This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

Efficient Service Function Chaining in Low Earth Orbit Satellite Networks with Genetic Simulated Annealing-based Particle Swarm Optimization

Haitao Yuan, *Senior Member, IEEE*, Dengyu Guo, *Student Member, IEEE,* Jing Bi, *Senior Member, IEEE,* Jia Zhang, *Senior Member, IEEE*, and Rajkumar Buyya, *Fellow, IEEE*

***Abstract*—Low Earth Orbit (LEO) satellite networks provide low-latency, high-bandwidth, and broad-coverage Internet access services to real-time applications for remote and rural regions. It becomes challenging to achieve Virtual Network Function (VNF) deployment in such networks, which is critical to enable efficient service provisioning in space-based systems. Optimizing VNF placement across satellites is complicated by limited computational resources, stringent communication constraints, and complex service requirements. This work addresses it by formulating it as a binary nonlinear program to maximize the service acceptance rate while ensuring balanced resource utilization. To solve it, this work proposes a hybrid metaheuristic algorithm called Genetic Simulated annealing-based Particle Swarm Optimization (GSPSO), which integrates the exploration capabilities of genetic algorithms, local optima avoidance of simulated annealing, and fast convergence of particle swarm optimization. Experimental results demonstrate that GSPSO reduces the mean fitness by 11.4%, 98.60%, 99.97%, 99.99%, 99.78%, and 98.73% compared with particle swarm optimization (PSO), self-adaptive PSO, simulated annealing, tyrannosaurus optimization algorithm, two-stage heuristic, and FAGD-Lite, respectively. Additional scalability and sensitivity experiments with up to 30 satellites and diverse Service Function Chaining (SFC) orchestration settings further demonstrate the robustness of GSPSO under larger and more complex LEO satellite network scenarios.**

***Index Terms*—LEO satellite networks, load fairness, service function chain orchestration, meta-heuristic optimization, space-air-ground integrated networks.**

I. Introduction

As human activities and communication demands expand, traditional terrestrial networks face limitations in coverage and real-time performance. Space-Air-Ground Integrated Networks (SAGINs), which integrate space, air, and ground segments, offer a solution for ubiquitous connectivity and are crucial for 6G networks. Low Earth Orbit (LEO) satellite networks, as a key part of the space segment, provide low-latency and high-speed Internet, and broad geographic coverage, bridging the digital divide in remote areas with several advantages. Their proximity to Earth (500–2,000 km) enables lower communication latency, crucial for real-time applications. Their global coverage extends to remote and rural regions, offering Internet access where terrestrial infrastructure is lacking. Additionally, they support high-bandwidth services such as telemedicine, environmental monitoring, disaster response, and autonomous driving, which are vital for modern societies.

However, the dynamic nature of LEO satellite networks poses significant challenges. The time-varying topology, propagation delays, and radio environment, coupled with the fast movement of satellites causing link interruptions and topological changes, render traditional static network management strategies ineffective. Efficiently orchestrating Service Function Chains (SFCs) in such a dynamic environment is crucial [1]. SFC orchestration must consider diverse service requests and resource constraints while adapting to topological changes to ensure continuous and high-quality service delivery. Moreover, the limited computing and communication resources on LEO satellites require efficient allocation and scheduling to meet service demands. Overloading individual satellites can lead to rapid equipment saturation, compromising the stability of the entire constellation and increasing operational cost [2].

Current studies on SFC orchestration in SAGINs predominantly focus on ground networks or static environments. Researchers have primarily focused on enhancing network coverage and exploring service provisioning in heterogeneous network environments [1], [3]–[6]. Research on service orchestration in dynamic LEO satellite networks remains limited [7]. Traditional search-based heuristic algorithms struggle with high computational complexity in large-scale networks, failing to meet real-time requirements. As a methodology capable of addressing intricate optimization challenges, deep reinforcement learning (DRL) has emerged as a focal point of research interest. DRL's particular strength lies in its applicability to problems structured as Markov Decision Processes (MDP), allowing algorithms to learn the optimal policies through environmental interaction. This framework facilitates swift and effective decision processing within SFC operational sequences, enabling systems to adapt to changing conditions

This work was supported by the Fundamental Research Funds for the Central Universities, the National Natural Science Foundation of China under Grant 62473014, and the Beijing Natural Science Foundation under Grant L233005. (Corresponding author: Haitao Yuan)

H. Yuan and D. Guo are with the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. (e-mail: yuan@buaa.edu.cn; 20373748@buaa.edu.cn).

J. Bi is with the College of Computer, Beijing University of Technology, Beijing 100124, China. (e-mail: bijing@bjut.edu.cn).

J. Zhang is with the Department of Computer Science, Southern Methodist University, Dallas, TX 75206, USA. (e-mail: jiazhang@smu.edu).

R. Buyya is with the Quantum Cloud Computing and Distributed Systems (qCLOUDS) Lab, School of Computing and Information Systems, University of Melbourne, Melbourne, VIC 3010, Australia (e-mail: rbuyya@unimelb.edu.au).

Copyright (c) 2026 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

while maintaining performance objectives [8]–[10]. Existing DRL methods also fall short in spatial awareness and feature extraction for graph-structured data [11]–[14]. Thus, there is a need for new algorithms, like improved heuristic algorithms, to achieve efficient, flexible, and adaptive SFC orchestration.

To address the above challenges, this work investigates the SFC orchestration problem in dynamic LEO satellite networks. Main contributions are summarized as:

- This work formulates the SFC orchestration problem in LEO satellite networks as a binary nonlinear optimization problem, which considers VNF placement, service admission, computing resource constraints, bandwidth constraints, propagation delay constraints, and stable inter-satellite link constraints, improving the service acceptance rate and load fairness among satellites.
- This work proposes a Genetic Simulated annealing-based Particle Swarm Optimization algorithm (GSPSO) to solve the formulated problem. GSPSO integrates genetic crossover and mutation operations, the Metropolis acceptance rule of simulated annealing, and the velocity-position updating mechanism of particle swarm optimization, enhancing population diversity, local optimum avoidance, and convergence efficiency.
- This work conducts extensive experiments to evaluate GSPSO, achieving the best fitness value and load fairness among the compared methods, while maintaining a high service acceptance rate and competitive computational efficiency among iterative metaheuristic algorithms.

The rest of the paper is organized as follows. Section II presents the related studies. Section III shows the Software Defined Network (SDN)/Network Functions Virtualization (NFV)-enabled LEO satellite network. Section IV describes the details of GSPSO. Section V gives simulation results. Section VI concludes this work.

II. Related Work

This section discusses the related studies on SFC orchestration from terrestrial data centers to broader SAGINs. These studies are categorized into three types: mathematical programming, heuristic and metaheuristic algorithms, and machine learning-based methods.

