering that finding a RST is a difficult task, an approximation algorithm called *Recover Aware Edge Reduction Algorithm* (RAERA) is proposed to address this problem.

A scalable multicast tree group management framework based on SDN and NFV is proposed in [20] for *Internet Services Provider* (ISP) networks. To facilitate the implementation and deployment of multicast services on edge networks, a multicast load-balancing routing algorithm is proposed, named *Lazy Load Balancing Multicast* (L2BM).

A multicast mechanism for NFV within SDN is proposed in [6], addressing both dynamic and static scenarios. This approach employs dynamic heuristics to manage users which join/abandon multicast sessions. This heuristic manages multiple concurrent multicast sessions, prioritizing efficient path management by assigning lower usage probabilities to more congested links. In [21], three heuristics are proposed to address the *Service Function Multicast Problem* (SFMP) considering both static or dynamic deployment scenarios. Within this approach, the static heuristic is applied to each multicast session. Subsequently, the dynamic heuristic is invoked in response to user join/leave events. Concurrently, the scalable heuristic manages requests for the addition or removal of functions within multicast sessions, thereby addressing evolving user requirements.

The *Multicast-oriented Virtual Network Function Placement* (MVNFP) is investigated by Wang et al. [22]. A two-step approach is presented to calculate the lesser propagation delay considering the summation of introduced delays because of links and virtual functions. The first step involves the construction of a multicast tree for the multicast request; the second step entails the placement of the VNF chain along each branch of the constructed tree. The first step employs Dijkstra's algorithm to compute paths from source to destination with minimal link propagation delay. The second step involves a *modified Genetic Algorithm* (mGA) with problem-specific chromosome encoding, crossover and mutation operators, aiming to minimize virtual functions processing delays.

The simulation results demonstrate a superior performance of mGA compared to other metaheuristic evolutionary algorithms, such as Ant Colony Optimization (ACO) [23] and Particle Swarm Optimization (PSO) [23] in terms of both solution quality and convergence. It is important to mention that mGA implementation considered a single multicast session.

The solution design for managing multiple concurrent multicast sessions within a VNF environment remains a pertinent research field due to the scarce number of existing proposals available in the state-of-the-art literature [6, 21, 22, 24–28].

Prior research works related to multicast within VNF has often focused on single-objective optimization context, such as minimizing link costs [5, 6, 21, 24, 25, 27–29], or minimizing link delays [22]. Recently, this research line has broadened to incorporate constraints related to maximum link capacity and maximum computational capacity at VNF nodes [7]. In [30], an ILP model is formulated to minimize resource utilization, while considering node, link, and delay constraints, for multiple multicast requests. While the objective function incorporates VNF instantiation costs, processing costs, and bandwidth consumption, these metrics are aggregated into a single-objective optimization problem.

In [31], an Integer Linear Programming (ILP) model is likewise formulated to address VNFP within multicast scenarios. This solution employs a Two-Stage SDV-based (TSDV) methodology, explicitly acknowledging a trade-off between efficiency and complexity.

This literature review indicates the need to develop multicast tree construction methodologies considering a multi-objective optimization problem (MOP) context, recognizing the inherent conflict between the total operative cost, the fairness of delay, and the number of blocking requests. In this context, the study proposed addressing the MSVNFP problem in a multi-objective optimization framework, extending the proposal presented in [7] under MOEA strategies.

3 Multi-Objective Optimization

3.1 Multi-Objective Optimization Problem

Multi-Objective Optimization problems, such as the problem addressed in this manuscript, are characterized by the presence of two or more conflicting objectives, wherein improvement in one objective results in the degradation of other objectives. While these problems present significant computational challenges, they are of realistic nature.

In general terms, a *Multi-objective Optimization Problem* (MOP) can be defined as follows [32]: Given $\mathtt{m}$ objective functions $OF_1 : \chi \to \mathcal{P}, ..., OF_{\mathtt{m}} : \chi \to \mathcal{P}$, which map a decision space χ to the objective space $\mathcal{P}$, then a MOP is given by the following definition:

$$OF = \text{MIN}(OF_1(\mathbf{x}), ..., OF_{\mathtt{m}}(\mathbf{x})), \tag{1}$$

subject to the following constraints:

$$g_j(\mathbf{x}) \leq 0, j = 1, 2, ..., g \tag{2}$$

Where $\mathbf{x} \in \chi$ represents a $\mathtt{n}$-variable decision vector $\mathbf{x} = (x_1, x_2, ..., x_{\mathtt{n}})^T$.

This definition does not imply a loss of generality, considering that function maximization can be readily transformed to minimization.

The elements $\mathbf{x}$ constitute a set of decision variables representing candidate solutions. Optimality for these solutions is defined in terms that a given solution may be considered optimal with respect to certain objective functions OF_i, while simultaneously being suboptimal when considering other objective functions OF_j. Consequently, simultaneous improvement across all objectives is not attainable.

This concept is formally defined as *Pareto Optimality*: A solution $\mathbf{x} \in \chi$ is Pareto-Optimal with respect to χ if and only if there does not exist another solution $\mathbf{x'} \in \chi$ for which $OF(x') = (OF_1(x'), ..., OF_m(x'))$ dominates $OF(x) = (OF_1(x), ..., OF_m(x))$.

The dominance relation between two solution vectors, $\mathbf{x}$ and $\mathbf{x'}$, denoted as $\mathbf{x} \succeq \mathbf{x'}$, is formally defined as follows: $\mathbf{x}$ dominates $\mathbf{x'}$ if and only if $\mathbf{x}$ is partially less than $\mathbf{x'}$. This condition is satisfied if and only if, the objective value of $\mathbf{x}$ is less or equal to the corresponding value of $\mathbf{x'}$, i.e., $\forall i \in 1, 2, ...m, OF_i(x) \leq OF_i(x')$, and there exists at least one objective function OF_i for which the objective value of $\mathbf{x}$ is strictly less than the corresponding value of $\mathbf{x'}$, i.e., $\exists i \in 1, 2, ...m : OF_i(x) < OF_i(x')$. Based in this definition, the *Optimal Pareto Set*, denoted as $\mathcal{P}^*$, for a MOP and its associated objective function vector $OF(x)$, is defined as follows:

$$\mathcal{P}^* = \{\mathbf{x} \in \chi \mid \nexists\ \mathbf{x'} \in \chi : OF(\mathbf{x'}) \preceq OF(\mathbf{x})\} \tag{3}$$

The optimal Pareto set, denoted as $\mathcal{P}^*$, is mapped into the objective space of the MOP, and the resulting image is the Pareto Front:

$$\mathcal{PF}^* = \{OF(\mathbf{x}) : \mathbf{x} \in \mathcal{P}^*\} \tag{4}$$

Figure 1 presents an example of the above concepts considering two decision variables and two objective functions. The figure shows a Pareto set $\mathcal{P}^*$ in the decision variable space and a Pareto front $\mathcal{PF}^*$ in the objective function space.

