Computer Networks 272 (2025) 111666

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Multi-objective optimization algorithm for VNF migration with priority awareness in dynamic networks

Yu Luo [a,b], Zhaogang Shu [a,b,*], Shuwu Chen [a,b], Qiang Tu [a,b], Xianzhang Wu [a,b], Qingjie Lin [a,b]

[a] The computer and information college, Fujian Agriculture and Forestry University, Fuzhou, 350002, China
[b] Engineering Research Center of Smart Sensing and Agricultural Chip Technology, Fujian Province University, Fuzhou, 350002, China

ARTICLE INFO

Keywords:
Network Function Virtualization
VNF migration
Priority awareness
Multi-objective optimization

ABSTRACT

With the continuous development of Network Function Virtualization (NFV) technology, Virtual Network Function (VNF) migration has become a crucial approach to optimizing network resource utilization, reducing service latency, and improving service quality. However, in dynamic network environments, VNF migration faces challenges such as resource overload, service request prioritization, migration cost optimization, routing overhead, and energy consumption. To address these challenges, this paper proposes a priority-aware and multi-objective optimization-based VNF migration algorithm, namely the Lagrangian Fish Optimization for VNF Migration (LFO-VNM) Algorithm. This algorithm integrates the Lagrangian relaxation method with the Artificial Fish Swarm Algorithm (AFSA) to dynamically adjust resource allocation and migration paths, optimizing migration cost, network performance, and node energy consumption while prioritizing high-priority service requests. First, a Mixed-Integer Linear Programming (MILP) model is established to quantify the impact of VNF migration on network link load, node resource consumption, and service performance. Based on this, a multi-objective optimization model is formulated, considering network bandwidth, latency, migration cost, and energy consumption. This model is decomposed into a series of linear subproblems, which are more efficiently solved using the Lagrangian relaxation method. Finally, leveraging the global search capability of AFSA, an efficient solution algorithm, LFO-VNM, is designed to optimize VNF migration decisions. Experimental results demonstrate that the proposed algorithm not only improves computational efficiency but also effectively reduces total cost and energy consumption, outperforming existing migration algorithms across various network topologies. This study provides an effective solution for VNF migration and resource scheduling in complex network environments.

1. Introduction

With the rise of Network Function Virtualization (NFV) [1] technology, the traditional network architecture's reliance on dedicated hardware gradually reveals its limitations, making it difficult to meet the increasingly complex and dynamic service demands. NFV enhances network flexibility and scalability by decoupling network functions from dedicated hardware and running them on general-purpose computing devices. It also enables efficient resource utilization and flexible deployment through virtualization. As one of the core technologies of NFV, Virtual Network Functions (VNFs) run in the form of virtual machines or containers, and multiple VNF instances are chained together through Service Function Chains (SFC) [2] to provide end-to-end network services. SFC, by combining the advantages of NFV and Software-Defined Networking (SDN) [3,4], not only enables dynamic management of service chains but also optimizes the mapping of physical network resources. As shown in Fig. 1, Network Function Virtualization transforms physical devices such as load balancers, firewalls, and network address translators into virtual functions through the virtualization layer, establishing the relationship between logical and physical chains.

However, in dynamic network environments, the random fluctuations of user requests and the uneven distribution of physical resources often lead to node or link resource overload. This not only increases service latency and may even cause service interruptions, but also significantly reduces the overall resource utilization efficiency of the network, thereby affecting the user experience. To mitigate these issues, VNF migration technology has emerged [5]. By migrating VNFs from overloaded nodes to low-load nodes, VNF migration can effectively balance resource allocation and improve network performance. However, the inevitable path adjustments, routing overhead, and increased energy consumption during the migration process further complicate

* Corresponding author.
E-mail address: zgshu@fafu.edu.cn (Z. Shu).

https://doi.org/10.1016/j.comnet.2025.111666
Received 2 March 2025; Received in revised form 26 June 2025; Accepted 27 August 2025
Available online 3 September 2025

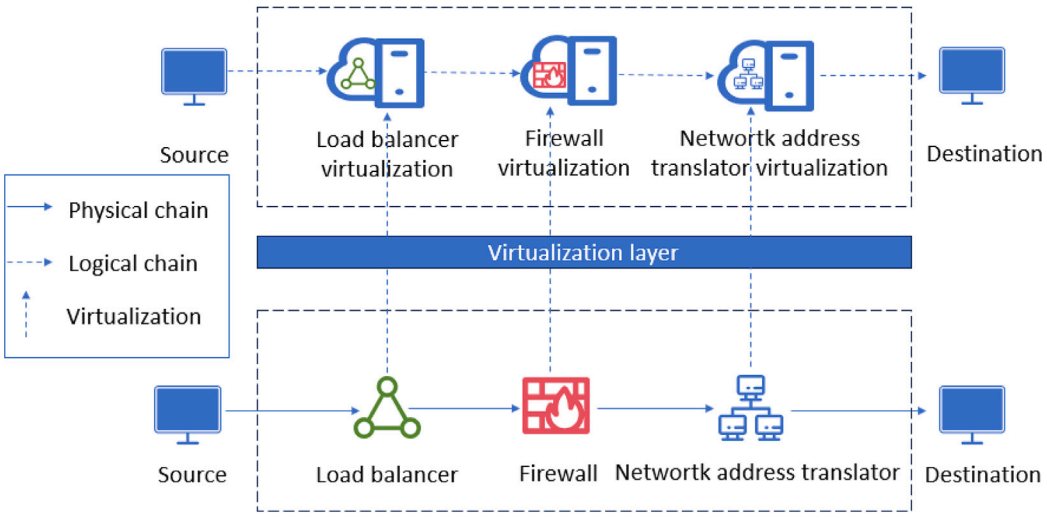


Fig. 1. Network Function Virtualization diagram.

resource management, while also impacting the cost efficiency and service quality assurance of operators.

In recent years, researchers have conducted in-depth explorations into VNF migration strategies [6,7], proposing various methods to optimize network performance and resource allocation [8], providing new solutions for service quality assurance in dynamic network environments [9–11]. Although existing studies [12–14] have achieved certain results in optimizing Quality of Service (QoS) metrics, most methods fail to fully consider environmental changes, making it difficult to adapt to the large-scale VNF migration needs in complex dynamic environments. Additionally, there are notable deficiencies in the adaptability of existing research regarding multi-objective optimization and global resource allocation. For instance, most methods assume that a VNF instance (VNFI) is used by a single Service Function Chain (SFC), neglecting the complexity of migration strategies in scenarios where VNFI is shared among multiple SFCs. Under this assumption, the migration design is relatively simple, only considering the impact of a single SFC. However, in practical networks, VNFI sharing is common, and its migration often generates a cascading impact on all associated SFCs, significantly increasing the complexity of migration strategy design. Furthermore, in dynamic networks, multiple nodes or links may become overloaded simultaneously, requiring the concurrent migration of multiple VNFI instances, yet research on this concurrent migration problem remains relatively scarce. More importantly, existing methods predominantly focus on optimizing specific objectives [15–17] and give insufficient consideration to trade-offs between multiple objectives. For instance, [12] employs a graph neural network coupled with deep reinforcement learning to adaptively capture traffic dynamics, yet its design is confined to a single trade-off between migration latency and computational efficiency; it neither integrates migration cost, energy consumption, or end-to-end performance into its optimization objective, nor can it satisfy real-time requirements due to protracted training and inference times in large-scale topologies. Similarly, [13] proposes a heuristic framework for SFC routing and VNFI migration that incorporates energy expenditure and reconfiguration overhead but depends on static load thresholds and a Markov decision process, rendering it incapable of dynamically responding to sudden, elastic traffic surges in deployments spanning thousands of nodes. Finally, [16] models SFC migration via open Jackson networks and introduces the PHS and AUB heuristics to minimize average delay; although effective in latency reduction, this approach omits migration overhead and energy metrics, overlooks fairness in resource allocation and priority-aware guarantees, and lacks any assessment of convergence properties or scheduling overhead under large-scale, heterogeneous conditions.. At the same time, these methods have limitations in handling service priorities, which may result in inadequate service quality for high-priority requests, thereby affecting the overall performance of the service chain. Therefore, how to effectively address resource overload, enhance multi-objective optimization capabilities, and prioritize high-priority service requests in dynamic environments has become a core issue that requires urgent resolution in current research.

To address these challenges, this study proposes a VNF migration strategy that combines priority awareness and multi-objective optimization. The strategy dynamically adjusts resource allocation and migration paths, optimizing not only migration costs and network performance but also ensuring the service quality of high-priority tasks, thereby enhancing the service efficiency and stability in dynamic network environments. The innovation of this study lies in improving the performance of existing methods in joint multi-objective optimization and service priority assurance, offering a novel approach to resource scheduling in complex dynamic network environments. The contributions of this paper are as follows:

1. We designed a multi-dimensional priority scheduling model that incorporates service chain length as an additional dimension beyond bandwidth and end-to-end latency, forming a tightly coupled priority score. SFC requests are then scheduled in order based on this score, improving the utilization of critical links and computing resources as well as the QoS of high-priority services.
2. We constructed a mixed integer linear programming model of migration cost, routing overhead, and energy consumption. We used Lagrange relaxation to embed resource and QoS constraints into the objective function, and combined the artificial fish school algorithm to balance all metrics, achieving lower migration overhead and better performance.
3. We propose a multi-dimensional node scoring mechanism based on remaining computing power, remaining storage, and network connectivity. Node priorities are dynamically updated to provide quantitative decision-making for VNF deployment and migration, improving convergence speed and load balancing effectiveness.

The rest of the paper is organized as follows: Section 2 discusses related work on VNF migration. Section 3 presents the problem formulation. Section 4 introduces our proposed algorithm. Section 5 analyzes the numerical results, and finally, Section 6 concludes the paper.

2. Related work

In this section, we introduce representative works related to VNF migration.