A. Mathematical programming for optimal placement

A prominent approach involves formulating SFC orchestration as a precise optimization problem using models such as Integer Linear Programming (ILP), Mixed-Integer Linear Programming (MILP), or their non-linear variants. Researchers have formulated ILP models to jointly minimize power consumption and spectrum usage [16], minimize the total deployment cost [24], or reduce service cost [21]. Similarly, studies [20] and [25] adopt ILP/Mixed-Integer Nonlinear Programming (MINLP) to optimize the delay and resource consumption. The study [34] focuses on energy efficiency by formulating an ILP model to jointly optimize VNF placement and traffic steering in telecom networks. The study [35] addresses dynamic VNF placement and routing by formulating an ILP problem that jointly minimizes resource consumption while satisfying multiple QoS constraints, including delay, packet loss, and jitter. They propose a polynomial algorithm to achieve near-optimal performance based on the Markov approximation, achieving a near-optimal long-term average performance. The study [36] tackles a joint problem of admission control and SFC embedding by designing an MILP problem for splittable VNFs and multi-path routing to enhance network performance. They formulate a mixed-integer linear program and employ relaxation, reformulation, and successive convex approximation to solve it. The study [37] addresses the inefficiency of single-resource-dimension models by proposing an MILP for network service embedding that optimizes placement across multiple resources. They propose a VNF bundle scheme that yields balanced and resource-wise VNF bundles to increase VNF placement, yielding notable resource efficiency.

Their primary limitation lies in NP-hard nature, resulting in computational complexity that grows exponentially with the network scale and problem size. While feasible for offline planning in static terrestrial networks, their computational demands render applications in highly dynamic LEO satellite environments, where network topology changes continuously. Compared with these studies, our work considers the distinctive resource limitations and link characteristics of LEO satellite networks, and jointly incorporates VNF placement, routing, propagation delay, computing capacity, link bandwidth, and stable-link constraints into a unified SFC orchestration model.

B. Heuristic and metaheuristic algorithms for scalable solutions

To overcome the scalability of exact methods, a vast body of work leverages heuristic and metaheuristic algorithms, providing near-optimal solutions in a reasonable time. Efforts include developing sophisticated modeling tools like multi-functional time-expanded graphs to characterize communication, storage, and computation [4], and designing specific placement strategies like multipath chaining to reduce latency [18]. Other heuristic methods focus on delay-aware mapping with prediction [29], or efficient service recovery *via* matching games [26]. This category also includes complex metaheuristics such as particle swarm optimization [15], meteor shower optimization [17], and hybrid algorithms to balance resource consumption and maximize profit [1], [19], [28]. The study [38] employs ant colony optimization to address the SFC composition problem, minimizing the total routing cost of 5G network environments. Small-size and medium-size 5G networks are evaluated, demonstrating its performance. The study [39] develops a two-phase optimization solution to solve the VNF placement problem by minimizing total network resource consumption while meeting delay requirements of requests. The solution includes a mapping phase to map SFC requests on servers and an adjustment phase to optimize the placement of VNFs and requests, enhancing network resource utilization. The study [40] introduces Weight-Oriented Embedding and Weight-Oriented Ratio Embedding to solve the slice-based SFC embedding problem for maximizing flow acceptance ratios and minimizing network costs. Simulation results prove

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

that it outperforms state-of-the-art algorithms and achieves near-optimal performance. The study [41] investigates the VNF placement problem under randomized data traffic, and designs a greedy approach and a simulated annealing-based one to minimize total end-to-end delay by determining the optimal deployment locations for VNFs, yielding 38.8% less end-to-end delay than genetic algorithm.

Although computationally efficient, the performance of these algorithms is often highly dependent on problem-specific design and parameter tuning. They are typically designed based on the assumptions of ground networks or SAGINs. They often fail to adequately model the most stringent constraints of the LEO environment with limited onboard computational and storage resources and high volatility of inter-satellite links (ISLs). Unlike existing satellite SFC studies that mainly focus on individual deployment objectives or simplified constraints, our method jointly optimizes service admission and load fairness while simultaneously considering VNF placement, physical-path mapping, resource capacity, stable-link availability, and loop-free routing constraints.

C. Machine learning for dynamic and adaptive orchestration

Machine learning (ML), particularly Deep Reinforcement Learning (DRL), has emerged as a powerful tool for managing network dynamism. Researchers employ reinforcement learning to optimize the trajectory of drone base stations for user coverage [3], and adopt DRL to optimize VNF placement in hybrid edge clouds [23], achieve robust orchestration [22], and maximize long-term network profit [27], [30]. Other ML approaches include federated learning to balance resource use [5] and graph attention networks for service mapping in LEO networks [31]. The study [42] proposes a scalable SFC orchestration scheme with federated reinforcement learning to train a global model for placing VNFs, improving resource utilization and scalability. Additionally, a loss-weight-based method is designed to yield reference samples in replay buffers for future training, thereby avoiding the strong relevance of samples, reducing placement errors and improving the resource utilization ratio. Similarly, the study [43] introduces a parallel VNF placement approach, which leverages federated deep reinforcement learning, exploiting VNF parallelism to significantly reduce end-to-end service latency. To improve the reliability of such distributed frameworks, the study [22] adopts a binary integer program to study the SFC orchestration problem, and designs a federated deep reinforcement learning orchestration algorithm that incorporates reputation theory to evaluate contributing models, enhancing the robustness of the training process. Shifting from reinforcement learning to predictive models, the study [44] employs a long short-term memory model to forecast node resource usage, thereby optimizing VNF placement in cloud-native environments. Despite their adaptability, ML-based approaches introduce significant new challenges for LEO deployments. The training process for DRL models is notoriously data-intensive and computationally expensive, making on-orbit training infeasible. Deploying a pre-trained model for real-time inference on a satellite's constrained hardware also presents a substantial overhead in terms of power consumption and processing capability.

To provide a clearer comparison of existing studies, Table I summarizes representative works according to several key features important for SFC/VNF orchestration. While existing literature provides a rich foundation, a distinct research gap emerges for VNF deployment specifically within LEO satellite networks. Exact methods are too computationally intensive for the dynamic LEO topology. Conventional heuristics are faster but are often ill-suited, as they do not sufficiently capture the severe onboard resource limitations and link volatility. Advanced ML methods face prohibitive hurdles related to on-orbit training and the high resource cost of inference. Our work addresses this challenge by proposing a hybrid metaheuristic algorithm to maximize service acceptance and ensure balanced resource utilization under stringent constraints of the LEO satellite environment.