The black circles ($\bullet$) are non-dominated solutions, while the white circles ($\circ$) are the dominated solutions.

The non-dominated solutions $\mathbf{x}_a$ to $\mathbf{x}_e$ belong to the Pareto set, whereas their objective function vector $OF(\mathbf{x}_a) = [OF_1(\mathbf{x}_a), OF_2(\mathbf{x}_a)]$ to $OF_{\mathbf{x}_e} = [OF_1(\mathbf{x}_e), OF_2(\mathbf{x}_e)]$ forms the Pareto front $\mathcal{PF}^*$.

Note that, the solutions $\mathbf{x}_f$, $\mathbf{x}_g$, $\mathbf{x}_h$, $\mathbf{x}_i$, y $\mathbf{x}_j$ are dominated solutions by $\mathbf{x}_d$. In particular, $OF_1(\mathbf{x}_d) < OF_1(\mathbf{x}_i)$ and $OF_2(\mathbf{x}_d) < OF_2(\mathbf{x}_i)$, therefore $\mathbf{x}_d \succeq \mathbf{x}_i$.

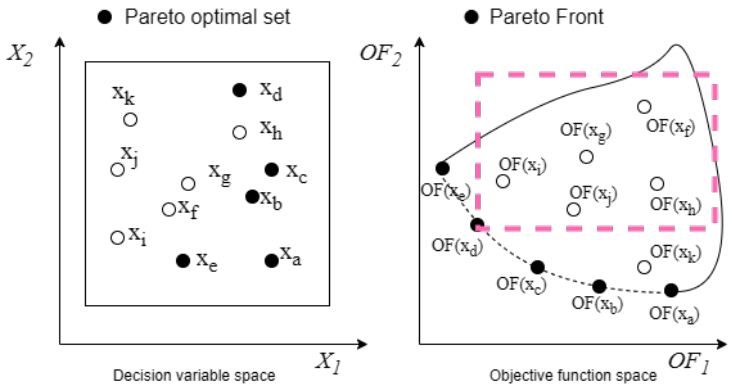

Figure 1: Example of a Pareto set and Pareto front.

This study focuses on obtaining sufficiently accurate approximations of $\mathcal{P}^*$ for the MSVNFP considering an MOP context.

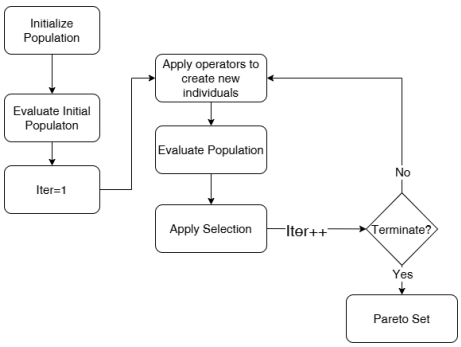

Figure 2: A general overview of MOEA operation.

3.2 Multi-Objective Evolutionary Algorithms

MOEAs are computational methods designed for the non-linear solution of MOPs. Their significance stems from their capacity to operate a set of candidate elements (population). This population-based approach enables the simultaneous generation of a set of solutions, thereby facilitating the efficient approximation of the Pareto set, often achieving faster convergence when compared to traditional mathematical programming techniques.

Figure (2) illustrates the general operational framework of MOEAs for solving MOPs. MOEAs perform computations based on an operational unit named an Individual, which represents an encoded candidate solution for the MOP. Each individual encodes a candidate solution, typically using a string representation designated as Chromosome. The collection of these individuals is named Population. This population serves as the basis for generating novel solutions which are expected exhibit improved characteristics as candidate solutions. Individuals are selected for the generation of new solutions through the application of operators named Selection, Crossover and Mutation. The Selection is predicated on the objective function vector, $OF(\mathbf{x})$, defined for the MOP. Individuals exhibiting superior $OF(\mathbf{x})$ values are preferentially selected as progenitors for subsequent generations, thus forming new populations. There exists a wide range of problems that can be addressed using frameworks based in MOEA. Several relevant problems and associated algorithms are described or analyzed in the literature, such as the works presented in [33], [8], [34].

4 Formal Definition of the Problem

4.1 Problem Statement

For a comprehensive understanding of the MOP-MSVNFP model formulation, Table 1 presents the notation used in this manuscript.

Given the graph $G = (V, E)$ representing a NFV-SDN network, composed of a set of nodes V and a set of non-directed links E, let $H \subseteq V$ denote the set of candidate nodes for VNF deployment, F represent the set of available VNF technologies, and M represent the set of multicast VNF requests.

Each multicast request $m \in M$ is defined as the tuple $m = (s_m, D_m, w_w, f_m)$ where $s_m \in V$ denotes the source node, $D_m \subset V$ represents the set of destination nodes, $w_m \in \Re^+$ signifies the bandwidth of traffic to be transmitted by s_m, and $f_m \in F$ the type of VNF required by m.

The data flow, or traffic, associated with each multicast session must originate from s_m and be transmitted to each destination $d \in D_m$, traversing a node $h \in H$ on which a VNF f_m is deployed.

Depending of the type of deployed VNF, the outbound bandwidth of the VNF-hosting node is given by $\alpha \cdot w_m$, where $\alpha \in [0, T]$ and w_m represents the inbound bandwidth to the node.

In order to define the cost associated with a multicast topology, it is assumed that each link $e \in E$ has an associated non-negative cost gw_e of utilization, a non-negative traffic capacity gc_e, and a non-negative transmission delay gd_e. The cost of activation and computational capacity of VNF-enabled nodes are denoted by $Cost_h$ and Cap_h, respectively.

For the addressed MOP-MSVNFP, multiple instances of VNFs are deployed for each multicast session. Each instance exhibits different computational processing costs for the network function, per session.

Given the data rate requirements for each multicast session, the objective of solving the MOP-MSVNFP problem is to minimize the total cost of utilized link and the activation costs of VNF-enabled nodes across all multicast sessions. The solution comprises a multicast tree for each session, that connects a source node to all its corresponding destination nodes, wherein each branch of the multicast tree includes a VNF-deployed node prior to reach each destination node.