In recent years, significant progress has been made in research on Virtual Network Function (VNF) migration and Service Function Chain (SFC) reconfiguration. [18] proposes the partial migration model for the first time, designed for scenarios where a single VNF instance is shared across multiple SFCs. Their approach selectively migrates only those VNFs whose relocation yields the greatest reduction in end-to-end latency, thereby minimizing both overall link delay and migration overhead. Additionally, it integrates real-time load monitoring functionality to dynamically adjust the migration subset based on fluctuations in resource usage. Although this method reduces unnecessary migration overhead, it focuses solely on local optimization during the migration process, without addressing global resource management. Some studies focus on VNF cluster migration and embedding optimization. [19] investigates the VNF cluster migration problem, where VNF instances within the same cluster are often reused by multiple service chains and proposed an Integer Linear Programming (ILP) model and designing an efficient migration algorithm to reduce embedding costs.directly supporting the efficient migration of multiple SFCs sharing a single VNF instance. However, this method is primarily suited to static topologies and has limited adaptability in dynamic environments. [20] proposes a priority-aware VNF migration method based on deep reinforcement learning (PAVM), which optimizes VNF migration strategies by learning the network state. Although this method enhances the intelligence of migration, reinforcement learning approaches have slower convergence rates and higher computational costs, and they do not optimize global resource management for SFC requests. [21] investigates the SFC migration problem in 5G edge networks and proposes a mobility-aware migration strategy, focusing on optimizing the impact of user mobility on SFCs. This method demonstrates strong adaptability in edge computing environments, but its applicability to data center environments is limited, and it does not consider migration cost optimization. Some studies focus on multi-criteria optimization to comprehensively improve VNF migration performance. [22] proposes a multi-criteria decision-making method aimed at minimizing the impact of VNF migration on multiple SFCs. This method considers VNF sharing and reduces the damage to Quality of Service (QoS) by optimizing migration decisions. However, it does not fully account for dynamic changes in network resources and does not involve migration cost optimization, making it difficult to adapt to large-scale dynamic network environments. [23] addresses dynamic SFC scheduling in geographically distributed cloud environments and proposes a cost-effective flow migration framework (CFM) that enhances SFC efficiency by optimizing migration costs and network performance after migration. Although CFM performs well in load balancing, it primarily focuses on flow migration rather than VNF migration decisions and does not consider the impact of SFC request priorities on resource allocation. With the development of Digital Twin (DT) technology, some studies begin to explore methods that combine resource demand prediction with intelligent migration. [24] combines Deep Reinforcement Learning (DRL) and Federated Learning (FL) to propose a resource demand prediction and VNF migration optimization method, mainly applied in DT networks. This method optimizes migration strategies by predicting resource demands, but its computational complexity is high, training costs are large, and it focuses more on resource prediction than directly optimizing VNF migration strategies. Additionally, some studies propose joint optimization methods for data center environments. [25] investigates the VNF migration problem in data center environments and proposes a joint resource optimization and latency-aware VNF migration method. This method uses an improved Genetic Algorithm (IHGE) to optimize VNF migration, assuming that each VNFI can be concurrently invoked by multiple SFCs, the cumulative link delay caused by shared instance migration and the cost savings from instance reuse are simultaneously incorporated into the objective function and constraints to optimize migration decisions. To support dynamic resource allocation, the algorithm periodically re-evaluates node load and re-initiates the IHGE search to adapt to changes in traffic patterns. However, this method does not incorporate service request priorities into the model, which may result in insufficient resource allocation for critical services and consequently impact their service quality.[26] introduces a multi-objective linear programming (MOLP) model to optimize dynamic SFC deployment in 5G networks, minimizing user service latency and VNF migration costs. The Trade-LCM framework proposed in the paper achieves better SFC mapping under network load imbalances, improving QoS. However, this method primarily focuses on the trade-off between migration costs and latency and does not fully consider global network resource optimization, with limited adaptability to dynamic environments. [27] proposes a Migration Index to measure the trend of node load changes and designs a fast and efficient heuristic migration algorithm based on this. Experimental results show that this method outperforms traditional deep learning methods in traffic prediction accuracy and can effectively reduce migration costs by approximately 20% . However, the limitation of this method lies in its over-reliance on traffic prediction accuracy, as large prediction errors may lead to unstable migration decisions. Additionally, this method primarily optimizes a single migration cost and does not address the priority scheduling issue of SFC requests.

Current research has made significant progress in optimizing migration paths, reducing migration costs and latency, and improving migration efficiency. However, model complexity, computational overhead, and real-time performance still limit its application in large-scale dynamic network environments. Existing methods struggle to strike a balance between computational efficiency and optimization quality, particularly in dynamic resource allocation, load balancing, and service priority assurance. To address this, this paper proposes a VNF migration algorithm based on multi-objective optimization and intelligent scheduling, integrating priority-awareness mechanisms, Lagrangian relaxation, and the Artificial Fish Swarm Algorithm (AFSA). The proposed method systematically optimizes migration costs, latency, energy consumption, and global resource utilization. It balances dynamic adaptability with computational efficiency, enhancing the stability of VNF migration while improving resource scheduling flexibility and service quality. To better illustrate the contributions of this paper, a detailed comparison of related works is provided in Table 1.

3. System model

In this section, we describe the system model from the following four aspects. For reference, the symbols used in this paper are summarized in Table 2.

3.1. Network model

The physical network is modeled as an undirected connected graph, denoted as $G = (N, L)$, where N represents the set of physical nodes, and L represents the set of physical links. The resources of each node n_i are defined as a triplet $n_i = \{C_i, M_i, B_i\}$, representing CPU resources, memory resources, and bandwidth resources, respectively. C_i^{total} denotes the total CPU resource capacity of node n_i, and M_i^{total} denotes the total memory resource capacity of node n_i. Each link $e_{(i,j)} \in E$ has properties $B_{(i,j)}^{total}$ and $L_{(i,j)}$, where $B_{(i,j)}^{total}$ represents the maximum link bandwidth capacity between node n_i and node n_j, and $L_{(i,j)}$ represents the transmission delay of link $e_{(i,j)}$. Additionally, the CPU resources of each node limit the number of shared VNF (Virtual Network Function) instances it can support. It is important to note that physical network nodes have different VNF support capabilities. Each node n_i can only support a specific set of VNFs, $V_i \in V$, where V is the set of all VNFs in the system. This limitation requires that the support capabilities of the nodes be fully considered when deploying and migrating VNFs.This paper assumes that all pending SFC requests have been initially deployed by the NFV orchestrator before the service arrives, each VNF in the service chain has been assigned to the corresponding physical node and an end-to-end link has been established. Overload detection

Table 1
Comparison of related works.

Literatures	VNF sharing	Dynamic resource allocation	Service request priority	Migration cost	Post-migration routing cost	Node energy consumption cost
[18]-2024	√	√	×	√	×	√
[19]-2024	√	×	√	√	×	×
[20]-2022	×	√	√	×	√	×
[21]-2024	√	√	×	√	×	×
[22]-2019	√	√	√	×	√	√
[23]-2024	×	√	×	√	×	√
[24]-2023	×	√	√	√	×	×
[25]-2021	√	√	×	×	×	×
[26]-2023	×	√	√	×	√	√
[27]-2021	×	×	√	√	×	√
Ours	√	√	√	√	√	√

• In this table, "√" means that this work takes this element into account, while "×" means that it does not.

Table 2
Summary of notations.

Notation	Definition
$G = (N, L)$	Physical network graph, where N represents the set of nodes, and L represents the set of physical links.
$C_i^{total}, M_i^{total}, B_{(i,j)}^{total}$	Total CPU, memory, and bandwidth resource capacities of node n_i.
$e_{(i,j)}$	Physical link connecting nodes n_i and n_j.
$B_{(i,j)}^{total}$	Maximum link bandwidth capacity of link $e_{(i,j)}$.
$D_{(i,j)}$	Transmission distance of link $e_{(i,j)}$.
V_i	The set of VNFs supported by node n_i.
V	The set of all VNFs in the system.
r_u	The u-th SFC request.
λ_u	Traffic demand of the SFC request r_u.
D_{max}^u	The maximum end-to-end delay limit for SFC request r_u.
R_k	The resources required to deploy VNF f_k^u.
R_n^{total}	The maximum available resource limit of node n.
τ_u	The processing time slot for SFC request r_u.
$\tilde{T}_{mig}$	The time required to migrate VNF f_k^u.
γ	The fixed migration overhead for migrating VNF f_k^u.
$L_{(i,j)}$	Transmission delay of link $e_{(i,j)}$.
$\omega_1, \omega_2, \omega_3$	Weight coefficients for bandwidth utilization, transmission distance, and link delay.
P_j^{min}	The basic energy consumption of node n_j.
P_j^{max}	The maximum energy consumption of node n_j.
$U_n(t)$	Resource utilization rate of node n at time slot t.
β_u	VNF migration time factor, used to measure the impact of migration time on the total cost.
$\delta_n(t)$	Binary variable indicating whether node n has sufficient remaining resources at time slot t.
$V_n^{(f_k^u)}$	Binary variable indicating whether node n supports VNF f_k^u.
$X_{ukn}(t)$	Binary variable indicating whether VNF f_k^u is deployed on node n at time slot t.
$Y_{uij}(t)$	Binary variable indicating whether SFC r_u uses link $e_{(i,j)}$ at time slot t.
$Z_{ukn}(t)$	Binary variable indicating whether VNF f_k^u is migrated to node n at time slot t.

and migration in subsequent stages are based on this deployment, and the initial mapping process will not be described in detail here.

3.2. SFC request model

In the model, the Service Function Chain (SFC) request is described as a sequence of Virtual Network Functions (VNFs) that must be executed in a specific order to provide end-to-end network services. Each SFC request is represented as a sextuple: $r_u = \{s(r_u), d(r_u), F_u, \lambda_u, D_{max}^u, \tau_u\}$, where $s(r_u)$ and $d(r_u)$ denote the source and destination nodes of the SFC request, respectively. $F_u = \{f_1^u, f_2^u, \ldots, f_{I_u}^u\}$ is the sequence of VNFs that must be executed in order to fulfill the network function of the service request. The traffic demand of the SFC request is represented by λ_u, which determines the required network bandwidth for packet transmission through the service chain. D_{max}^u represents the maximum end-to-end transmission delay limit for the SFC request, ensuring that data is transmitted within an acceptable time frame. The physical resources consumed by each VNF f_k^u during execution are primarily reflected in CPU and memory consumption. Additionally, the traffic demand λ_u of the SFC request directly affects its transmission bandwidth requirement. τ_u represents the time slot in which the request occurs. Within the time slot τ_u, the deployment of all VNFs and the allocation of transmission paths for the SFC request must be completed to ensure service quality.