III. Problem Formulation

Fig. 1 illustrates an SDN/NFV-enabled LEO satellite network management architecture including space network, terrestrial infrastructure, and user component. The space segment features LEO satellite constellations providing low-latency communication, supported by GEO satellites offering stable domain-level supervision for the dynamic LEO topology. The terrestrial segment works as a coordination center for service orchestration and network management, containing earth stations with gateway systems and SDN controllers. These implement NFV capabilities to optimize resource allocation and service pathways, enhancing operational flexibility and efficiency. VNFs provide software-based implementations of network functions that replace traditional dedicated hardware [25], [32], [33]. They can be dynamically deployed and managed in SFCs to meet diverse service requirements. VNFs leverage SDN and NFV to enhance flexibility and resource utilization in integrated networks.

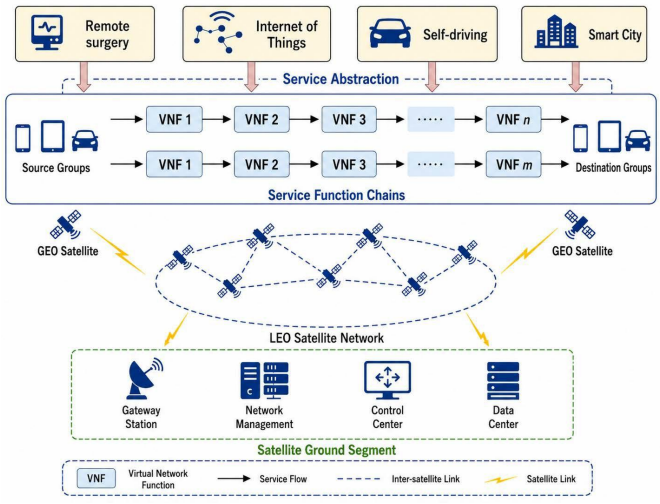

Fig. 1. LEO satellite network framework

A. Network Model

We model a generalized Walker Star LEO satellite constellation with N satellites uniformly distributed across M orbital

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

TABLE I
SUMMARY OF RELATED WORK ON SFC/VNF ORCHESTRATION.

Work	SAGIN/LEO	SFC/VNF	Service Admission	QoS/Delay	Resource Limit	Load Balance	Training-free
[1]	✓	✓	△	△	✓	×	✓
[25]	✓	✓	×	△	✓	×	✓
[5]	✓	✓	×	△	✓	✓	×
[27]	✓	✓	✓	✓	✓	×	×
[30]	✓	✓	×	△	✓	×	×
[31]	✓	✓	✓	△	✓	✓	△
[35]	×	✓	△	✓	✓	✓	✓
[36]	×	✓	✓	×	✓	×	✓
[39]	×	✓	×	✓	✓	×	✓
[42]	×	✓	×	×	✓	×	×
[43]	×	✓	×	✓	✓	✓	×
[22]	×	✓	×	✓	✓	×	×
This work	✓	✓	✓	✓	✓	✓	✓

✓: explicitly supported; △: partially or indirectly supported; ×: not explicitly considered.

planes. The physical network is represented as a graph $G = (\mathcal{F}, \mathcal{E})$, where $\mathcal{F}$ denotes a set of satellites and $\mathcal{E}$ represents inter-satellite links. Our framework implements a proximity-based access protocol where users connect to satellites with minimal line-of-sight distance. Each satellite contains four transceivers: two for intra-orbital and two for inter-orbital communications. The bandwidth between satellites n and m is denoted as $B_{n,m} \in B$. We assume circular orbits, with satellite positions expressed in Cartesian coordinates as:

$$\mathbf{p}_n = \begin{bmatrix} p_{x,n} \\ p_{y,n} \\ p_{z,n} \end{bmatrix} = (h + R_e)\boldsymbol{\rho}_n = (h + R_e) \begin{bmatrix} \rho_{x,n} \\ \rho_{y,n} \\ \rho_{z,n} \end{bmatrix}, \quad (1)$$

where $\mathbf{p}_n$ denotes the Cartesian position vector of satellite n, and $\boldsymbol{\rho}_n = [\rho_{x,n}, \rho_{y,n}, \rho_{z,n}]^T$ is the normalized direction vector, whose components are defined as:

$$\begin{aligned} \rho_{x,n} &= \cos(\xi + \tau)\cos\beta - \sin(\xi + \tau)\cos\alpha\sin\beta, \\ \rho_{y,n} &= \cos(\xi + \tau)\sin\beta + \sin(\xi + \tau)\cos\alpha\cos\beta, \\ \rho_{z,n} &= \sin(\xi + \tau)\sin\alpha. \end{aligned} \quad (2)$$

The position of the satellite n in the Cartesian reference frame is defined by coordinates $p_{x,n}, p_{y,n}, p_{z,n}$ as shown in (1). Key orbital parameters include: h (orbital altitude), R_e (Earth's radius), α (inclination angle), β (right ascension of ascending node), τ (true anomaly), and ξ (perigee argument). β establishes the primary orientation of the entire orbital plane by specifying its swivel or azimuthal rotation relative to a fixed inertial direction, measured in the Earth's equatorial plane to the point where the satellite crosses into the Northern Hemisphere. Once this plane is set, ξ orients the elliptical path within that plane by defining the angle from the ascending node to the orbit's point of closest approach. Finally, with the orbital path fully defined in space, τ pinpoints the satellite's exact instantaneous position along this ellipse, measured as an angle from the perigee to the satellite's current position vector. Thus, while τ and ξ define the static geometry of the orbit, τ is the dynamic element that describes the satellite's motion over time. The distance between satellites n and m is represented as a function $\delta(n, m)$, *i.e.*,

$$\delta(n,m) = \sqrt{(p_{x,n} - p_{x,m})^2 + (p_{y,n} - p_{y,m})^2 + (p_{z,n} - p_{z,m})^2} \quad (3)$$

The propagation delay between satellites, denoted as $d_{n,m}$, is calculated by taking the ratio of the inter-satellite separation distance for pair (n,m) to the electromagnetic wave propagation velocity (speed of light c), which is expressed as:

$$d_{n,m} = \frac{\delta(n,m)}{c} \ \forall (n,m) \in \mathcal{E}. \quad (4)$$

We concentrate on the satellites' computational processing and communication transmission capabilities. The operational burden on satellite n, characterized as its load factor and mathematically represented by l_n, is defined as:

$$l_n = \zeta_1 c_n + \zeta_2 b_n \quad (5)$$

where ζ_1 and ζ_2 are weighting coefficients that represent the relative importance of computational and communication resource utilization, respectively. c_n and b_n quantify the computational and communication resources utilized by satellite n during the operation.