Symbol	Explanation
G	Network Graph, G= (V, E)
V	Node set, $V = \{v_1, v_2, \ldots, v_{\mathcal{V}}\}$
E	Link set, $E = \{e_1, e_2, \ldots, e_{\mathcal{E}}\}$ where $e = (v_i, v_j)$
H	Set of candidate nodes for VNF Deployment, H=$\{h_1, h_2, \ldots, h_{\mathcal{H}}\} \subseteq V$
F	Set of VNF technologies, $F = \{f_1, f_2, \ldots, f_{\mathcal{F}}\}$
gw_e	Link e Utilization Costs
gd_e	Transmission delay costs for link e
gc_e	Link e capacity
$Cost_h$	VNF Activation Costs for node h
Cap_h	Computational capacity for a VNF-capable node h
s_m	Origin node for request m, $s_m \in V$.
D_m	Destination set for request m, $D_m = \{d_1, d_2, \ldots, d_{\mathcal{D}_m}\} \subset V$.
w_m	Bandwidth transmitted from the origin node for request m.
f_m	VNF type required for request m, $f_m \in F$.
z_m	Computational capacity of required VNF by request m.
M	Multicast request set $M = \{m_1, m_2, \ldots, m_{\mathcal{M}}\}$, where a request is defined as $m = (s_m, D_m, w_m, f_m, z_m)$.
$\alpha_m \cdot w_m$	Bandwidth rate after traversing a VNF-capable node with VNF of type f_m.
$\mathfrak{S}$	Solution for instance M, $\mathfrak{S} = \{S_1, S_2, \ldots, S_{\mathcal{M}}\}$.
S_m	Solution for request m, $S_m = (t_m, n_m)$
t_m	Multicast tree for request m, $t_m = (V_m, E_m)$, $V_m \subset V, E_m \subset E$.
n_m	Node set in t_m with VNF implementation, $n_m \subset H$.
$p_m^{s,d}$	Path or branch of tree t_m which connects node s and node d of request m, $p_m^{s,d} \subset t_m$.
$\Delta(p_m^{s,d})$	Delay of a path.
x_m^h	Binary variable which denotes VNF activation on node h for request m (1= VNF is deployed; 0= VNF not deployed).
$y_m^{e,d}$	Binary variable indicating link utilization e on path $p_m^{s,d}$ (1= link is utilized, 0= link not utilized)
$y1_m^e$	Binary variable which denotes link e utilization on tree t_m prior to traversing a VNF-capable node (1= link is utilized, 0= link not utilized).
$y2_m^e$	Binary variable indicating link e utilization on tree t_m after traversing a VNF-capable node (1=link is utilized, 0=link not utilized).
b_m	Binary variable which denotes a successful computation of tree t_m (1= request is accepted, 0= request is blocked).

Table 1: Notation of the MOP-MSVNFP model.

The aggregate traffic flow on each link must not exceed the link's total capacity. Similarly, the computational load imposed by deployed VNFs within each candidate node must not exceed the node's computational capacity.

4.2 MOP-MSVNFP Formulation

Based on the notation defined in Section 4.1, the following formulation for the MSVNFP problem in a MOP context is proposed.

Objective Function Vector:

$$OF = \text{MIN } (OF_1, OF_2, OF_3) \tag{5}$$

where:

- $OF_1 = Cost_E + Cost_{VNF}$ represents the total cost, encompassing both link utilization costs and VNF node activation costs, and is calculated as follows:

$$Cost_E = \sum_{\forall m} \sum_{\forall e} gw_e \cdot (y1_m^e + y2_m^e), \tag{6}$$

$$Cost_{VNF} = \sum_{\forall m} \sum_{\forall h} Cost_h \cdot x_m^h. \tag{7}$$

- $OF_2 = \text{MAX}_{\forall m}\{(\sigma_m)^2\}$ represents the maximum delay variance across all the trees obtained from non-blocked requests:

$$(\sigma_m)^2 = \frac{\sum_{\forall d \in D_m} (\Delta(p_m^{s,d}) - \mu_m)^2}{\mathcal{M}}, \tag{8}$$

$$\mu_m = \frac{\sum_{\forall d \in D_m} \Delta(p_m^{s,d})}{\mathcal{D}_m}, \tag{9}$$

$$\Delta(p_m^{s,d}) = \sum_{\forall e} \sum_{\forall d \in D_m} y_m^{e,d} \cdot gd_e \tag{10}$$

  The objective is to minimize the maximum delay variance, which correlates with improved fairness in data traffic delay across all destination nodes. A higher variance indicates disparity in delays that some destination nodes experience, with some nodes receiving data with significantly greater latency than others.

  Suppose we have three requests $M = \{m_1, m_2, m_3\}$ whose respective trees t_{m_1}, t_{m_2}, and t_{m_3} make up the solution. Suppose also that each tree has three destinations $|D_{m_j}| = 3$ and consequently of three paths $p_{m_j}^{s,d}$ that connect the source node $s \in m_j$ to its destinations $d \in D_{m_j}$. If the path delays for each tree is defined by the following vectors $\Delta(p_{m_1}^{s,d}) = [2, 2, 2]$, $\Delta(p_{m_2}^{s,d}) = [5, 2, 1]$, and $\Delta(p_{m_3}^{s,d}) = [5, 5, 1]$, then the average delays are $\mu_{m_1} = 2$, $\mu_{m_2} = 2.67$, $\mu_{m_3} = 3.67$ and the delay variances $(\sigma_{m_1})^2 = 0$, $(\sigma_{m_2})^2 = 4.33$, and $(\sigma_{m_3})^2 = 5.33$. In this context, $OF_2 = \text{MAX}\{0, 4.33, 5.33\} = 5.33$. Note that t_{m_3} has higher inequality in the delays to destinations.

- $OF_3 = \sum_{\forall m} (1 - b_m)$ is the number of blocked multicast requests.

  Suppose we have three requests $M = \{m_1, m_2, m_3\}$ where the algorithm only calculated two trees t_{m_1} and t_{m_2}. The t_{m_3} tree could not be calculated due to the unavailability of network resources. In this context, the decision variable vector $b = [b_1, b_2, b_3] = [1, 1, 0]$ indicates that only requests m_1 and m_2 are solvable while request m_3 was blocked. In this context, $OF_3 = (1-b_1)+(1-b_2)+(1-b_3) = 0+0+1 = 1$ indicates that one request was blocked.

Each OF defined is subject to the following restrictions:

- Path existence from source node to destination nodes: The flow associated with each request must traverse from the origin node to each requesting destination node.

$$\exists p_m^{s,d} \subset t_m, \forall m, \forall d \in D_m. \tag{11}$$

- Link Capacity Constraint: The aggregate flow traversing any given link must not exceed its maximum capacity:

$$\sum_{\forall m} w_m \cdot y1_m^e + \alpha \times w_m \cdot y2_m^e \leq gc_e, \forall e. \tag{12}$$

- Computational Capacity Constraint: The aggregate computational demand imposed by the VNFs processed at each node h must not exceed the node's computational capacity.

$$\sum_{\forall m} z_m \cdot x_m^h \leq Cap_h, \forall h. \tag{13}$$

- VNF Processing Constraint: The flow associated with each request must be processed by a VNF deployed on a node h prior to reaching destination:

$$\sum_{\forall h \in p_m^{s,d}} x_m^h = 1, \forall m, \forall d \in D_m. \tag{14}$$

A comprehensive example of a single-objective optimization formulation minimizing total cost can be found in García and Pinto [7].

Notice the connection between a solution of the Pareto set and a solution of the addressed problem, this is, $\mathbf{x} = \mathfrak{S}$.

Observations on the model:

- The connection between a solution of the Pareto set and a solution of the addressed problem, i.e., $\mathbf{x} = \mathfrak{S}$.
- The structure and parameters of the network do not change over time, i.e., they are fixed configurations.
- The model supposes static traffic; therefore, the Pareto front is unique for each scenario and does not change over time.