3.3. Priority-aware model

3.3.1. Priority-awareness of network nodes

To ensure the rational selection of target nodes during VNF migration, a priority-aware mechanism based on node resource utilization, load balancing, and network connectivity is introduced. For any node n_j, the priority calculation formula is as follows:

$$R_j = \alpha \cdot \frac{C_j^{remain}}{C_j^{total}} + \beta \cdot \frac{M_j^{remain}}{M_j^{total}} + \gamma \cdot \frac{B_j^{remain}}{B_j^{total}} \tag{1}$$

$$\bar{C} = \frac{\sum_{n \in N} \frac{C_n^{used}}{C_n^{total}}}{|N|} \tag{2}$$

$$\bar{M} = \frac{\sum_{n \in N} \frac{M_n^{used}}{M_n^{total}}}{|N|} \tag{3}$$

$$L_j = 1 - \left| \frac{C_j^{used}}{C_j^{total}} - \bar{C} \right| - \left| \frac{M_j^{used}}{M_j^{total}} - \bar{M} \right| \tag{4}$$

$$C_j = \frac{1}{\text{avg\_hop}(n_j)} \tag{5}$$

$$P_j = w_1 \cdot R_j + w_2 \cdot L_j + w_3 \cdot C_j \tag{6}$$

In this model, R_j represents the resource availability of node n_j, which indicates the current availability of resources at the node. A higher value of R_j means that the node has more available resources. C_j^{remain} denotes the current available CPU resources of node n_j, and C_j^{total} denotes the total CPU resources of node n_j. M_j^{remain} represents the current available memory resources of node n_j, and M_j^{total} denotes the total memory resources of node n_j. B_j^{remain} represents the current available bandwidth resources of node n_j, while B_j^{total} denotes the total bandwidth resources of node n_j.

α, β, and γ are the weight coefficients used to balance CPU, memory, and bandwidth in the resource availability calculation. This formula calculates the ratio of remaining resources at node n_j, using a weighted approach to measure the remaining resources of CPU, memory, and bandwidth, which reflects the degree of resource availability at the node. L_j represents the load balancing of node n_j, which indicates how close the node's resource utilization is to the network's average load. A higher value of L_j means that the node's load is more balanced. C_j^{used} denotes the CPU resources used by node n_j, and $\bar{C}$ represents the average CPU utilization across all nodes in the network. M_j^{used} denotes the memory resources used by node n_j, and $\bar{M}$ represents the average memory utilization across all nodes in the network. This formula calculates the load balancing of node n_j by measuring the deviation between the CPU and memory utilization of the node and the average utilization across the network. The smaller the deviation, the closer L_j is to 1, which indicates that the node's load is more balanced.

C_j represents the network connectivity of node n_j, calculated by the average network hop count of the node. P_j represents the priority of node n_j, which is used to measure the importance of the node in the VNF migration process. A higher value of P_j means that the node is more suitable as the target node for VNF migration. w_1, w_2, and w_3 are the weight coefficients for resource availability, load balancing, and connectivity in the priority calculation. The importance of these factors may vary depending on the network environment. In this paper, equal weights are used to enhance the algorithm's adaptability, ensuring that it maintains optimal performance in different application scenarios without requiring complex parameter tuning for specific environments. This formula calculates the comprehensive priority score of each node by weighting resource availability, load balancing, and connectivity. Nodes with higher priority are given preference during the VNF migration process to ensure that the target node has sufficient resources, balanced load, and good connectivity. This priority calculation method helps in selecting target nodes with lighter loads and sufficient resources during VNF migration, achieving better migration performance under network resource constraints.

3.3.2. Priority-awareness of SFC

To handle SFC requests more reasonably during VNF migration and resource allocation, this paper introduces a Service Function Chain (SFC) priority-awareness mechanism. This mechanism comprehensively considers the bandwidth requirements, end-to-end delay constraints, and service chain length of the SFC requests to calculate the priority of each SFC request, thereby ensuring fairness and efficiency in resource allocation. For any SFC request r_u, its priority is calculated as follows:

$$P_{total}^u = \omega_1 \cdot P_{bw}^u + \omega_2 \cdot P_{latency}^u + \omega_3 \cdot P_{length}^u \tag{7}$$

$$P_{bw}^u = \frac{B_{max} - \lambda_u}{B_{max}} \tag{8}$$

$$P_{latency}^u = \frac{D_{max}^u - D_{current}}{D_{max}^u} \tag{9}$$

$$P_{length}^u = \frac{1}{|F_u|} \tag{10}$$

Where P_{bw}^u represents the bandwidth priority of the SFC request r_u,$B_{\max}$ indicates the maximum bandwidth requirement among all pending SFC requests in the system. $P_{latency}^u$ represents the delay priority of the SFC request r_u, and P_{length}^u represents the service chain length priority of the SFC request. D_{max}^u denotes the maximum end-to-end delay for the SFC request, and $D_{current}$ denotes the current end-to-end delay of the network path. $|F_u|$ represents the number of VNFs required for the SFC request. ω_1, ω_2, and ω_3 are the weight coefficients for bandwidth, delay, and service chain length, respectively. Since it is not possible to determine in advance which factor is more critical than the others, we set equal weights to ensure the algorithm is applicable in a broader range of application scenarios and does not exhibit imbalance under specific conditions. The design of this priority mechanism aims to balance multi-dimensional resource demands and ensure reasonable allocation of network resources during VNF deployment and migration. The priority-aware mechanism proposed in this paper aims to achieve better resource allocation and management in VNF migration and SFC request processing by integrating resource availability, load balancing, and network connectivity. The priority-awareness of network nodes is determined through multi-dimensional calculations of the suitability of candidate nodes, allowing for the selection of optimal target nodes in VNF migration. The priority-awareness of service function chains calculates priorities based on bandwidth requirements, delay constraints, and service chain length, prioritizing resource allocation to critical SFC requests.

This dual-priority-awareness mechanism effectively enhances the utilization efficiency of network resources, reduces network congestion, and ensures service quality. By considering multiple key factors within a multi-objective optimization framework for resource allocation, this model provides a scientifically sound and rational VNF migration and SFC request allocation scheme for complex NFV environments, ultimately achieving the optimization of network performance and resource utilization.

3.4. VNF migration model

To illustrate the partial VNF migration model, we first describe the utilization relationships between SFCs and VNFs, the associations between SFCs and physical links, and the mapping relationships between VNFs and nodes. This section defines the key binary and continuous variables in the VNF migration model.

$$X_{ukn}(t) = \begin{cases} 1, & \text{if VNF } f_k^u \text{ is deployed on node } n \text{ at time slot } t \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

$$Y_{uij}(t) = \begin{cases} 1, & \text{if SFC request } r_u \text{ uses link } e_{ij} \text{ at time slot } t \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

$$Z_{ukn}(t) = \begin{cases} 1, & \text{if VNF } f_k^u \text{ is migrated to node } n \text{ at time slot } t \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

These variables are used to characterize the usage states of nodes and links during the migration process and form the mathematical foundation for the subsequent objective functions and constraints.

3.5. Problem formulation

This section introduces the mathematical formulation of the multi-objective SFC deployment problem we propose, including the constraints and objectives.

3.5.1. Constraints

(1) Each node can deploy multiple VNFs, but it must satisfy the resource limitations of the node and the VNF compatibility constraints.

$$\sum_{(u,k)} X_{ukn}(t) \cdot R_k \le R_n^{total} \tag{14}$$

(2) Not all nodes can deploy any type of VNF. Only VNFs that meet the node's capability conditions can be deployed on the respective

nodes.

$$X_{ukn}(t) \le V_n^{(f_k^u)} \tag{15}$$

(3) When VNF f_k^u is migrated to a new node, the node must have sufficient remaining resources and support the specific VNF type.

$$Z_{ukn}(t) \le V_n^{(f_k^u)} \cdot \delta_n(t) \tag{16}$$

(4) The VNFs must be executed in the order specified in the SFC request to ensure service integrity:

$$\sum_{n\in N} X_{u(k+1)n}(t) \le \sum_{n\in N} X_{ukn}(t) \tag{17}$$

This constraint ensures that f_k^u must be completed before f_{k+1}^u can be deployed on the physical node.It thereby enforces service chain integrity by preventing out-of-order deployment. This mechanism is distinct from the minimum post-migration runtime τ_u, which guarantees each migrated VNF remains stably online for the required duration.

(5) There must be a physical link between the migrated VNFs, forming a complete transmission path:

$$X_{ukn}(t) + X_{u(k+1)n}(t) \le \sum_{(i,j)\in E} Y_{uij}(t) \tag{18}$$

This constraint ensures that if f_k^u and f_{k+1}^u are deployed on different nodes, a valid network path must exist. Specifically, when these two VNFs reside on distinct physical nodes, the sum on the left-hand side equals 2, and the right-hand side requires at least one active link indicator $Y_{u,i,j}(t) = 1$ along the edges connecting their hosting nodes. Conversely, if they share the same node, the left-hand side is at most 1 and the constraint is trivially satisfied without additional path requirements. This formulation thus guarantees physical connectivity for each adjacent VNF pair in the service chain, preserving end-to-end service integrity.

(6) Node resources cannot exceed their physical capacity, and the migrated VNF must comply with the resource limits:

$$C_n^{used}(t) = \sum_{(u,k)} X_{ukn}(t) \cdot C_k^{cpu} \le C_n^{total} \tag{19}$$

$$M_n^{used}(t) = \sum_{(u,k)} X_{ukn}(t) \cdot M_k^{mem} \le M_n^{total} \tag{20}$$

$$B_n^{used}(t) = \sum_{(u,k)} X_{ukn}(t) \cdot B_k^{band} \le B_n^{total} \tag{21}$$

Here, C_k^{cpu} and M_k^{mem} denote the CPU and memory resource consumption of VNF instance v_k on its hosting node; B_k^{band} indicates its bandwidth demand on the node's network interfaces. These constraints ensure that, at any time, the aggregated resource usage on node n does not exceed its physical availability.