We denote the comprehensive load profile across the entire LEO satellite constellation as vector l. To objectively assess load distribution equity within the satellite network infrastructure, we incorporate Jain's fairness index as the evaluation metric, which provides a quantitative assessment of resource allocation balance, is calculated as:

$$f_t = \frac{\left(\sum_{n \in \mathcal{F}} l_n\right)^2}{|\mathcal{F}| \cdot \sum_{n \in \mathcal{F}} l_n^2} \quad (6)$$

B. Service Modeling

The system implements SFC as its operational service paradigm, wherein VNFs are arranged sequentially in predetermined configurations. Effective service provisioning necessitates the sequential deployment of each VNF onto appropriate network nodes while satisfying users' end-to-end latency constraints. We assume that incoming service requests are aggregated prior to each orchestration decision interval, with a collective set of these services denoted as $\mathcal{Q} = \{q | q = 1, 2, \ldots, |\mathcal{Q}|\}$. The service taxonomy distinguishes between two principal categories: services primarily constrained by bandwidth requirements and those dominated by computational demands. We represent the origin and termination points

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

for these services as sets $\{s_q|q \in \mathcal{Q}\}$ and $\{d_q|q \in \mathcal{Q}\}$, respectively. For any particular service q, its associated VNF chain is formally defined as sequence $\mathrm{v}_q = \{v_i|i{=}1, 2, \ldots, |v_q|\}$, where component v_i signifies the i-th functional element and $|v_q|$ quantifies the chain's total function count. The inter-function traffic flows between sequential VNFs are characterized by an edge set $\mathcal{E}_q = \{(v_i, v_j)|v_i, v_j \in \mathrm{v}_q, q \in \mathcal{Q}\}$.

For VNF placement modeling, we use a binary variable $x_{v_i,n,q}$ to represent function-node mapping, where $x_{v_i,n,q} = 1$ when the i-th VNF of service q is deployed on node n. The placement configuration is denoted by $\mathbf{x} = \{x_{i,n,q}|v_i \in \mathrm{v}_q, n \in \mathcal{F}\}$. For link mapping, a binary variable $y^{(n,m)}_{(v_i,v_j),q}$ equals 1 when virtual link (v_i, v_j) of service q traverses physical link (n, m). The complete mapping is $\mathbf{y}{=}\{y^{(n,m)}_{(v_i,v_j),q}|\,(v_i, v_j) \in \mathcal{E}_q, (n, m) \in \mathcal{E}, q \in \mathcal{Q}\}$. Service admission is controlled by a binary variable z_q, where $z_q{=}1$ indicates service q is accepted. The admission vector is $\{z_q|q \in \mathcal{Q}\}$.

Meeting latency constraints is critical for users' QoS. While SFC performance includes both processing and transmission delays, we focus on communication delay as the primary factor in end-to-end latency, expressed as:

$$t_q = \sum_{(n,m)\in\mathcal{E}} \sum_{(v_i,v_j)\in\mathcal{E}_q} y^{(n,m)}_{(v_i,v_j),q} d_{n,m} \quad \forall q \in \mathcal{Q}. \tag{7}$$

C. Constraints of LEO

The network service deployment mechanism operates with predetermined specifications regarding the source node, destination node, and the ordered sequence of VNFs. To maintain topological integrity of the service chain, two placement constraints must be enforced: constraint C_1 ensures that the entry-point VNF is exclusively allocated to the designated source node, while constraint C_2 governs the terminal VNF's assignment to the specified destination node. These critical boundary conditions can be mathematically formulated as:

$$\begin{aligned} C_1 &: x_{v_1,s_q,q} = z_q \quad \forall q \in \mathcal{Q} \\ C_2 &: x_{v_{|q|},d_q,q} = z_q \quad \forall q \in \mathcal{Q} \end{aligned} \tag{8}$$

Constraints C_1 and C_2 impose a hard-anchoring policy, fixing the first and last VNFs of an SFC to the specified source and destination nodes, respectively. While this appears to limit placement flexibility, it is a deliberate design choice that reflects a significant class of real-world services, particularly in edge and access networks. This model is specifically tailored for scenarios involving ingress/egress processing, where the initial VNF (v_1) must perform a function like security filtering, traffic authentication, or protocol conversion at the point of traffic origin (s_q) before it enters the core network. Similarly, the terminal VNF ($v_{|q|}$) may handle functions like data formatting for the end-user or egress logging, which are most logically placed at the traffic's exit point (d_q). By enforcing these constraints, our model accurately represents these edge-native service requirements and focuses the optimization challenge on the complex problem of routing and placing the intermediate VNFs across the satellite constellation. This simplification also makes the optimization more tractable by significantly reducing the search space for endpoint placement.

To maintain the structural integrity of SFCs, our model enforces a placement singularity constraint whereby each component VNF within an admitted service request must be allocated to precisely one physical node in the network infrastructure. This fundamental mapping restriction can be mathematically given as:

$$C_3 : \sum_{n\in\mathcal{F}} x_{v_i,n,q} = z_q \quad \forall v_i \in \mathrm{v}_q \quad \forall q \in \mathcal{Q} \tag{9}$$

Furthermore, to guarantee service performance objectives, each successfully deployed service chain must satisfy stringent temporal requirements. Specifically, the cumulative transmission latency experienced along the service path must not exceed the service-specific deadline parameter T_q. This critical QoS constraint is formally characterized by:

$$C_4 : t_q \le T_q \quad \forall q \in \mathcal{Q} \tag{10}$$

Within the SFC architecture, which organizes VNFs in predetermined processing sequences connected by directional data flows, adherence to flow conservation principles becomes essential for maintaining functional integrity. This network flow equilibrium requires that for each intermediate node in the service path, the aggregate incoming traffic must precisely balance the aggregate outgoing one, which can be mathematically given as:

$$\begin{aligned} C_5 &: \sum_{m\in\mathcal{F}} y^{(n,m)}_{(v_i,v_j),q} - \sum_{m\in\mathcal{F}} y^{(m,n)}_{(v_i,v_j),q} = x_{v_i,n,q} - x_{v_j,n,q} \\ &\forall\,(v_i, v_j) \in \mathcal{E}_q \quad \forall n \in \mathcal{F} \quad \forall q \in \mathcal{Q}. \end{aligned} \tag{11}$$

Regarding infrastructure capacity limitations, the distributed satellite network operates under strict resource constraints. Each satellite node possesses finite computational and communication capabilities that must not be exceeded during service provisioning. These resource utilization boundaries are governed by:

$$\begin{aligned} C_6 &: \sum_{q\in\mathcal{Q}} \sum_{v_i\in\mathbf{v}_q} x_{v_i,n,q} c_{v_i,q} \le C_n \quad \forall n \in \mathcal{F} \\ C_7 &: \sum_{q\in\mathcal{Q}} \sum_{(v_i,v_j)\in\mathcal{E}_q} y^{(n,m)}_{(v_i,v_j),q} b_q \le B_{n,m} \quad \forall (n, m) \in \mathcal{E} \end{aligned} \tag{12}$$

where $c_{v_i,q}$ quantifies the computational resource consumption of the i-th VNF belonging to service q, while b_q represents the communication resource requirement for service q. The parameter $C_n \in \mathbf{C}$ denotes the maximum computational processing capacity available at satellite node n, with $\mathbf{C}$ encompassing a complete set of computational capacities across the constellation. Constraint C_6 enforces adherence to computational resource limitations, while constraint C_7 ensures that bandwidth allocation remains within the physical transmission capabilities of the network links.