5 Algorithm Proposal

This section presents the general solution approach for the MOEA-based solution to the MSVNFP problem, designated MSVNFP-MOEA.

5.1 General Scheme for MOEA-based MOP-MSVNFP Algorithm

The proposed approach is called MSVNFP-MOEA which receives as input the MOEA evolutionary parameters and the specific parameters to the MOP-MSVNFP problem. The evolutionary parameters include the population size (P_t), the crossover rate (p_c), the mutation rate (p_m), the number of generations ($stop$), and fitness evaluation operators. The problem-specific parameters consist of the network graph (G), the characteristics of the links and server nodes, the set of multicast requests (M), the available VNF technologies, and the set of nodes designated for VNF placement (H).

Algorithm 1 MSVNFP-MOEA

1: **Input**: Evolutionary Parameters, Problem Parameters
2: **Output**: Set of Pareto Solutions
3: $t \leftarrow 0$
4: Generate initial Population P_t
5: Evaluate Objective Functions for P_t
6: Evaluate dominance and fitness of each individual in P_t
7: **while** Stop Criterion is not reached **do**
8: $S \leftarrow$ Apply Selection operator on P_t
9: $C \leftarrow$ Apply Crossover operator on S
10: $M \leftarrow$ Apply Mutation on C
11: Evaluate Fitness Functions of M
12: $R \leftarrow P_t \cup M$
13: Calculate fitness and dominance of R
14: $P_{t+1} \leftarrow$ Select individuals of R
15: $t \leftarrow t + 1$
16: **end while**
17: **Return** Non-dominated Solutions of P_t

The Algorithm commences by generating an initial set of candidate solutions. Subsequently, the objective functions, dominance relationships, and fitness values of this initial set are evaluated (as shown in lines 4 and 5, respectively). The initialization of the set is performed using a stochastic process.

The evolutionary process generates subsequent candidate populations by performing the operations detailed in lines 7-15, iterating until the defined termination criterion is reached. In this work, the termination criterion is defined as a predetermined number of evolutionary generations.

In each generational step, the following processes are performed: selection of parent chromosomes (line 8), which are subsequently employed for the crossover operation (line 9) and thus generating the offspring chromosomes. These newly generated individuals inherit genetic information from the preceding generation. This inheritance can potentially lead to premature convergence or stagnation within the search space. In order to mitigate this problem, the mutation operator is applied (line 10), introducing novel genetic material into the population. Subsequently, the objective functions are evaluated for each new individual (line 11).

The newly generated population and the current population are combined to form a super population, enabling a survival tournament which is performed (line 12). The fitness and subsequent dominance relationships among the individuals within this super set are evaluated to select the fittest individuals, which are then retained for the next evolutionary iteration (line 13). The fitness and dominance of each individual are determined by the specific MOEA strategy adopted. In this work, the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) [35] and Strength Pareto Evolutionary Algorithm 2 (SPEA2) [36] are considered as MOEA strategies. In NSGA-II, fitness is defined by the concept of non-dominated ranks and crowding distance, whereas in SPEA2 fitness is defined by the number of solutions dominated by a given individual.

Upon termination of the evolutionary process, the output consists of the non-dominated solution set, which constitutes the Pareto set.

5.2 Chromosome

The MSVNFP-MOEA chromosome encodes the processing order for multicast sessions and the list of candidate VNF-deployed nodes. This structure was adapted from [7] and comprises two distinct genes: one representing the multicast session processing order and the other chromosome representing the VNF-enabled nodes to be activated, i.e., a routing gene and a location gene, respectively. The routing gene comprises a

permutation vector, while a binary number vector represents the location gene. The permutation vector dictates the processing sequence of requests, whereas the binary vector indicates each node's VNF enablement status.

Consider, as an illustrative example, a NFV-SDN network comprising ten nodes, where nodes three through six are candidates for VNF server deployment, and a set of multicast requests $M = \{m_1, m_2, m_3\}$. Here, a chromosome represented as $[3, 1, 2|0001010000]$ indicates that requests are processed in the sequence m_3, m_1 y m_2, meanwhile the nodes four and six are selected as candidates for the deployment of the required multicast functions.

5.3 Evolutionary Operators

As this proposal employs evolutionary algorithms, the core process relies in the sequential application of selection, crossover and mutation operators (as detailed in lines 8-10 of Algorithm 1). These operators have been adapted from the methodology described in [7].

5.3.1 Selection Operator

For individual selection, a k-ary tournament procedure is utilized. In this procedure, k = 3 individuals are stochastically selected from the population, and the individual exhibiting the highest fitness is designated as a winner. This tournament is conducted twice in order to select two distinct winning individuals, one for each tournament. These selected individuals are then employed as parents for the crossover operator, thus generating new offspring.

5.3.2 Crossover operator

Crossover between two individuals is performed with a probability $p_c = 100\%$. This operator comprises two disctint components: crossover for the routing gene and crossover for the location gene. For both genes, the single-point *Order Crossover* [8] is utilized. Specifically the routing genes of descendant individuals are generated from the routing genes of the parent individuals. An analogous procedure is applied over the location gene for the descendants.

5.3.3 Mutation operator

Mutation is executed immediately following the crossover procedure, with a probability $p_m = 20\%$. This operator mutates an individual's genes in a two-step process: first, the routing gene is mutated, and subsequently the location gene is mutated. The mutation procedure is identical for both genes and consists in a reciprocal exchange strategy [8].

5.4 Objective Functions Calculation

The Objective Functions Evaluator Algorithm computes multicast trees sequentially, adhering to the order defined by the routing gene. For each multicast request, the algorithm constructs the solution tree progressively, connecting each destination node sequentially to the existing multicast tree. Since each branch connecting the source node to a destination node (d_i) must traverse a VNF-deployed node, as determined by the location gene, the following strategy, adapted from [7], is employed:

1. For each node h_j already incorporated within the tree structure, the shortest path from destination node d_i is computed.
2. For each node h_j not yet incorporated into the tree structure, both the shortest path from the destination node d_i and the shortest path from source node are computed.
3. For each destination node d_j ($i \neq j$) already incorporated within the tree structure, the shortest path from d_i is computed.
4. The computed paths are compared, and the path exhibiting the minimal cost is selected.
5. The links comprising the selected path are incorporated into the tree structure, and the capacities of these newly added links are decremented accordingly.
6. The capacities of the newly added links are summed to the tree's cumulative cost.
7. The node h_j belonging to the selected branch is incorporated into the tree as a NVF-enabled node, and its computational capacity is decremented accordingly.

8. The activation cost of the VNF-enabled node h_j is added to the Total Cost of the tree. However, if h_j has been previously incorporated into the tree, its activation cost is not added again.