(7) The data transmitted over the link must not exceed the physical link's bandwidth:

$$B_{(i,j)}^{used}(t) = \sum_{u} Y_{uij}(t) \cdot \lambda_u \le B_{(i,j)}^{total} \tag{22}$$

(8) The end-to-end delay on the link must meet the maximum delay limit of the SFC:

$$\sum_{(i,j)\in P_{(i,j)}} D_{uij}(t) \le D_{max}^u \tag{23}$$

(9) When VNF migration occurs, resources must be reallocated at the target node:

$$Z_{ukn}(t) \ge X_{ukn}(t) - X_{ukn}(t-1) \tag{24}$$

(10) After the VNF migration is completed, the resource status of the target node must be updated to reflect the change in resource usage.

$$R_n^{used}(t+1) = R_n^{used}(t) + \sum_{(u,k)} Z_{ukn}(t) \cdot R_k \tag{25}$$

This constraint ensures that the resource allocation after migration is correctly updated to reflect the new resource consumption.

3.5.2. Objective

The total cost of VNF migration consists of three components: migration cost, post-migration routing cost, and node energy consumption cost.

Migration cost: The migration cost primarily measures the additional resource consumption and service disruption risk caused by the VNF migration process. In a network environment, when a VNF migrates from the source node to the target node, it consumes transmission resources and may trigger service disruptions. In this paper, we reasonably consider bandwidth consumption, migration data volume, and link transmission distance as the main factors affecting migration cost, which aligns with actual network resource consumption. We express it as follows:

$$C_{mig}(t) = \sum_{(u,k,n)} Z_{ukn}(t) \cdot \left(\lambda_u \cdot D_{(i,j)} + \beta_u \cdot T_{mig} + \gamma\right) \tag{26}$$

Post-migration routing cost: The routing cost measures the efficiency of network transmission during migration and its pressure on links, which is particularly useful for dynamic traffic management. In this paper, we fully consider the congestion level of the links and transmission distance as important factors influencing network performance, which aligns with congestion control mechanisms in actual network transmission. We express it as follows:

$$C_{route}(t) = \sum_{(i,j)\in E} Y_{(i,j)} \cdot \left(\omega_1 \cdot \frac{\lambda_u}{B_{(i,j)}} + \omega_2 \cdot D_{(i,j)} + \omega_3 \cdot L_{(i,j)}\right) \tag{27}$$

Node energy consumption cost: The energy consumption cost reflects the energy consumed by nodes in the network when processing SFC requests and performing migration. The energy consumption cost directly affects the network's sustainability and energy efficiency management. We effectively consider the node's base energy consumption and the energy variation dependent on the load, which aligns with the energy consumption characteristics in modern data centers and distributed networks. We express it as follows:

$$U_n(t) = \frac{C_n^{used}(t)}{C_n^{total}} + \frac{M_n^{used}(t)}{M_n^{total}} \tag{28}$$

$$C_{energy}(t) = \sum_{n\in N} \left[P_j^{min} + (P_j^{max} - P_j^{min}) \cdot U_n(t)\right] \tag{29}$$

Finally, we use the following formula to represent our total objective:

$$C_{total}(t) = C_{mig}(t) + C_{route}(t) + C_{energy}(t) \tag{30}$$

The core objective of this cost function is to minimize the sum of the three components, subject to the resource constraints and QoS requirements of network services. It comprehensively considers migration cost, routing cost, and node energy consumption cost, capturing the impact of VNF migration on network performance. It covers the multidimensional factors such as bandwidth, transmission distance, link congestion, and energy consumption. Supporting dynamic migration modeling across multiple time slots, it is applicable to complex dynamic NFV environments. In an NFV environment, this objective function can effectively guide the selection of VNF migration strategies, reducing the risk of service disruptions and network congestion caused by migration. It helps improve network resource utilization efficiency, reduce energy consumption, and enhance overall network performance and sustainability.

3.5.3. Complexity of the algorithm

In terms of computational complexity, the core of the LFO-VNM algorithm consists of two modules—AFSA swarm search and Lagrangian relaxation—that alternate in an iterative process. In each iteration, for a swarm of size M, each individual must undergo one evaluation of service chain construction, functional graph embedding, and on-chain scheduling migration, yielding a single-evaluation complexity of $O(|V| + |E| + \sum_{u\in\mathcal{U}} |\text{SFC}_u|)$, where $|V|$ and $|E|$ denote the number of

network nodes and links, respectively, and $\sum|\mathrm{SFC}|$ is the total length of service chains to be processed in the current time slot. If the total number of iterations is T, the overall time complexity becomes $O\left(T \cdot M \cdot \left(|V| + |E| + \sum|\mathrm{SFC}|\right)\right)$. Although this polynomial complexity increases significantly with network scale and request volume R, in practical deployments, by judiciously choosing the number of iterations and swarm size, and leveraging efficient implementation of each evaluation stage, LFO-VNM can complete a full dynamic scheduling and migration decision within minutes on a typical operator-scale network, thereby satisfying the real-time requirements of online deployment.

4. Algorithm design

The VNF migration problem is an NP-Hard [28] multi-objective optimization problem, involving multiple optimization objectives such as resource allocation, migration cost, energy consumption, and routing cost. To address this complex multi-constraint problem, this paper proposes a hybrid heuristic algorithm based on Lagrangian Relaxation (LR) [29]and the Artificial Fish Swarm Algorithm (AFSA) [30]. This method combines the ability of Lagrangian Relaxation to handle resource constraints with the global search capability of AFSA to improve migration efficiency and resource utilization in large-scale network environments.

4.1. Algorithm design overview

In this paper, the proposed Virtual Network Function (VNF) migration algorithm aims to effectively manage computational and transmission resources in dynamic network environments through a multi-stage [31,32] optimization strategy, minimizing migration costs. To describe the migration process more systematically, VNF migration can be divided into the following continuous stages to ensure a comprehensive balance between service quality (QoS) and resource utilization efficiency. Fig. 2 illustrates the complete system operation implemented in Cernet, primarily involving the process of the LFO-VNM algorithm. The system first monitors the CPU and memory utilization of each physical node in real time and combines this with SFC priority scores to identify overloaded nodes experiencing resource bottlenecks due to hosting high-priority services. Subsequently, for each VNF to be migrated, the system selects a set of mappable nodes from all non-overloaded nodes across the network based on the VNF's CPU and memory requirements and the remaining end-to-end latency budget of its associated SFC. For these candidate nodes, the system calculates a comprehensive priority score based on available computing resources, storage capacity, and network connectivity to calculate a comprehensive priority score, and the node with the highest score is selected to generate the initial migration mapping, completing the node priority deployment of VNFs on overloaded nodes (Algorithm 3). Next, the algorithm uses this initial mapping as the starting point for the fish swarm population, combines the current global deployment status and network resource information, and applies Algorithm 2 for global search—dynamically switching between foraging, following, aggregating, and dispersing behaviors to balance local fine-grained search and global multi-peak exploration, while continuously evaluating resource and QoS constraints. Meanwhile, Lagrange relaxation techniques are used to relax coupled constraints such as CPU, memory, and latency into the objective function. After each iteration, the corresponding multipliers are updated based on the degree of violation, and the constraint penalties are fed back into the fitness evaluation to guide the subsequent swarm solution away from the infeasible solution region. This process is repeated until the swarm converges or no nodes are overloaded, ultimately outputting the optimal VNF migration plan. Through the above steps, LFO-VNM achieves a closed-loop control process from priority-driven local initial mapping to global optimization of multiple behavior patterns, and then to dynamic constraint adjustment, demonstrating good reproducibility and scalability.

Stage one: Overload Detection and Migration Trigger. Monitor the CPU/memory utilization of all physical nodes to identify overloaded nodes; filter only the VNFs with the highest resource consumption or traffic contribution on these overloaded nodes to construct the set of VNFs to be migrated in this round. Specifically, when the resource utilization of node n exceeds its threshold θ, it is considered overloaded, as shown in the following formulas:

$$U_{cpu}(n) = \frac{R_{cpu\_used}(n)}{R_{cpu\_total}(n)} > \theta_{cpu} \tag{31}$$

$$U_{mem}(n) = \frac{R_{mem\_used}(n)}{R_{mem\_total}(n)} > \theta_{mem} \tag{32}$$

When a node becomes overloaded, the migration trigger mechanism is activated, and the system further analyzes the resource consumption of each VNF within the node to select the VNF instances that need to be migrated. The goal of this stage is to prevent network congestion and QoS degradation caused by resource overload.

Stage two: Candidate Target Node Selection. After determining which VNF instances need to be migrated, an appropriate target node must be selected for migration. The construction of the candidate node set is based on the following two constraints: 1. Resource Constraint: The target node must have sufficient resources to accommodate the VNF to be migrated. 2. VNF Support Constraint: The target node must support the specific VNF type. The candidate target node set can be expressed as:

$$N_{v_i}^{can} = \{n_j | R_{cpu}(n_j) \geq V_{cpu}(v_i) \cap R_{mem}(n_j) \geq V_{mem}(v_i)\} \tag{33}$$

Where $V_{cpu}(v_i)$ and $V_{mem}(v_i)$ represent the CPU and memory resources required by VNF v_i, respectively. $N_{v_i}^{can}$ represents the candidate target node set that satisfies the resource constraint. This stage ensures the feasibility of the migration and avoids migration failures due to insufficient resources at the target node.