To partially reflect topology dynamics of LEO constellations within an orchestration window, we introduce a link-stability parameter derived from time-varying ISL availability. Instead of performing rolling-horizon re-optimization, the current model ensures that the selected physical links remain available during the considered service window [45], [46]. This model

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

operates over a defined service lifetime horizon, denoted by the interval [0, T]. A decision to admit and deploy an SFC is made at t=0, under the critical condition that the assigned resources and the entire service path remain topologically valid throughout this interval.

To formalize this, we first introduce a predetermined and time-varying parameter, $a_{nm}(t) \in \{0,1\}$, which indicates whether the physical Inter-Satellite Link (ISL) between satellite nodes n and m is available at a given time t. $a_{nm}(t)$ is derived from the constellation's ephemeris data. Based on this, we define a stable physical link parameter, L_{nm}, which holds true only if the link is continuously available for the entire service duration.

$$L_{nm} = \begin{cases} 1 & \text{if } a_{nm}(t) = 1, \quad \forall t \in [0,T] \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

D. SFC Orchestration Problem

The optimization framework for SFC deployment targets dual objectives: maximizing the service admission ratio while simultaneously achieving equitable resource utilization across the satellite constellation. This multi-objective optimization challenge is mathematically formalized as a binary nonlinear program (BNLP) with constraints C_1–C_{10}. The final optimization problem is formulated as:

$$\begin{aligned} P: \max_{\mathbf{x},\mathbf{y},\mathbf{z}} \quad & \sum_{q\in\mathcal{Q}} z_q + \gamma_1 \frac{\left(\sum_{n\in\mathcal{F}} l_n\right)^2}{|\mathcal{F}| \cdot \sum_{n\in\mathcal{F}} l_n^2} \\ \text{s.t.} \quad & C_1 - C_7 \\ & C_8 : \mathbf{x},\mathbf{y},\mathbf{z} \in \{0,1\} \\ & C_9 : y^{(n,m)}_{(v_i,v_{i+1}),q} \le L_{nm}, \\ & \forall q \in \mathcal{Q}, (v_i, v_{i+1}) \in E_q, (n,m) \in E \end{aligned} \tag{14}$$

The coefficient γ_1 remains a tunable weighting parameter to balance the two objectives. The coefficient γ_1 is a dimensionless preference weight in the weighted-sum scalarization [47]. Since accepting one additional service increases the first term by one, the increase of δ_J in Jain's index contributes $\gamma_1\delta_J$ units of admission-equivalent utility. For two feasible deployments A and B, the preferred solution changes when $\gamma_1 = (\sum_q z_q^A - \sum_q z_q^B)/(J(l^B) - J(l^A))$, such that the denominator is nonzero. Thus, larger γ_1 emphasizes load balancing and tends to spread VNFs more evenly, whereas smaller γ_1 emphasizes service admission and may permit more concentrated resource utilization. We select γ_1 according to the intended operating preference and preliminary sensitivity tests. C_9 is the stable routing constraint. It explicitly forbids the routing variable $y_{nm}^{v_i v_{i+1},q}$ from taking a value of 1, if that link is not stable ($L_{nm} = 0$). This rigorously enforces service continuity and makes the model robust against the inherent dynamics of the LEO network, directly addressing the limitations of a purely static formulation. For a node subset $S \subseteq F$, $E(S) = \{(n,m) \in E | n \in S, m \in S\}$. To guarantee that each virtual link is mapped onto a single loop-free physical path and that a rejected service occupies no physical-link resources, we introduce the following routing constraints, *i.e.*,

$$\text{C10a}: \quad \sum_{m:(n,m)\in E} y^{(n,m)}_{(v_i,v_{i+1}),q} \le z_q, \quad \forall n \in F, \tag{15a}$$

$$\text{C10b}: \quad \sum_{m:(m,n)\in E} y^{(m,n)}_{(v_i,v_{i+1}),q} \le z_q, \quad \forall n \in F, \tag{15b}$$

$$\text{C10c}: \quad \sum_{(n,m)\in E(S)} y^{(n,m)}_{(v_i,v_{i+1}),q} \le (|S|-1) z_q, \quad \forall S \subseteq F, \quad |S| \ge 2. \tag{15c}$$

Constraints C10a and C10b prevent branching by limiting the incoming and outgoing degree of each node for each virtual link. Constraint C10c eliminates directed subtours. Together with the flow-conservation constraint C5, these constraints guarantee that each accepted virtual link is mapped onto one simple physical path. When z_q=0, C3 and C10 require all placement and routing variables associated with service q to zero. Therefore, a rejected service does not consume computing or physical-link resources.

Although the problem is formulated as a maximization problem, the implementation does not directly minimize the negative value of (14). Instead, GSPSO evaluates each candidate solution with a nonnegative penalized fitness cost, *i.e.*,

$$\begin{aligned} F(\boldsymbol{\chi}) &= P(\boldsymbol{\chi}) + L_{\mathrm{s}}(\boldsymbol{\chi}) + L_{\mathrm{b}}(\boldsymbol{\chi}) \\ &= P(\boldsymbol{\chi}) + 100\left(|\mathcal{Q}| - \sum_{q\in\mathcal{Q}} z_q\right) + 50\left[1 - J(\boldsymbol{l})\right], \end{aligned} \tag{16}$$

where $P(\boldsymbol{\chi})$ denotes the constraint-violation penalty. It is nonnegative and becomes positive only when the candidate solution violates one or more constraints. The service-rejection loss is defined as:

$$L_{\mathrm{s}}(\boldsymbol{\chi}) = 100\left(|\mathcal{Q}| - \sum_{q\in\mathcal{Q}} z_q\right), \tag{17}$$

which is positive whenever not all service requests are accepted. The load-imbalance loss is defined as:

$$L_{\mathrm{b}}(\boldsymbol{\chi}) = 50\left[1 - J(\boldsymbol{l})\right], \tag{18}$$

which is positive whenever the Jain fairness index is smaller than one.