Prior to commencing the solution computation, destination nodes are ordered in ascending order based on their shortest distance to the origin node. If the computation of a feasible multicast tree is successful, the Evaluator Algorithm returns the following: the multicast tree total cost, the delay variance across the destination nodes, the set of VNF-enabled candidate nodes, the updated link capacities, and the updated computational capacities of the VNF-enabled candidate nodes. In the event that a feasible multicast tree cannot be computed, the Evaluator Algorithm returns a request as blocked. Utilizing this per-request information, it is thus possible to compute the total number of blocked requests, the aggregate cost, and the maximum variance.

6 Experimental Tests

This section presents the results obtained from the conducted experimental tests, which evaluate the performance of the Genetic Algorithm proposed in [7], named MSVNFP-GA, against the proposals developed in this work: MSVNFP-MOEA implementations based on NSGA-II and SPEA2.

It is pertinent to point that both MSVNFP-GA and MSVNFP-MOEA were developed for general applicability to any type of VNF. Particularly, MSVNFP-GA employs a single-objective approach, focusing solely on minimizing total cost.

The empirical tests were conducted using a laptop computer running a Linux operating system (Ubuntu 20.04 LTS) with the following hardware specifications: an Intel Core i7 processor, 16GB RAM. The software development of MOEAs were made using Python 3.8 in conjunction with the Pymoo [37] library for multi-objective optimization.

The evolutionary parameters used in the experiments performed on the NSF network topologies were used in [7]. In this study, they performed experimental tests for their determination. These are shown below:

- Population Size: 100
- Tournament size for selection operator: 3
- Mutation rate: 20%
- Crossover rate: 100%
- Number of generations: 100

The results underwent numerical analysis using standard Python libraries, including Numpy, Paretoset, and Matplotlib, to facilitate visualization of solution convergence and diversity achieved by the algorithms.

The problem-specific parameters for the three VNF types were determined empirically and are detailed as follows:

1. VNF1:
    - Function: Bandwidth preservation ($\alpha = 1$).
    - Function operational cost: 10 units.
2. VNF2:
    - Function: Bandwidth reduction by 30 % ($\alpha < 1$).
    - Function operational cost: 20 units.
3. VNF3:
    - Function: Bandwidth increase by 30 % ($\alpha > 1$).
    - Function operational cost: 5 units.

For the numerical simulations, two kind of tests were conducted, with distinct topologies, as shown below:

1. Test 1 was conducted employing the NSFNet network topology (Figure 3-a). This experiment was designed to ascertain if MOP-MSVNFP is of multi-objective nature, and to provide a comparative analysis against the state-of-the-art GA proposal. In this test, only objective functions FO_1 and FO_2 were considered, and the blocking constraint was relaxed. This methodological choice was made to ensure comparability with the state-of-the-art GA proposal, which does not incorporate blocking considerations.

2. Test 2 was designed to evaluate the efficiency of two competitive state-of-the-art metaheuristics: NSGA-II and SPEA2, in solving MOP-MSVNFP. These evaluations were conducted under a range of scenarios:

   - NSFNet topology (Figure 3-a).
   - EON topology (Figure 3-b).
   - USA topology (Fig. 3-c).

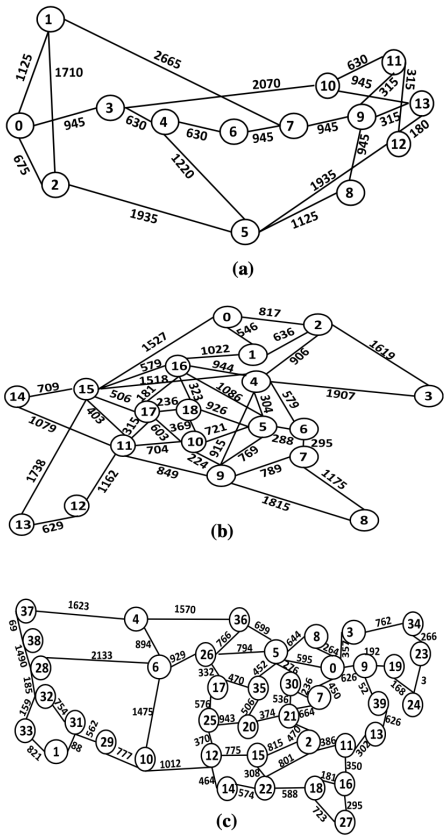

Figure 3: Topologies employed for the tests. a) NSFNet topology (14 nodes, 21 links) b) EON topology (19 nodes, 39 links). c) USA topology (40 nodes, 58 links) [38].

The following points detail the methodology employed in the simulations:

P1. In this initial set of tests, a MOEA was employed to compare its performance against a GA. This comparison aimed to demonstrate the existence of a trade-off relationship within MSVNFP set of solutions, and confirm that the objective functions are contradictory in nature. To this end, Pareto fronts were generated employing NSGA-II exclusively, and the best solution computed by single-objective GA was also obtained. For each traffic load scenario, the algorithms were executed 30 times independently, and the best results obtained were subsequently recorded. The traffic load scenarios are detailed below:

   - Scenario 1: In these tests, it was assumed that 14 requests required the use of VNF1.
   - Scenario 2: In these tests, it was assumed that 14 required the use of VNF2.
   - Scenario 3: In these tests, it was assumed that 14 requests required the use of VNF3.

   For each of the three scenarios, a test environment with homogeneous characteristics on network links was configured:

   - Link costs: Homogeneous and fixed in 50 units.
   - Link capacities: Homogeneous and fixed in 380 bandwidth units.

P2. These test were conducted in order to empirically compare the NSGA-II and SPEA2 metaheuristics, considering the following Pareto Front performance indicators: *Generational Distance* (GD) [39], *Generational Distance Plus* (GD+) [40], *Inverted Generational Distance* (IGD) [41], *Inverted Generational*

Distance Plus (IGD+) [40], *Hypervolume* (H) [42]. The software implementation of these indicators are available within the Pymoo library. For each traffic load scenario the algorithms were executed 30 times independently in order to collect the best results. These traffic load scenarios are detailed below:

(a) NSFNet topology (Figure 3-a), consisting in 14 nodes and 21 links:

- Scenario 1: Tests were conducted using a link capacity of 95 units.
- Scenario 2: Tests were conducted using a link capacity of 190 units.
- Scenario 3: Tests were conducted using a link capacity of 285 units.
- Scenario 4: Tests were conducted using a link capacity of 380 units.

(b) EON topology (Figure 3-b), consisting in 19 nodes and 39 links:

- Scenario 1: Tests were conducted using a link capacity of 50 units.
- Scenario 2: Tests were conducted using a link capacity of 100 units.
- Scenario 3: Tests were conducted using a link capacity of 150 units.
- Scenario 4: Tests were conducted using a link capacity of 200 units.

(c) USA topology (Figure 3-c), consisting in 40 nodes and 58 links:

- Scenario 1: Tests were conducted using a link capacity of 300 units.
- Scenario 2: Tests were conducted using a link capacity of 450 units.
- Scenario 3: Tests were conducted using a link capacity of 600 units.
- Scenario 4: Tests were conducted using a link capacity of 850 units.