Stage three: Fitness Function Design. In the proposed VNF migration optimization model, the core objective of the fitness function is to sum the three key performance indicators: migration cost, routing cost, and energy consumption cost, in order to comprehensively assess the overall performance of the candidate solutions. This function not only focuses on the efficiency of network resource utilization but also considers the extent to which service quality and resource constraints are satisfied during the migration process.

$$F(X) = C_{total}(X) + P(X, \lambda, \mu) \tag{34}$$

The term $P(X, \lambda, \mu)$ is used to suppress solutions that violate resource constraints, representing the degree of violation. This term is activated when the CPU or memory resources of a node are overloaded, thus increasing the fitness value and forcing the algorithm to move away from infeasible solutions. This design approach directly adds up the three types of costs and is suitable for multi-objective optimization scenarios that require balancing multiple performance factors without a clear prioritization. Its advantage lies in being easy to interpret and implement, while also effectively constraining the generation of infeasible solutions through the penalty factor λ. If the node resource usage exceeds its total capacity, the max function activates the penalty, thus increasing the fitness value.

Stage four: Multi-objective Optimization Solution. To further improve the global optimality of migration decisions, a hybrid solution method based on Lagrangian Relaxation and Artificial Fish Swarm Algorithm is used in this paper. Lagrangian Relaxation Principle: Lagrangian relaxation introduces multipliers to convert the constraints into penalty terms and add them to the objective function for easier solving. In VNF migration, the goal is to minimize migration cost, energy consumption, and routing cost, while satisfying node resource

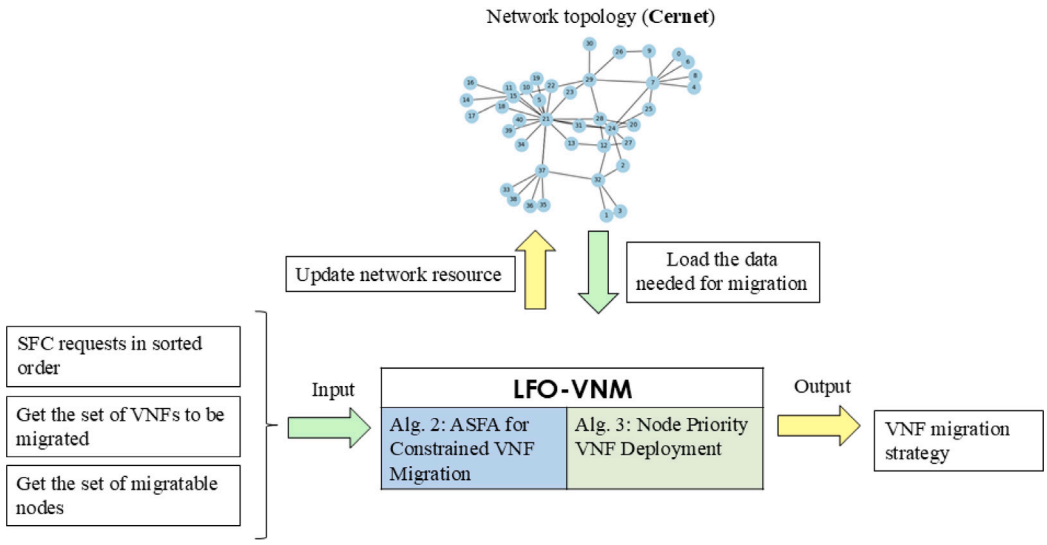


Fig. 2. Complete system operation implementation in Cernet.

constraints:

$$R_{cpu\_used}(n) \le R_{cpu\_total}(n) \tag{35}$$

$$R_{mem\_used}(n) \le R_{mem\_total}(n) \tag{36}$$

Lagrangian multipliers λ_n and μ_n are introduced to penalize the resource constraints:

$$P(X,\lambda,\mu) = F(X) + \sum_{n\in N}\left(\lambda_n \cdot \max(0, R_{cpu\_used}(n) - R_{cpu\_total}(n))\right) + \sum_{n\in N} \mu_n \cdot \max(0, R_{mem\_used}(n) - R_{mem\_total}(n)) \tag{37}$$

In the optimization process, to gradually increase the penalty for resource constraints, a penalty factor $penalty\_factor$ is introduced to dynamically adjust the update rate of the Lagrangian multipliers. It is defined as:

$$penalty\_factor = P_{base} \cdot (1 + r \cdot t) \tag{38}$$

where P_{base} is the base penalty factor, which controls the initial penalty strength. r is the penalty growth rate, which controls the speed of increase of the penalty factor after each iteration. t is the current iteration number. Update of Lagrangian Multipliers: After each iteration, the multipliers are updated as follows:

$$\lambda_n^{(t+1)} = \max(0, \lambda_n^t + \alpha \cdot penalty\_factor \cdot (R_{cpu\_used}(n) - R_{cpu\_total}(n))) \tag{39}$$

$$\mu_n^{(t+1)} = \max(0, \mu_n^t + \alpha \cdot penalty\_factor \cdot (R_{mem\_used}(n) - R_{mem\_total}(n))) \tag{40}$$

where α is the learning rate, which controls the update speed of the multipliers.

Fish Behaviors in Algorithm 2: The fish behaviors consist of four main actions: Prey, Swarm, Follow, and Random Walk. The mathematical descriptions for each behavior are provided below. Prey Behavior: The fish in the swarm move toward a better position based on the fitness function of the target area. If the fitness of the new position is better, the fish moves to the new position.

$$X_i(t+1) = X_i(t) + \text{step} \cdot \frac{(X_j(t) - X_i(t))}{\|X_j(t) - X_i(t)\|} \cdot \text{rand}() \tag{41}$$

Where $X_j(t)$ is a random new position within the fish's visibility range. If $f(X_j) > f(X_i)$, the fish moves to the new position; otherwise, it performs a random move. $f(X_i)$ represents the fitness at position X_i, and rand() is a random number between [0, 1].

Swarm Behavior: The fish exhibit a swarm behavior, moving toward the center of the surrounding fish to increase food concentration while avoiding overcrowding. The center of the fish within the visibility range X_c is calculated as:

$$X_c = \frac{\sum_{j=1}^{n_f} X_j}{n_f} \tag{42}$$

The movement equation is:

$$X_i(t+1) = X_i(t) + \text{step} \cdot \frac{(X_c - X_i(t))}{\|X_c - X_i(t)\|} \cdot \text{rand}() \tag{43}$$

If $f(X_c) > f(X_i)$ and $n_f/N < \delta$, the fish move toward the center. n_f represents the number of fish in the visibility range, and δ is the crowding threshold.

Follow Behavior: Fish follow the best individual within their perception range. The best fish position X_b in the current visibility range is determined as:

$$X_i(t+1) = X_i(t) + \text{step} \cdot \frac{(X_b - X_i(t))}{\|X_b - X_i(t)\|} \cdot \text{rand}() \tag{44}$$

Where X_b is the position of the fish with the highest fitness in the current visibility range.

Random Walk: If the Prey, Swarm, and Follow behaviors cannot find a better solution, the fish will perform a random walk:

$$X_i(t+1) = X_i(t) + \text{Visual} \cdot \text{rand}() \tag{45}$$

Adaptive Step Length and Visual Range: The adaptive step length step and the visual range Visual gradually decrease with the number of iterations to improve convergence accuracy:

$$\text{Visual}(t) = \text{Visual}_{\min} + (\text{Visual}_{\max} - \text{Visual}_{\min}) \cdot \left(1 - \frac{t}{T_{\max}}\right) \tag{46}$$

$$\text{step}(t) = \text{step}_{\min} + (\text{step}_{\max} - \text{step}_{\min}) \cdot \left(1 - \frac{t}{T_{\max}}\right) \tag{47}$$

Where $T_{\max}$ represents the maximum number of iterations. $\text{Visual}_{\min}$ and $\text{step}_{\min}$ represent the minimum visual range and step length, respectively.

4.2. The procedure of the algorithm

As shown in Algorithm 1, lines 1 to 16 correspond to the implementation process of the algorithm. In the main workflow of the LFO-VNM algorithm, the priority of the Service Function Chains (SFCs) is first calculated according to the formula (line 1) to determine which SFCs should be prioritized. Next, the SFC requests are processed, and the list of Virtual Network Functions (VNFs) to be migrated is constructed (line

Algorithm 1: Lagrangian Fish Optimization for VNF Migration(LFO-VNM) Algorithm

Input: Physical network graph $G(N, L)$, SFC requests, node resources
Output: VNF migration solution Mig^s

```
Calculate SFC priority based on Eqs. (7) ;
Get the list of VNFs to be migrated ;
Initialize Lagrangian multipliers λ[node] for each node ;
for iteration ← 1 to max_iterations do
    population, best_solution ← Call Algorithm 2 ;
    penalty_factor ← according to Eqs. (38) ;
  for each node in N do
      Calculate cpu_violation ;
      Calculate memory_violation ;
      adjustment ← penalty_factor × (cpu_violation +
      memory_violation) ;
      λ[node] ← according to Eqs. (39)(40) ;
  end
end
return Mig^s ;
```

Algorithm 2: ASFA for Constrained VNF Migration

Input: Population, fitness function, constraints
Output: Population, best solution

```
Initialize population ← Call Algorithm 3 ;
pareto_front ← Ø;
Initialize BestSolution ← Population[0] ;
step ← according to Eqs. (46) ;
visual ← according to Eqs. (47) ;
for each fish in population do
    NewSolution ← fish_behaviors() ;
    if check constraints (14-24) then
        update fitness ;
        if NewFitness < BestFitness then
            BestFitness ← NewFitness ;
        end
    end
  for each individual in population do
      if not dominated by any other individual in population
      then
          add individual to pareto_front ;
      end
  end
end
return population, best_solution ;
```

Algorithm 3: Node Priority VNF Deployment

Input: SFC requests, node resources
Output: Placement of VNFs in the network

```
placement ← Ø; ;
Calculate node priority; ;
for each SFC in SFC_requests do
  for each VNF in SFC.vnfs do
      Get candidate nodes; ;
      Select target_node; ;
      placement[VNF] ← target_node; ;
      update node resources; ;
      Calculate node priority; ;
  end
end
return placement; ;
```

2). Then, the Lagrangian multipliers are initialized for each network node (line 3) to dynamically adjust and constrain resource allocation. Algorithm 2 is subsequently called for global search and optimization, while the calculated penalty factors are used to correct resource violations, gradually converging to the optimal VNF migration solution (lines 4–14).