Let $U(\boldsymbol{\chi})$ denote the original objective defined in (14). Since γ_1=0.5 in the experiments, the implemented fitness can be expressed as:

$$\begin{aligned} F(\boldsymbol{\chi}) &= P(\boldsymbol{\chi}) + 100|\mathcal{Q}| + 50 - 100\sum_{q\in\mathcal{Q}} z_q - 50J(\boldsymbol{l}) \\ &= P(\boldsymbol{\chi}) + 100\left[|\mathcal{Q}| + 0.5 - U(\boldsymbol{\chi})\right]. \end{aligned} \tag{19}$$

For any feasible solution, $P(\boldsymbol{\chi}) = 0$. Therefore, $F(\boldsymbol{\chi})$ differs from $-100U(\boldsymbol{\chi})$ only by the fixed additive constant $100|\mathcal{Q}| + 50$. Consequently, minimizing $F(\boldsymbol{\chi})$ is equivalent to maximizing the original objective defined in (14).

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

All the solutions reported in the experimental results are subjected to constraint verification after decoding and are confirmed to satisfy constraints C1–C10. Thus, the constraint-violation penalty is 0, *i.e.*, $P(\boldsymbol{\chi}) = 0$ for each reported solution. It is emphasized that a positive fitness value does not indicate the existence of constraint violations. For a feasible solution, the implemented fitness reduces to (20), *i.e.*,

$$F(\boldsymbol{\chi}) = 100\left(|\mathcal{Q}| - \sum_{q\in\mathcal{Q}} z_q\right) + 50\left[1 - J(\boldsymbol{l})\right], \tag{20}$$

where the first term remains positive whenever not all service requests are accepted, while the second term remains positive whenever the Jain fairness index is smaller than one. Hence, even when all constraints are satisfied, the fitness value is generally positive unless all service requests are accepted and perfectly balanced resource utilization is achieved.

IV. Genetic Simulated annealing-based Particle Swarm Optimization (GSPSO)

To address the BNLP problem, this work proposes an algorithm called GSPSO. Various optimization methods exist for solving BNLP problems, but each has its limitations. For example, genetic algorithms (GA) exhibit high individual diversity, while simulated annealing (SA) demonstrates strong global search capabilities. However, both algorithms suffer from slow convergence. Particle swarm optimization (PSO), on the other hand, is known for its fast convergence but tends to get trapped into local optima. To overcome these limitations, GSPSO integrates the advantages of these methods. Specifically, it combines the Metropolis acceptance rule from SA and genetic operations from GA to enhance the search efficiency of PSO. Unlike traditional PSO, each particle in GSPSO has a unique superior particle that stands out within the population. This design helps improve the algorithm's ability to explore the solution space while maintaining rapid convergence. Each particle i has a corresponding superior particle, whose position is denoted as $\dot{\chi}_i$. Within the population, there is a globally optimal position $\hat{chi}$. $\dot{\chi}_i$ is given as:

$$\dot{\chi}_i = \frac{c_1\theta_1\check{\chi}_i + c_2\theta_2\hat{\chi}}{c_1\theta_1 + c_2\theta_2} \tag{21}$$

where c_1 and c_2 are parameters that reflect the influence of $\check{\chi}_i$ and $\hat{\chi}$, respectively. Meanwhile, θ_1 and θ_2 represent two random constants that are uniformly generated in (0, 1).

GSPSO updates each particle i' velocity v_i and position $\chi_i^{\varsigma+1}$ as:

$$\begin{aligned} v_i &= \omega\cdot v_i + c\cdot\mu\cdot(\dot{\chi}_i - \chi_i^{\varsigma}) \\ \chi_i^{\varsigma+1} &= \chi_i^{\varsigma} + v_i \end{aligned} \tag{22}$$

After binary discretization, the resulting particle is processed by a feasibility-preserving decoder. Service requests are decoded sequentially. For each admitted request, the decoder verifies endpoint placement, unique VNF placement, propagation delay, computing capacity, residual link bandwidth, stable-link availability, flow conservation, and loop-free routing.

When mapping a virtual link, the decoder only selects physical links with sufficient residual bandwidth and excludes previously visited nodes. The required bandwidth is temporarily reserved after each successful physical-link mapping.

If any VNF or virtual link of a service request cannot be mapped without violating C1–C10, the entire request is rejected by setting z_q=0. All placement and routing variables associated with this request are reset to zero, and all temporarily reserved computing and bandwidth resources are released. Therefore, each deployment solution returned by the decoder satisfies C1–C10.

$$\bar{\chi}_{i,d}^{\varsigma+1} = \begin{cases} 1, & \mathrm{clip}_{[0,1]}\left(\chi_{i,d}^{\varsigma+1}\right) \geq \tau_b, \\ 0, & \mathrm{clip}_{[0,1]}\left(\chi_{i,d}^{\varsigma+1}\right) < \tau_b, \end{cases} \qquad \tau_b = 0.5, \tag{23}$$

Here, μ denotes a vector of uniformly distributed random numbers in (0,1). The particle position represents a D-dimensional encoding of the binary decision variables x, y, and z. (22) is evaluated in an auxiliary continuous search space. After the position update, each coordinate is clipped to [0,1] and discretized according to (23). The midpoint threshold τ_b of 0.5 (τ_b=0.5) treats two binary states symmetrically and introduces no prior bias toward either selecting or rejecting a decision bit. A smaller threshold produces denser binary vectors and may increase the number of initially selected placements or links, but it also increases the likelihood of resource conflicts. Conversely, a larger threshold produces sparser solutions and may reduce resource consumption at the cost of rejecting more services. The threshold operation may discard small differences between continuous particle positions and can lead to premature fixation of some decision bits. GSPSO alleviates this effect by retaining the continuous position and velocity information across iterations, while crossover, fixed-probability mutation, and the Metropolis acceptance rule preserve search diversity. Therefore, the adopted strategy provides a simple and stable interface between continuous particle updating and binary SFC deployment decisions.

$\chi_i^{\varsigma+1}$ is chosen when $f\left(\chi_i^{\varsigma+1}\right) \leq f\left(\chi_i^{\varsigma}\right)$; otherwise, it is chosen if:

$$e^{\left(\frac{f(\chi_i^{\varsigma}) - f\left(\chi_i^{\varsigma+1}\right)}{T_{\varsigma}}\right)} > \xi \tag{24}$$

where T_{ς} represents the temperature at iteration ς, and ξ is a vector of uniformly distributed random numbers in (0,1).

Fig. 2 outlines critical steps of GSPSO, and Algorithm 1 presents its pseudo codes. GSPSO lies in its computational efficiency and suitability for on-orbit deployment, especially when compared to contemporary deep learning approaches like DRL or Graph Neural Networks (GNNs). The computational complexity of GSPSO is determined by the number of iterations, the population size, and the cost of the fitness evaluation for each particle. The fitness evaluation, the most intensive step, involves assessing VNF placements and routing paths for $|Q|$ service requests across $|\mathcal{F}|$ satellite nodes, leading to a polynomial time complexity on the order of $O(I{\cdot}P{\cdot}f(|Q|,|\mathcal{F}|))$, where f represents the cost of evaluating one solution.