In these experimental scenarios every type of VNF node was utilized with equal probability (30%) and the link costs were uniformly set to a constant value of 50 units.

6.1 Analysis of Test 1 experimental results.

The numerical results obtained from the scenarios conducted for the first set of experiments are presented on Figure 4. These scenarios were designed to ensure acceptance of all requests, and therefore resulting in zero blocking events. The axes of the figures represent the total cost and the maximum delay variance. This specific configuration is due to the fact that the state-of-the-art GA algorithm proposed in [7] was not designed to address request blocking events.

Analysis of these results unequivocally demonstrates that the MSVNFP constitutes a multi-objective problem, as evidenced by the effective construction of a Pareto front using NSGA-II. It is also noteworthy that the GA was designed with considerable efficiency, as the achieved solution is within the Pareto front, exhibiting the minimal cost. However, this cost minimization is achieved at a significant trade-off in terms of delay variance. This minimal cost implies that certain destination nodes receive information with a rather elevated delays, exceeding those experienced by other destinations within the same multicast group. Depending on the specific application context, such delay variation among destination nodes within a single multicast group is not desirable.

	GD		**GD+**		**IGD**		**IGD+**		**HV** ($\times 10^9$)	
Load	**NSGA-II**	**SPEA2**	**NSGA-II**	**SPEA2**	**NSGA-II**	**SPEA2**	**NSGA-II**	**SPEA2**	**NSGA-II**	**SPEA2**
95	10.381	119.341	42.570	5548	4.715	121.310	2.758	5,62	451	450
190	213.148	39.991	195.203	0,52	38.689	40.577	0	0	571	522
285	64.548	467	0	104	104.110	11.825	36	0,21	488	495
380	0	130.881	0	33.584	0	204.867	0	0	84	78

Table 2: Average values of performance indices obtained by the proposed MOEAs using the NSFNet topology.

	GD		**GD+**		**IGD**		**IGD+**		**HV** ($\times 10^9$)	
Load	**NSGA-II**	**SPEA2**	**NSGA-II**	**SPEA2**	**NSGA-II**	**SPEA2**	**NSGA-II**	**SPEA2**	**NSGA-II**	**SPEA2**
50	144.864	465.239	0.19	0.23	28395.15	2179.857	2.54	964.38	71	75
100	9216,232	15822.751	11.6	0.83	16017.771	41315.389	3.8	32134.694	155	136
150	11810.338	24970.761	7126.108	23.6	71698.498	17480.341	56633.351	1.345	131	161
200	0	103550.43	0	310	0	116978.675	0	49239.51	45	36

Table 3: Average values of performance indices obtained by the proposed MOEAs using the EON topology.

Load	GD		GD+		IGD		IGD+		HV ($\times 10^9$)	
	NSGA-II	SPEA2	NSGA-II	SPEA2	NSGA-II	SPEA2	NSGA-II	SPEA2	NSGA-II	SPEA2
300	3183.7	9090.1	0.24	0.71	60684.2	63589.76	51030.3	0.64	5	6
450	47442.4	5342.9	0.51	66.7	32074.9	39485.9	8.4	20337.5	5	5
600	70946.7	102.777	35039.5	0.16	21774.6	102.77	21711.4	0.16	2	3
850	0	70	0	70	0	2321587.5	0	2321587.4	2	1

Table 4: Average values of performance indices obtained by the proposed MOEAs using the USA topology.

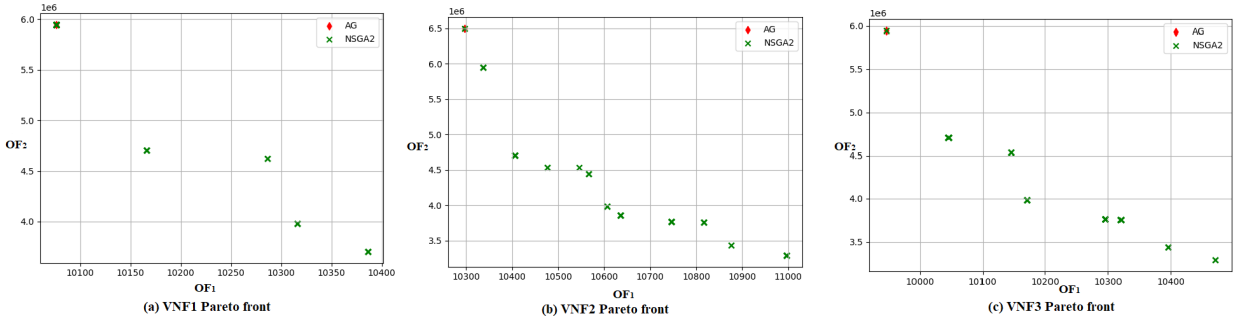

Figure 4: Numerical results derived from Test 1.

6.2 Analysis of Test 2 experimental results.

For Test 2, 30 independent runs were performed for each scenario, generating 30 Pareto sets per algorithm. Solutions from the 60 sets were aggregated into a single super set, from which a unified Pareto front, designated the *Optimal-Pareto front*, was constructed. These optimal Pareto fronts are depicted in Figures 5 to 7 for NSFNet, EON, and USA networks, respectively. For the computation of metrics for each Pareto front, the generated "Optimal Pareto Front" generated was employed as a reference. Subsequently, the average of these metrics was calculated, and is presented in Tables (2, 3, 4) respectively. ANOVA tests performed yielded a p-value less than 5% for each index, thereby indicating the statistical validity of comparing the computed averages presented in the tables.

NSGA-II has achieved better results considering the Hiper-Volume (HV) metric when compared to SPEA2, indicating that the volume covered by the Pareto fronts obtained by NSGA-II is higher and thus can be considered better. Taking the remaining performance metrics, lower values indicate superior Pareto front quality. Within this context, it is observed that NSGA-II consistently outperformed SPEA2 across all scenarios when evaluated with these performance indices. Therefore, we can conclude that, for the addressed problem, within the considered scenarios, NSGA-II represents a more promising algorithmic approach than SPEA2 for solving MSVNFP.

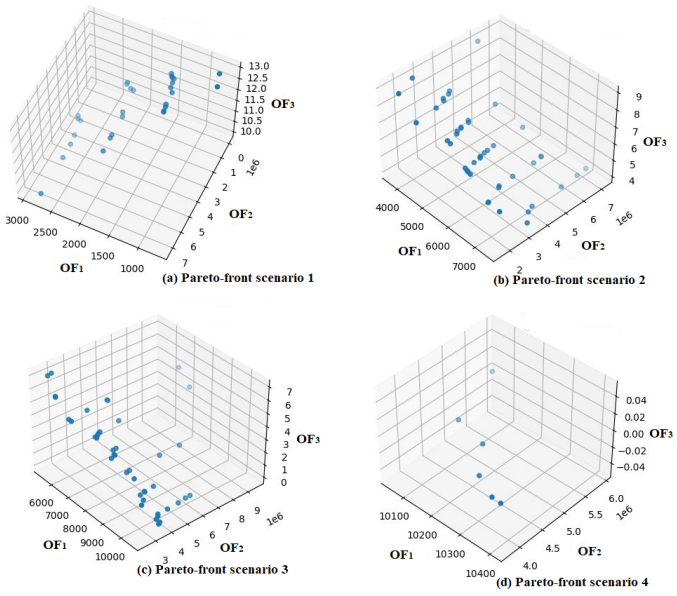

Figure 5: Optimal Pareto fronts for Test 2 - NSFNet topology.