Algorithm 2 first calls Algorithm 3 to generate an initial population, where each individual represents a VNF migration solution, and creates a Pareto front set to store non-dominated solutions (lines 1–3). Next, the search parameters, including step size and visual range, are set (lines 4–5). Then, each individual in the population is iterated through, performing a random walk to generate new solutions. The constraints for each solution are checked, and if the solution satisfies the constraints, its fitness is calculated. The fitness is compared with the current best solution, and if necessary, the best solution and the Pareto front set are updated, ensuring a combination of global search and local fine-tuning (lines 6–14).

Algorithm 3 Begin with an empty placement map and compute a static priority score for every node based on residual CPU/memory, connectivity, and load-balancing factors (lines 1–2).construct a set of candidate nodes that satisfy CPU/memory capacity, VNF type matching, and the remaining end-to-end latency budget of the corresponding SFC(line 3) its candidate node set by enforcing CPU, memory, and end-to-end delay budget constraints (line 4). Select the candidate with the highest node-priority score (line 5), assign the VNF there (line 6), then deduct its resource demands and recompute that node's score (line 7). Once all VNFs are placed, output this initial, constraint-aware mapping to Algorithm 2 for further refinement(lines 10–12).

5. Evaluation and discussion of results

In this section, we provide a detailed analysis and comparison of the performance of the proposed algorithm with other existing solutions. To validate the effectiveness of our algorithm in real-world application scenarios, we evaluate it on real network topologies. We use similar network, VNF, and SFC configurations to ensure the generality of the data.

5.1. Experimental and simulation parameter settings

Network topology: In this study, we use two real network topologies from the Zoo [33] database, namely Cernet (with 41 nodes and 59 links) and Cogentco (with 197 nodes and 245 links). The nodes in the network are configured with CPU and memory resources in the range of [500, 1000] units. The bandwidth capacity of all physical links is set to [1000, 1500] Mbps, and the propagation delay is between [1, 10] ms.

SFC and VNF: In this paper, we define ten types of VNFs, with each SFC consisting of [2, 8] VNFs. The service rate ranges from [1, 10] Gbps. The maximum tolerable delay for services is divided into two categories: [80, 201] ms and [300, 501] ms, representing requests with different quality of service requirements. The source and destination nodes are randomly selected from the network topology. Additionally, the CPU and memory resource requirements for VNFs are configured according to the service rate, with every 10 Mbps of service requiring [1, 5] units of resource.

Algorithm parameters: In this simulation, we consider CPU, memory, and bandwidth of the physical network as equally important, and set ω_1, ω_2, and ω_3 to 1/3. Furthermore, in the migration mechanism, we set the node overload threshold to 0.4 to achieve more VNF migration across nodes effectively, as shown in Fig. 3.

5.2. Algorithm benchmark

In this simulation, our primary goal is to evaluate the overhead caused by VNF migration. We introduce two VNF migration algorithms, and the details of these algorithms are as follows:

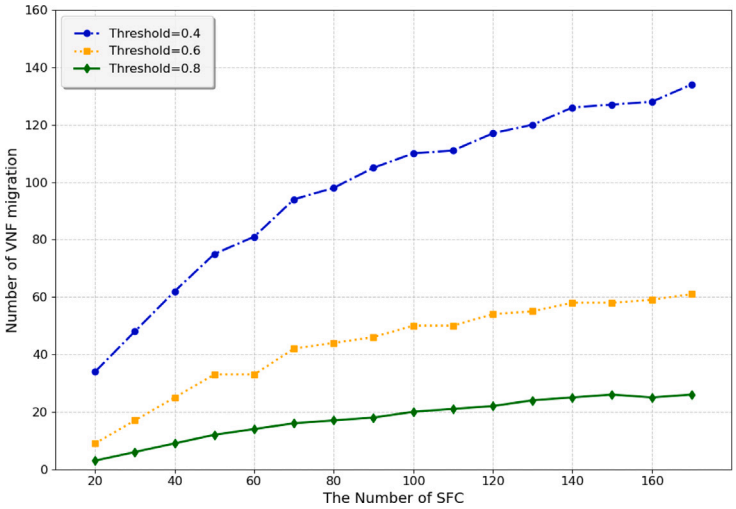


Fig. 3. Number of VNF migration.

LFO-VNM_Ran is a controlled variant of our proposed LFO-VNM algorithm in which the node-priority deployment step (Algorithm 3) is replaced by random placement. This isolates the benefit of our priority-based deployment policy on both runtime and migration overhead.

MSH-OR [25] represents a multi-stage, rule-based heuristic: it selects VNFs with the highest service rates on overloaded nodes and migrates them to lower-load destinations, minimizing end-to-end SFC delay while balancing network load. Its fast, greedy approach provides a load-aware benchmark.

DLAPM [18] exemplifies evolutionary optimization via a genetic algorithm that dynamically chooses subsets of VNFs for latency-aware migration, trading off total SFC delay against migration cost.

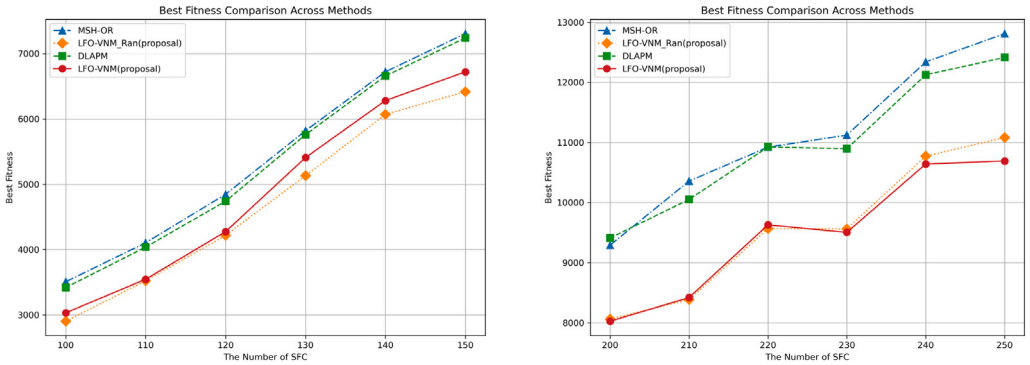
We selected MSH-OR and DLAPM as baselines because they respectively represent two dominant VNF migration paradigms—heuristic and evolutionary—and both explicitly target node load balancing. Although they were originally designed to optimize "migration cost + latency", they can be directly evaluated in our extended "migration cost + routing cost + energy cost" objective space. Under the same topology and workload, this alignment ensures that observed performance differences reflect each algorithm's inherent advantages rather than mismatched evaluation criteria. By benchmarking LFO-VNM_Rand, MSH-OR, and DLAPM, we demonstrate that our combined node-priority deployment and global multi-objective optimization framework achieves lower overall migration overhead and faster runtime.

Fig. 4 shows the minimization results of these algorithms in terms of optimal total cost. The best fitness reflects the performance of the optimization algorithm in achieving the objective. A lower total cost indicates better optimization performance. The LFO-VNM method demonstrates significant fitness optimization capability under both network topologies, effectively reducing the total cost. Particularly in large-scale SFC scenarios, it outperforms MSH-OR and DLAPM by 13.6% and 13.8%, respectively, showing a clear advantage. Compared to the LIFO-VNM_Ran method, LFO-VNM exhibits better fitness, and under the Cogentco network topology, this difference gradually increases as the number of SFCs increases. The fitness is 3.4% to 4.7% lower than LFO-VNM_Ran.

Fig. 5 shows the minimization results of these algorithms in terms of optimal migration cost. On the Cernet network topology, the migration cost of LFO-VNM increases by 4.6% compared to the MSH-OR method. Although the migration cost has increased, this increase is acceptable considering the optimization of fitness. Compared to the LIFO-VNM_Ran method, the migration cost of LFO-VNM increases by 2.1%, but the improvement in fitness and global optimization capability suggests that this cost is justified. Compared to the DLAPM method, the migration cost of LFO-VNM increases by 7.1%, but DLAPM does not show significant advantages in migration cost optimization. On the Cogentco network topology, the migration cost of LFO-VNM is 12.3% higher than MSH-OR, but compared to the LIFO-VNM_Ran method, the migration cost decreases by 5.2%, demonstrating an advantage in migration optimization. Compared to DLAPM, the migration cost increases by 13.8%, but LFO-VNM provides a more comprehensive optimization capability, compensating for the increase in migration cost. Although LFO-VNM shows a slight increase in migration cost, its significant advantages in fitness and other key metrics make this cost acceptable. Especially when compared to the LIFO-VNM_Ran method, LFO-VNM shows better migration efficiency.

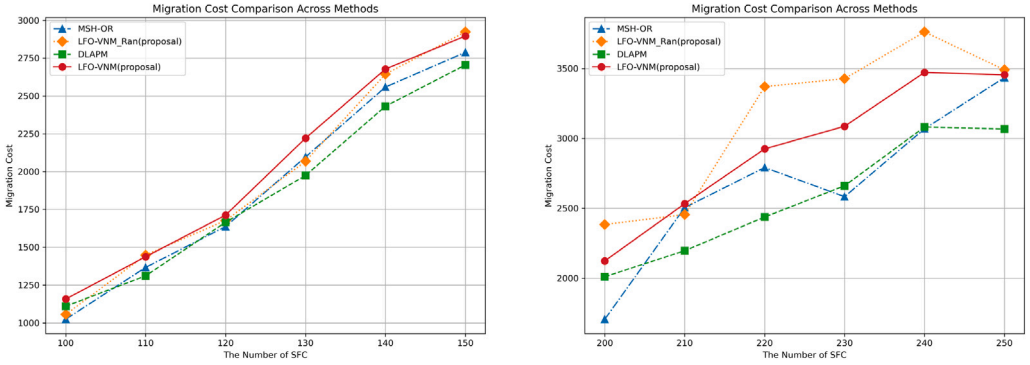
Fig. 6 shows the minimization results of these algorithms in terms of optimal routing cost. On the Cernet network topology, the routing cost of LFO-VNM increases by 13.2% compared to MSH-OR, but this increase is a necessary cost after optimizing other metrics. Compared to the LIFO-VNM_Ran method, the routing cost increases by 4.8%, which is still within an acceptable range, and the global optimization effect is more comprehensive. Compared to the DLAPM method, the routing cost increases by 17.6%, but LFO-VNM demonstrates stronger performance in overall optimization. On the Cogentco network topology, the routing cost of LFO-VNM increases by 9.6% compared to MSH-OR and by 5.2% compared to LFO-VNM_Ran. However, compared to DLAPM, the increase in routing cost is smaller, and LFO-VNM still achieves a good balance with a more comprehensive optimization effect. The routing cost of LFO-VNM slightly increases under both network topologies, but this increase is the cost of optimizing other metrics, and its overall optimization effect remains superior to traditional methods.