In addition, DRL and GNN methodologies are characterized by two distinct and demanding phases: offline training and online inference. The offline training phase is exceptionally

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

resource-intensive, often requiring millions of simulation steps and powerful GPU hardware for backpropagation through deep neural networks. This makes it not only costly but also impractical for the rapidly changing LEO network topology, where a trained model can quickly become stale. While the online inference of a pre-trained model might seem fast, it still involves a forward pass through multiple layers of a neural network, entailing numerous matrix-vector multiplications that can impose a significant computational and energy burden on the resource-constrained processing units aboard a satellite.

GSPSO, however, bypasses the need for a training phase entirely. Its iterative refinement process, while computationally non-trivial, consists of fundamentally simpler arithmetic and logical operations compared to the complex calculations within a deep neural network. Thus, GSPSO strikes a more pragmatic balance: it offers a sophisticated and adaptive search capability without the prohibitive offline training requirements and high online inference overhead of deep learning models, rendering it a more feasible and resource-efficient solution for real-time decision-making in the LEO satellite environment.

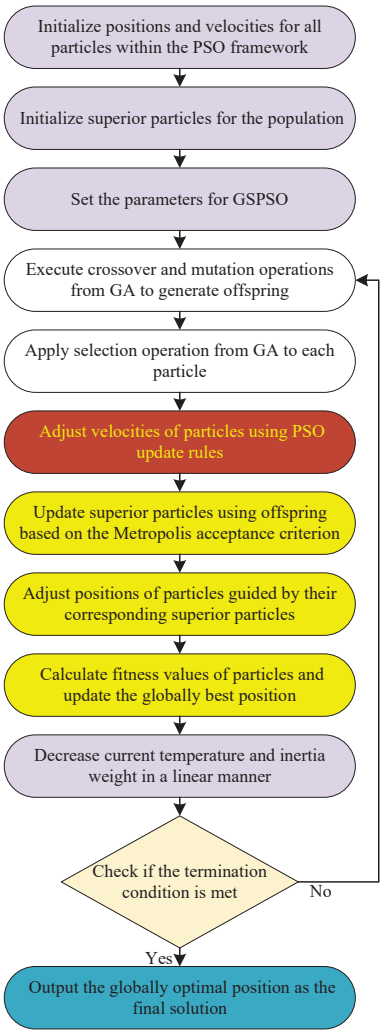

Fig. 2. The process of GSPSO

For compactness, let $\mathcal{D}(\cdot)$ denote the complete binary mapping, decoding, repair, and feasibility-validation procedure. If an SFC request cannot be feasibly repaired, it is rejected as a whole. Therefore, the solution returned by $\mathcal{D}(\cdot)$ satisfies constraints C1–C10.

Algorithm 1 Genetic Simulated Annealing-Based Particle Swarm Optimization (GSPSO)

Require: Model in (14); $\mathcal{G}$, $\mathcal{Q}$, N_{pop}, $T_{\max}$, c_1, c_2, p_m, T_0, c, $\hat{\omega}$, and $\check{\omega}$.
Ensure: Best feasible solution $\boldsymbol{\chi}^*$, fitness $F(\boldsymbol{\chi}^*)$, and objective $U(\boldsymbol{\chi}^*)$.
Initialize particle positions $\{\boldsymbol{s}_i^0\}_{i=1}^{N_{\text{pop}}}$, velocities $\{\boldsymbol{v}_i^0\}_{i=1}^{N_{\text{pop}}}$, and temperature $T \leftarrow T_0$.
for $i = 1$ to N_{pop} **do**
  $\boldsymbol{\chi}_i^0 \leftarrow \mathcal{D}(\boldsymbol{s}_i^0)$ and $F_i^0 \leftarrow F(\boldsymbol{\chi}_i^0)$ using (16).
  $\boldsymbol{p}_i \leftarrow \boldsymbol{s}_i^0$ and $F_{p,i} \leftarrow F_i^0$.
end for
$k \leftarrow \arg\min_i F_{p,i}$ and $\boldsymbol{g} \leftarrow \boldsymbol{p}_k$.
for $t = 1$ to $T_{\max}$ **do**
  $\omega_t \leftarrow \hat{\omega} - \frac{t}{T_{\max}}(\hat{\omega} - \check{\omega})$.
  for $i = 1$ to N_{pop} **do**
    Select particle $j \neq i$ and generate $\tilde{\boldsymbol{s}}_i^t$ by crossover and mutation with the fixed probability p_m.
    Generate $r_1, r_2 \sim \mathcal{U}(0,1)$.
    $\boldsymbol{v}_i^t \leftarrow \omega_t \boldsymbol{v}_i^{t-1} + c_1 r_1(\boldsymbol{p}_i - \tilde{\boldsymbol{s}}_i^t) + c_2 r_2(\boldsymbol{g} - \tilde{\boldsymbol{s}}_i^t)$.
    $\boldsymbol{s}_{i,\text{cand}}^t \leftarrow \tilde{\boldsymbol{s}}_i^t + \boldsymbol{v}_i^t$.
    $\boldsymbol{\chi}_{i,\text{cand}}^t \leftarrow \mathcal{D}(\boldsymbol{s}_{i,\text{cand}}^t)$ and $F_{i,\text{cand}}^t \leftarrow F(\boldsymbol{\chi}_{i,\text{cand}}^t)$.
    $\Delta F \leftarrow F_{i,\text{cand}}^t - F_i^{t-1}$.
    if $\Delta F \leq 0$ or $\text{rand}(0,1) < \exp(-\Delta F/T)$ **then**
      Accept the candidate: $\boldsymbol{s}_i^t \leftarrow \boldsymbol{s}_{i,\text{cand}}^t$, $\boldsymbol{\chi}_i^t \leftarrow \boldsymbol{\chi}_{i,\text{cand}}^t$, and $F_i^t \leftarrow F_{i,\text{cand}}^t$.
    else
      Retain $\boldsymbol{s}_i^t \leftarrow \boldsymbol{s}_i^{t-1}$, $\boldsymbol{\chi}_i^t \leftarrow \boldsymbol{\chi}_i^{t-1}$, and $F_i^t \leftarrow F_i^{t-1}$.
    end if
    if $F_i^t < F_{p,i}$ **then**
      $\boldsymbol{p}_i \leftarrow \boldsymbol{s}_i^t$ and $F_{p,i} \leftarrow F_i^t$.
    end if
  end for
  $k \leftarrow \arg\min_i F_{p,i}$, $\boldsymbol{g} \leftarrow \boldsymbol{p}_k$, and $T \leftarrow cT$.
end for
$\boldsymbol{\chi}^* \leftarrow \mathcal{D}(\boldsymbol{g})$.
Calculate $U(\boldsymbol{\chi}^*)$ according to (14).
return $\boldsymbol{\chi}^*$, $F(\boldsymbol{\chi}^*)$, and $U(\boldsymbol{\chi}^*)$.