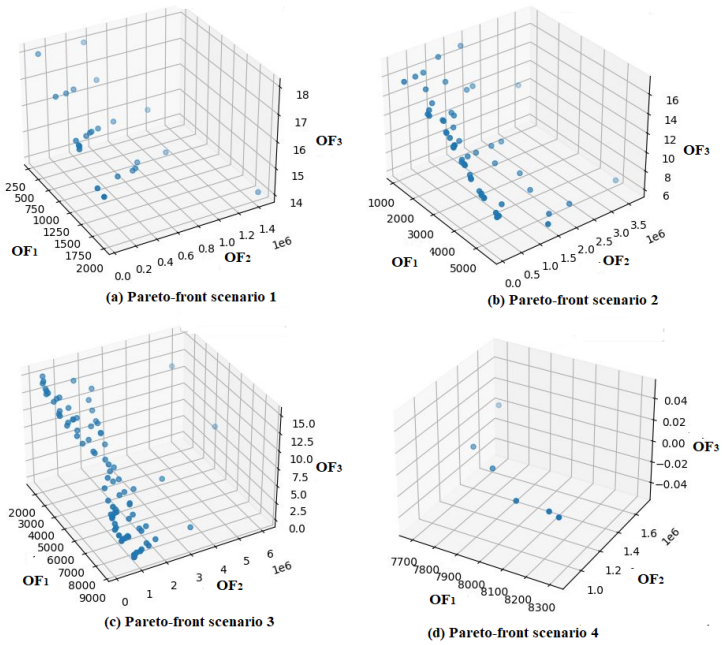

Figure 6: Optimal Pareto fronts for Test 2 - EON topology.

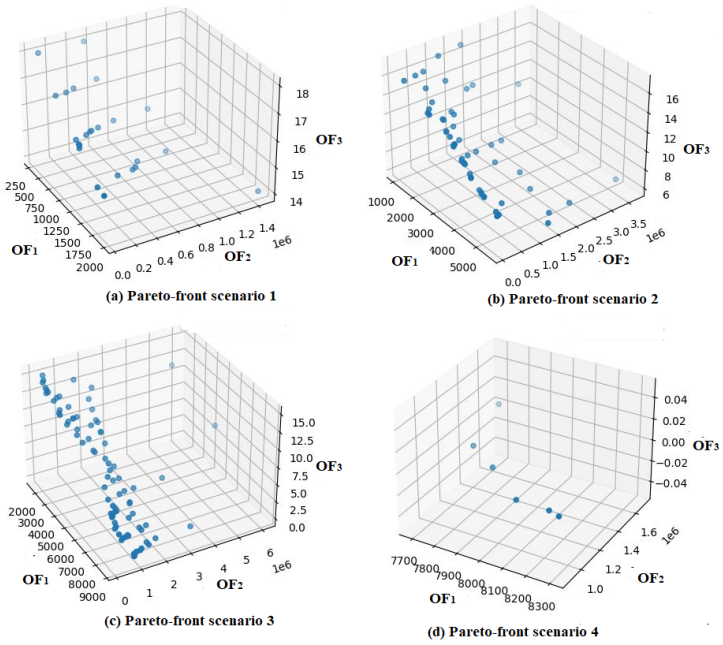

Figure 7: Optimal Pareto fronts for Test 2 - USA topology.

7 Concluding Remarks and Future Works

This manuscript has investigated MSVNFP within a multi-objective context. Its contributions include the tri-objective functions formulation of the problem and the development of MOEA-based solution methodologies, employing competitive, state-of-the-art algorithms such as NSGA-II and SPEA2.

The proposed algorithms effectively minimize blocking states, total cost, and maximum delay variance across destination nodes within multicast trees.

Numerical simulations demonstrate that obtained solutions for the MOP-MSVNFP problem constitute a non-dominated set located on the frontier of optimal solutions (Pareto front), thereby indicating a trade-off relationship among the studied objective functions.

Another key finding is that NSGA-II exhibits promising performance in addressing the MOP-MSVNFP problem, thus establishing its competitiveness as an algorithm for solving this class of problems.

As future research topics, the authors propose extending this study by considering: (a) additional relevant objective functions related to cost, quality of service, and robustness, (b) network conditions with a more realistic data layer parameters, (c) dynamic traffic, and (d) failure survivability. Furthermore, we will evaluate alternative state-of-the-art approaches, such as metaheuristic algorithms for static or dynamic multicast and reinforcement learning for dynamic multicast.

References

[1] S. E. Deering, "Host extensions for IP multicasting," Tech. Rep., 1988.

[2] A. Iyer, P. Kumar, and V. Mann, "Avalanche: Data center multicast using software defined networking," in *2014 sixth international conference on communication systems and networks (COMSNETS)*. IEEE, 2014, pp. 1–8.

[3] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 27–51, 2014.

[4] F. Callegati, W. Cerroni, C. Contoli, and G. Santandrea, "Sdn controller design for dynamic chaining of virtual network functions," in *2015 Fourth European Workshop on Software Defined Networks*, 2015, pp. 25–30.

[5] B. Yi, X. Wang, K. Li, M. Huang *et al.*, "A comprehensive survey of network function virtualization," *Computer Networks*, vol. 133, pp. 212–262, 2018.

[6] S. Q. Zhang, Q. Zhang, H. Bannazadeh, and A. Leon-Garcia, "Routing algorithms for network function virtualization enabled multicast topology on SDN," *IEEE Transactions on Network and Service Management*, vol. 12, no. 4, pp. 580–594, 2015.

[7] G. García and D. P. Pinto-Roa, "Multicast routing and virtual network function placement in NFV-SDN networks: a genetic algorithms approach," in *Proceedings of the 2022 Latin America Networking Conference*, ser. LANC '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 10–17. [Online]. Available: https://doi.org/10.1145/3545250.3560846

[8] C. A. C. Coello, *Evolutionary algorithms for solving multi-objective problems*. Springer, 2007.

[9] S. Vrontis, S. Xynogalas, and E. Sykas, "Steiner tree compilation of multicast under differentiated services constraints," *Journal of Communications and Networks*, vol. 9, no. 1, pp. 84–92, 2007.

[10] K. Mahajan, D. Sharma, and V. Mann, "Athena: Reliable multicast for group communication in SDN-based data centers," in *2017 9th International Conference on Communication Systems and Networks (COMSNETS)*. IEEE, 2017, pp. 174–181.