Fig. 7 shows the minimization results of these algorithms in terms of optimal energy cost. This metric measures the cumulative node energy consumption generated by all running VNF instances in the network during the subsequent scheduling cycle after migration adjustment is complete. On the Cernet network topology, LFO-VNM demonstrates exceptional energy-saving capability, reducing energy consumption by 56.2% compared to MSH-OR. Compared to DLAPM and LFO-VNM_Ran, energy consumption is also reduced by 56.2% and 2.9%, respectively. This advantage highlights the significant energy efficiency of LFO-VNM, particularly in large-scale network environments, where it can effectively reduce energy consumption. On the Cogentco network topology, LFO-VNM also shows outstanding energy efficiency, reducing energy consumption by 52.1% compared to MSH-OR. Its energy consumption is the same as that of DLAPM, and there is minimal change in energy consumption compared to LFO-VNM_Ran, demonstrating its energy efficiency advantage in comprehensive optimization. In both



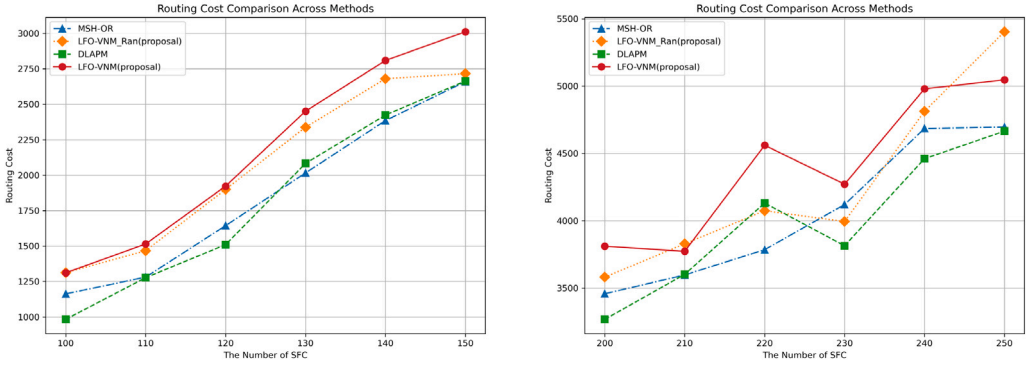
(a) Cernet with 41 nodes (b) Cogentco with 197 nodes

Fig. 4. Best fitness of algorithms(minimal total cost found).



(a) Cernet with 41 nodes (b) Cogentco with 197 nodes

Fig. 5. Migration cost of algorithms.



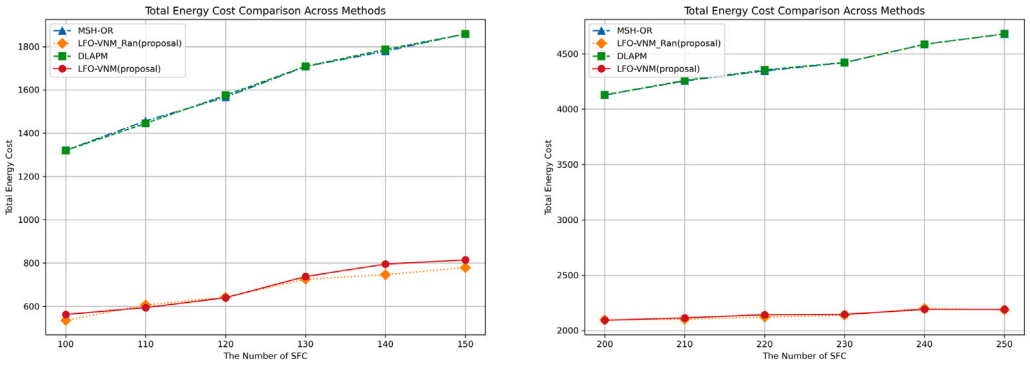
(a) Cernet with 41 nodes (b) Cogentco with 197 nodes

Fig. 6. Routing cost of algorithms.

network topologies, the LFO-VNM method exhibits a significant advantage in energy-saving.

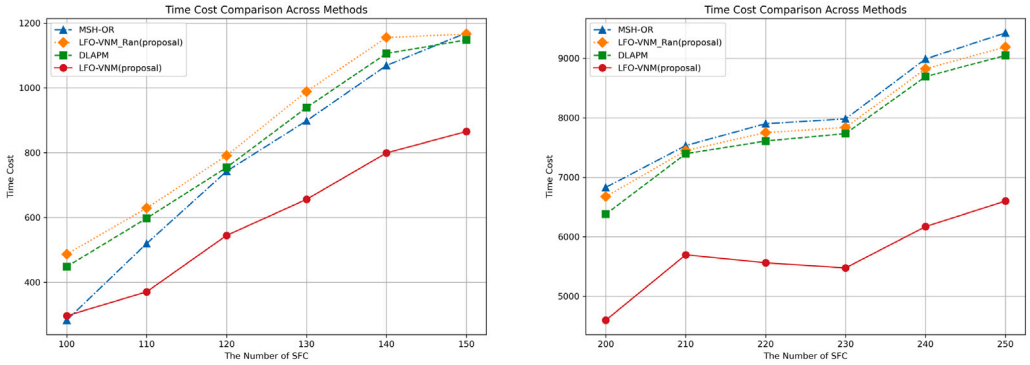
Fig. 8 shows the minimization results of these algorithms in terms of runtime. On the Cernet network topology, LFO-VNM significantly reduces computation time, with a 26.0% reduction compared to MSH-OR. Compared to LFO-VNM_Ran and DLAPM, the time overhead is reduced by 31.1% and 30.6%, respectively, demonstrating a significant improvement in computation efficiency with LFO-VNM. On the Cogentco network topology, LFO-VNM also shows a significant advantage in computation time, with the time cost reduced by 31.4% compared to MSH-OR and by 30.2% compared to LFO-VNM_Ran, greatly improving the efficiency of the optimization process. LFO-VNM significantly reduces time overhead, exhibiting a clear computational efficiency advantage when handling large-scale SFCs. Compared to other methods, LFO-VNM provides optimization results in a shorter time.

The LFO-VNM algorithm demonstrates significant optimization capability across multiple network topologies. Compared to other traditional methods, LFO-VNM shows superior overall optimization performance in key metrics such as total cost, migration cost, routing cost, energy consumption, and computation time, particularly in large-scale



(a) Cernet with 41 nodes (b) Cogentco with 197 nodes

Fig. 7. Energy cost of algorithms.



(a) Cernet with 41 nodes (b) Cogentco with 197 nodes

Fig. 8. Running time of algorithms.

SFC scenarios, where its advantages in energy efficiency and computational efficiency are especially notable. Although migration cost and routing cost increase in some cases, these increases are acceptable when considering the improvements in fitness and global optimization capability. Therefore, the LFO-VNM algorithm exhibits strong practicality and application potential in balancing multiple optimization objectives and is well-suited for SFC optimization in large-scale network environments.

6. Conclusion

With the continuous development of Network Function Virtualization (NFV) technology, the migration of Virtual Network Functions (VNFs) and the optimization of Service Function Chains (SFCs) have become particularly important in dynamic network environments. This study proposes a VNF migration algorithm that combines priority awareness and multi-objective optimization, aiming to address the challenges faced by traditional methods in dynamic environments, including resource overload, service quality assurance, and trade-offs between multiple objectives. By constructing a Mixed-Integer Linear Programming (MILP) model and combining Lagrangian Relaxation and the Artificial Fish Swarm Algorithm (AFSA), we systematically quantify the impact of VNF migration on network performance, migration cost, routing overhead, and energy consumption. Experimental results show that, compared to several existing algorithms, the proposed LFO-VNM method demonstrates excellent optimization performance across various network topologies. Although migration costs increase slightly in some cases, the improvement in overall optimization capability makes this increase acceptable. Particularly in large-scale network environments, the LFO-VNM method exhibits significant advantages in computational efficiency and energy efficiency. In future research, we aim to integrate machine learning algorithms to predict network load changes and further optimize resource scheduling, with the goal of enhancing the efficiency and stability of migration decisions and exploring more optimal solutions.

CRediT authorship contribution statement

Yu Luo: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Data curation, Conceptualization. **Zhaogang Shu:** Writing – review & editing, Project administration, Funding acquisition. **Shuwu Chen:** Funding acquisition, Formal analysis. **Qiang Tu:** Validation. **Xianzhang Wu:** Validation, Methodology, Data curation. **Qingjie Lin:** Validation.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: The authors declare that they have no known competing finan- cial interests or personal relationships that could have appeared to influence thework reported in this paper.

Acknowledgments

This work was supported by the Project of Fu-Xia-Quan Collaborative Innovation Platform in Fujian Province, China (No. 2025E3012), the Project of Industry-University-Institute Cooperation in Colleges and Universities in Fujian Province, China (No. 2024H6007), the Project of Industry-University-Institute Joint Innovation in Colleges and Universities in Fujian Province, China (No. 2024H6030), the Industry-Research Project from Network Communication Company, China (No. KH230139A, No. KH240131A), and the Educational and Scientific Research Project for Young Teachers of Fujian Province (No. KLY24207XA).

Data availability

The authors do not have permission to share data.

References

[1] R. Mijumbi, J. Serrat, J.L. Gorricho, et al., Network function virtualization: State-of-the-art and research challenges, IEEE Commun. Surv. & Tutorials 18 (1) (2016) 236–262.
[2] M. Mechtri, C. Ghribi, O. Soualah, et al., NFV orchestration framework addressing SFC challenges, IEEE Commun. Mag. 55 (6) (2017) 16–23.
[3] Y. Li, M. Chen, Software-defined network function virtualization: A survey, ISSS J. Micro Smart Syst. 3 (2015) 2542–2553.
[4] M. Rashid, S. Joan, G. Juan-Luis, B. Niels, D.T. Filip, B. Raouf, et al., Network function virtualization: State-of-the-art and research challenges, IEEE Commun. Surv. Tutorials 18 (1) (2015) 236–262.
[5] X. Wang, X. Chen, C. Yuen, et al., Delay-cost tradeoff for virtual machine migration in cloud data centers, J. Netw. Comput. Appl. 78 (2017) 62–72.
[6] Y. Wang, Z. Shu, S. Chen, J. Lin, Z. Zhang, A cost and demand sensitive adjustment algorithm for service function chain in data center network, Comput. Netw. 242 (4) (2024) 110254, SCI, IF: 5.493.
[7] H. Feng, Z. Shu, T. Tarik, Y. Wang, Z. Liu, An aggressive migration strategy for service function chaining in the core cloud, IEEE Trans. Netw. Serv. Manag. (2022) http://dx.doi.org/10.1109/TNSM.2022.3231186.
[8] Z. Liu, Z. Shu, S. Chen, Y. Zhong, J. Lin, Low-latency virtual network function scheduling algorithm based on deep reinforcement learning, Comput. Netw. 246 (1) (2024) 110418, SCI, IF: 5.493.
[9] B. Yi, X. Wang, M. Huang, et al., Design and implementation of network-aware VNF migration mechanism, IEEE Access 8 (2020) 44346–44358.
[10] F. Shoura, A. Gharaibeh, S. Alouneh, Optimization of migration cost for network function virtualization replacement, in: 2020 21st International Arab Conference on Information Technology, ACIT, 2020, pp. 1–7.
[11] S. Xu, B. Liao, B. Hu, et al., A reliability-and-energy-balanced service function chain mapping and migration method for internet of things, IEEE Access 8 (2020) 168196–168209.
[12] P. Sun, J. Lan, J. Li, et al., Efficient flow migration for NFV with graph-aware deep reinforcement learning, Comput. Netw. 183 (2020) 107575.
[13] V. Eramo, E. Miucci, M. Ammar, et al., An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures, IEEE/ACM Trans. Netw. 25 (4) (2017) 2008–2025.
[14] V. Eramo, M. Ammar, F.G. Lavacca, Migration energy aware reconfigurations of virtual network function instances in NFV architectures, IEEE Access 5 (2017) 4927–4938.
[15] S. Song, C. Lee, H. Cho, et al., Clustered virtualized network functions resource allocation based on context-aware grouping in 5G edge networks, IEEE Trans. Mob. Comput. 19 (5) (2019) 1072–1083.
[16] W. Liang, L. Cui, F.P. Tso, Low-latency service function chain migration in edge-core networks based on open jackson networks, J. Syst. Archit. 124 (2022) 102405.
[17] D. Cho, J. Taheri, A.Y. Zomaya, et al., Real-time virtual network function (VNF) migration toward low network latency in cloud environments, 2017 IEEE 10th Int. Conf. Cloud Comput. (CLOUD) (2017) 798–801.
[18] D. Liu, Z. Zhou, D. Zhang, et al., Efficient service reconfiguration with partial virtual network function migration, Comput. Netw. 241 (2024) 110205.
[19] S.N. Afrasiabi, A. Ebrahimzadeh, N. Promwongsa, et al., Cost-efficient cluster migration of VNFs for service function chain embedding, IEEE Trans. Netw. Serv. Manag. 21 (1) (2024) 979–993.
[20] H. Qu, K. Wang, J. Zhao, Priority-awareness VNF migration method based on deep reinforcement learning, Comput. Netw. 208 (2022) 108866.
[21] J.L. Vieira, E.L.C. Macedo, A.L.E. Battisti, et al., Mobility-aware SFC migration in dynamic 5G-edge networks, Comput. Netw. 250 (2024) 110571.
[22] B. Yi, X. Wang, M. Huang, et al., A multi-criteria decision approach for minimizing the influence of VNF migration, Comput. Netw. 159 (2019) 51–62.
[23] W. Chen, Z. Wang, H. Zhang, et al., Cost-efficient flow migration for SFC dynamical scheduling in geo-distributed clouds, Comput. Netw. 249 (2024) 110496.
[24] Q. Liu, L. Tang, T. Wu, et al., Deep reinforcement learning for resource demand prediction and virtual function network migration in digital twin network, IEEE Internet Things J. 10 (21) (2023) 19102–19116.
[25] B. Li, B. Cheng, X. Liu, et al., Joint resource optimization and delay-aware virtual network function migration in data center networks, IEEE Trans. Netw. Serv. Manag. 18 (3) (2021) 2960–2974.
[26] A. Karim, J. Ema, T. Yasmin, et al., Latency and cost-aware deployment of dynamic service function chains in 5G networks, in: 2023 International Symposium on Networks, Computers and Communications, ISNCC, 2023, pp. 1–6.
[27] F. Zhang, H. Lu, F. Guo, et al., Traffic prediction based VNF migration with temporal convolutional network, in: 2021 IEEE Global Communications Conference, GLOBECOM, 2021, pp. 1–6.
[28] J. Xia, Z. Cai, M. Xu, Optimized virtual network functions migration for NFV, in: 2016 IEEE 22nd International Conference on Parallel and Distributed Systems, ICPADS, IEEE, 2016, pp. 340–346.
[29] M. Xiao-lei, S. Xuan-liang, Z. Zhao, et al., Optimization of autonomous bus scheduling based on Lagrangian relaxation, China J. Highw. Transp. 32 (12) (2019) 10–24.
[30] N. Mehdi, S. Ghodrat, S. Mehdi, et al., Artificial fish swarm algorithm: a survey of the state-of-the-art, hybridization, combinatorial and indicative applications, Artif. Intell. Rev. 42 (4) (2014) 965–997.
[31] Y. Yue, X. Tang, Z. Zhang, et al., Virtual network function migration considering load balance and SFC delay in 6G mobile edge computing networks, Electronics 12 (12) (2023) 2753.
[32] Y. Yue, X. Tang, W. Yang, et al., Virtual network function migration considering load balance and SFC delay in cloud datacenter, in: 2023 IEEE 16th International Conference on Cloud Computing, CLOUD, IEEE, 2023, pp. 1–6.
[33] S. Knight, H.X. Nguyen, N. Falkner, et al., The internet topology zoo, IEEE J. Sel. Areas Commun. 29 (9) (2011) 1765–1775.

Yu Luo is currently pursuing a Master's degree in Computer Science and Technology at Fujian Agriculture and Forestry University. His research includes Virtual Network Function (VNF) migration, network resource optimization, and multi-objective optimization in dynamic networks. He is currently working on developing priority-aware and cost-efficient VNF migration strategies to enhance network performance and reduce migration costs in complex network environments.

Zhaogang Shu is currently an Associate Professor at the College of Computer and Information Science, Fujian Agriculture and Forestry University, Fuzhou, China. He also is the director of Cloud Computing Lab, Fujian Agriculture and Forestry University. He received B.S. and M.S. degrees in computer science from Shantou University, China in 2002 and 2005 respectively. He also received Ph.D. degree from South China University of Technology, Guangzhou, China, in 2008. From Sept. 2008 to July 2012, he worked as a senior engineer and project manager at Ruijie Network Corporation, Fuzhou, China. From Oct. 2018 to Oct. 2019, he worked as a visiting professor in MOSIAC lab at the Department of Communications and Networking, Aalto University, Finland. He directed more than 10 research projects and was the author of more than 30 papers and 5 patents. His research interests include software-defined network, network function virtualization, 5G network and next generation network architecture, network security, machine learning based network optimization, cloud computing and edge computing. He serves as the reviewers of many famous journals on network and communications, including IEEE Network, IEEE/ACM Transactions on Networking, IEEE Transactions on Network Service and Management, ACM/Springer Mobile Networks, Elsevier Computer networks and so on. He also is the member of CCF (China Computer Federation) and Fujian Computer Society.

Shuwu Chen is currently a professor at the College of Computer and Information Science, Fujian Agriculture and Forestry University, Fuzhou, China. He also is the director of Innovation Lab of IoT technology, Fujian Agriculture and Forestry University. He received bachelor's degree in industrial automation from Chang'an University, China, in 1998. And, he received master's degree in radio physics from Xiamen University, China, in 2003. He is the Co founder of Xiamen FourFaith Communication Technology Co., Ltd., which focus on IoT technology and solutions. He directed dozens research projects and was the author of more than 10 patents. His research interests include IoT technology, edge computing and AI algorithm.

Qiang Tu received the Ph.D. degree in communication and information systems from the Department of Communication Engineering, Xiamen University, Xiamen, China, in 2024. He is currently a Lecturer in the College of Computer and Information Sciences, Fujian Agriculture and Forestry University. His current research interests include the general area of the Internet of Things and underwater acoustic signal processing.

Xianzhang Wu received the B.S. degree in mathematics and applied mathematics from Minjiang University, Fuzhou, China, in 2014, the M.S. degree in applied mathematics from Fuzhou University, Fuzhou, in 2018, and the Ph.D. degree in communication and information system from Sun Yat-sen University, Shenzhen, China, in 2023. He is currently a Lecturer with the College of Computer and Information Science, Fujian Agriculture and Forestry University, Fuzhou. His research interests include combinatorics, graph theory, caching networks, and their interactions.

Qingjie Lin is currently pursuing the master's degree at computer science and technology, Fujian Agriculture And Forestry University. His research interests include software defined networking and routing-aware algorithms. He is currently working on routing optimization problems in data center networks.