[11] L. Huang, X. Zhi, Q. Gao, S. Kausar, and S. Zheng, "Design and implementation of multicast routing system over SDN and sFlow," in *2016 8th IEEE International Conference on Communication Software and Networks (ICCSN)*. IEEE, 2016, pp. 524–529.

[12] L.-H. Huang, H.-C. Hsu, S.-H. Shen, D.-N. Yang, and W.-T. Chen, "Multicast traffic engineering for software-defined networks," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[13] H.-S. Kim, S. Yun, H. Kim, H. Shin, W.-T. Kim *et al.*, "An efficient SDN multicast architecture for dynamic industrial IoT environments," *Mobile Information Systems*, vol. 2018, 2018.

[14] Y.-D. Lin, Y.-C. Lai, H.-Y. Teng, C.-C. Liao, and Y.-C. Kao, "Scalable multicasting with multiple shared trees in software defined networking," *Journal of Network and Computer Applications*, vol. 78, pp. 125–133, 2017.

[15] J. Tao, Y. Shen, Y. Yan, Y. Wu, Y. Zhang, and J. Wan, "A distributed heuristic multicast algorithm based on QoS implemented by SDN," in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*. IEEE, 2017, pp. 23–29.

[16] R. Mohammadi, R. Javidan, M. Keshtgari, and R. Akbari, "A novel multicast traffic engineering technique in SDN using TLBO algorithm," *Telecommunication Systems*, vol. 68, pp. 583–592, 2018.

[17] Y. Xiao, K. Thulasiraman, X. Fang, D. Yang, and G. Xue, "Computing a most probable delay constrained path: NP-hardness and approximation schemes," *IEEE Transactions on Computers*, vol. 61, no. 5, pp. 738–744, 2011.

[18] A. M. Allakany and K. Okamura, "Efficient Multicasting Algorithm Using SDN," *IJCSNS International Journal of Computer Science and Network Security*, vol. 17, pp. 292–297, 2017.

[19] S.-H. Shen, L.-H. Huang, D.-N. Yang, and W.-T. Chen, "Reliable multicast routing for software-defined networks," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 181–189.

[20] H. Soni, W. Dabbous, T. Turletti, and H. Asaeda, "NFV-based scalable guaranteed-bandwidth multicast service for software defined ISP networks," *IEEE Transactions on Network and Service Management*, vol. 14, no. 4, pp. 1157–1170, 2017.

[21] B. Yi, X. Wang, M. Huang, and L. Ma, "SDN and NFV enabled service function multicast mechanisms over hybrid infrastructure," *Peer-to-Peer Networking and Applications*, vol. 12, pp. 405–421, 2019.

[22] X. Wang, H. Xing, and H. Yang, "On multicast-oriented virtual network function placement: a modified genetic algorithm," in *Signal and Information Processing, Networking and Computers: Proceedings of the 5th International Conference on Signal and Information Processing, Networking and Computers (ICSINC)*. Springer, 2019, pp. 420–428.

[23] D. Beasley, D. R. Bull, and R. R. Martin, "An overview of genetic algorithms: Part 1, fundamentals," *University computing*, vol. 15, no. 2, pp. 56–69, 1993.

[24] S. Q. Zhang, A. Tizghadam, B. Park, H. Bannazadeh, and A. Leon-Garcia, "Joint NFV placement and routing for multicast service on SDN," in *NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2016, pp. 333–341.

[25] O. Alhussein, "On the Orchestration and Provisioning of NFV-enabled Multicast Services," 2020.

[26] S. Q. Zhang, Q. Zhang, H. Bannazadeh, and A. Leon-Garcia, "Network function virtualization enabled multicast routing on SDN," in *2015 IEEE International Conference on Communications (ICC)*. IEEE, 2015, pp. 5595–5601.

[27] Z. Xu, W. Liang, M. Huang, M. Jia, S. Guo, and A. Galis, "Approximation and online algorithms for NFV-enabled multicasting in sdns," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 625–634.

[28] B. Yi, X. Wang, M. Huang, and A. Dong, "A multi-stage solution for NFV-enabled multicast over the hybrid infrastructure," *IEEE Communications Letters*, vol. 21, no. 9, pp. 2061–2064, 2017.

[29] Y. Cheng and L. Yang, "VNF deployment and routing for NFV-enabled multicast: A steiner tree-based approach," in *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2017, pp. 1–4.

[30] C. Ren, X. Chen, H. Xiang, Y. Wang, Y. Li, and H. Li, "On efficient delay-aware multisource multicasting in nfv-enabled softwarized networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 3371–3386, 2022.

[31] M. Asgarian, G. Mirjalily, and Z.-Q. Luo, "Trade-off between efficiency and complexity in multi-stage embedding of multicast vnf service chains," *IEEE Communications Letters*, vol. 26, no. 2, pp. 429–433, 2022.

[32] M. T. Emmerich and A. H. Deutz, "A tutorial on multiobjective optimization: fundamentals and evolutionary methods," *Natural computing*, vol. 17, pp. 585–609, 2018.

[33] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm and evolutionary computation*, vol. 1, no. 1, pp. 32–49, 2011.

[34] L. G. Moré, M. A. Brizuela, H. L. Ayala, D. P. Pinto-Roa, and J. L. V. Noguera, "Parameter tuning of CLAHE based on multi-objective optimization to achieve different contrast levels in medical images," in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 4644–4648.

[35] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[36] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," *TIK report*, vol. 103, 2001.

[37] J. Blank and K. Deb, "pymoo: Multi-objective Optimization in Python," *IEEE Access*, vol. 8, pp. 89 497–89 509, 2020.

[38] F. Lezama, A. F. Martínez-Herrera, G. Castañón, C. Del-Valle-Soto, A. M. Sarmiento, and E. M. de Cote, "Solving routing and spectrum allocation problems in flexgrid optical networks using precomputing strategies," *Photonic Network Communications*, vol. 41, pp. 17–35, 2021.

[39] D. A. Van Veldhuizen, *Multiobjective evolutionary algorithms: classifications, analyses, and new innovations*. Air Force Institute of Technology, 1999.

[40] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima, "Modified distance calculation in generational distance and inverted generational distance," in *Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015, Guimarães, Portugal, March 29–April 1, 2015. Proceedings, Part II 8*. Springer, 2015, pp. 110–125.

[41] C. A. Coello Coello and M. Reyes Sierra, "A study of the parallelization of a coevolutionary multiobjective evolutionary algorithm," in *MICAI 2004: Advances in Artificial Intelligence: Third Mexican International Conference on Artificial Intelligence, Mexico City, Mexico, April 26-30, 2004. Proceedings 3*. Springer, 2004, pp. 688–697.

[42] C. Fonseca, L. Paquete, and M. Lopez-Ibanez, "An Improved Dimension-Sweep Algorithm for the Hypervolume Indicator," in *2006 IEEE International Conference on Evolutionary Computation*, 2006, pp. 1157–1163.