V. PERFORMANCE EVALUATION

A. Parameter setting

This work conducts several simulation experiments to validate the feasibility of the proposed GSPSO and evaluates its performance by comparing it against several established algorithms. Table II shows the primary parameter settings. The detailed parameters of the SFC deployment optimization problem are shown in Table III. The hyperparameters in Table II are selected based on preliminary experiments and the roles of different search operators in GSPSO. The mutation probability is fixed at 0.05 (p_m=0.05) throughout the optimization process. This value maintains population diversity without excessively disrupting promising solutions. Unlike the inertia weight, the mutation probability is not dynamically adjusted during an individual run. The acceleration coefficients are chosen to balance the influence of individual experience and population guidance, while the velocity coefficient controls the step size of particle movement. The initial temperature is set to a relatively large value such that inferior solutions can be accepted with a certain probability in the early search stage, which helps the algorithm escape from local optima. The inertia weight gradually decreases from a large value to a

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

small one, enabling GSPSO to emphasize global exploration in early iterations and local exploitation in the later iterations. To further evaluate the robustness of these parameter choices, a sensitivity analysis on population size and mutation probability is conducted in Section V-B. To further demonstrate the performance of the proposed algorithm, we consider larger LEO satellite network scales. Specifically, the number of satellites is extended to 30 to examine the scalability of GSPSO as the deployment dimension and the number of candidate VNF mappings increase. It is worth noting that the satellite number is very large in reality because for example, Australia owns and operates 7 LEO satellites in space[1] and Italy currently has 16 LEO satellites in total[2]. In addition, we vary the number of SFC requests, SFC length, link capacity ratio, delay bound ratio, and computing resource ratio to evaluate the robustness of the proposed algorithm under different service densities, chain complexities, network capacities, delay constraints, and resource conditions.

TABLE II
PARAMETER SETTING OF GSPSO.

P	p_m	$c_1(c_2)$	c	T_0	ϖ	$\hat{\omega}$	$\check{\omega}$
40	0.05	1.5	1.8	10000	0.98	0.9	0.2

TABLE III
PARAMETER SETTING OF SFC DEPLOYMENT OPTIMIZATION PROBLEM

Parameter	Value
Number of satellites in constellation (N)	5–30
Orbital altitude of satellites (h)	2,000 km
Earth radius (R_e)	6,371 km
Processing capacity per satellite node (C_n)	1,000–1,500 CPU unit
Bandwidth requirement per service flow (b_q)	10–60 MHz
Computational demand of individual VNF ($c_{v_i,q}$)	25–150 CPU unit
Link capacity between satellite pairs ($B_{n,m}$)	500 MHz
Electromagnetic propagation constant (c)	2.99×10^8 m/s

B. Experimental Results

This work first evaluates its convergence behavior across different scenarios. GSPSO is compared with several baseline algorithms, including three well-known general-purpose meta-heuristic algorithms, *i.e.*, PSO [48], Self-Adaptive PSO (SAPSO) [49], and Tyrannosaurus Optimization Algorithm (TROA) [50], as well as a problem-specific two-stage heuristic (TH) algorithm [51]. Moreover, we add FAGD-Lite as another problem-specific lightweight baseline [52]. FAGD-Lite follows a fast adaptive greedy deployment strategy. For each incoming SFC request, it sequentially places VNFs on feasible satellite nodes according to residual computing resources, link feasibility, and incremental deployment cost. Compared with population-based meta-heuristic algorithms, FAGD-Lite can rapidly construct feasible solutions with low search complexity. However, it lacks global exploration and stochastic escaping mechanisms, and may suffer from locally favorable but globally suboptimal VNF placement decisions. The TH baseline is designed for SFC deployment and network resource allocation. It first sorts service requests according to their resource demands and delay requirements, and greedily places VNFs on feasible satellite nodes while considering residual computing resources and propagation distances. To ensure the reliability of experimental results, each algorithm undergoes 90 independent runs.

TABLE IV
GSPSO FITNESS STATISTICS FOR DIFFERENT NUMBERS OF SFC REQUESTS.

Q	Best	Mean	Worst	Std.
5	0.042021	0.145675	0.244554	0.051454
10	0.000832	0.057652	0.141314	0.032909
15	0.010780	0.043910	0.175788	0.042969
20	0.001136	0.015797	0.052256	0.013174

TABLE V
GSPSO FITNESS STATISTICS FOR DIFFERENT NUMBERS OF SATELLITES.

N	Best	Mean	Worst	Std.
5	0.000038	0.001669	0.006599	0.001317
10	0.009816	0.031436	0.055389	0.012566
15	0.074572	0.146797	0.247825	0.042974
20	0.327580	0.559762	0.926727	0.145671

Table IV summarizes the final fitness statistics of GSPSO under different numbers of SFC requests while the number of satellites is fixed at 10, *i.e.*, $N = 10$. The number of SFC requests, Q, is varied from 5 to 20, and each setting is evaluated over 30 independent runs. Since a lower fitness value indicates a better solution, the decrease in the mean fitness from 0.145675 when $Q = 5$ to 0.015797 when $Q = 20$ demonstrates that the average final solution quality does not deteriorate as the number of requests increases under the tested settings. In particular, the setting of $Q = 20$ obtains the lowest mean fitness, worst-case fitness, and standard deviation, with values of 0.015797, 0.052256, and 0.013174, respectively. The lowest best-case fitness is 0.000832 when $Q = 10$, followed by 0.001136 when $Q = 20$. These results indicate that GSPSO maintains stable final optimization performance.

Table V presents the final fitness statistics of GSPSO for different numbers of satellites while the number of SFC requests is fixed at $Q = 10$. The number of satellites, N, varies from 5 to 20, with 30 independent runs for each setting. The mean fitness increases from 0.001669 at $N = 5$ to 0.559762 at $N = 20$. Meanwhile, the best fitness increases from 0.000038 to 0.327580, and the worst fitness increases from 0.006599 to 0.926727. The standard deviation also rises from 0.001317 to 0.145671 as the number of satellites increases. These results show that larger instances are associated with higher and more dispersed final fitness values, reflecting the increased scale and variability of the corresponding optimization problem.

Fig. 3 compares the overall performance of GSPSO with PSO, SAPSO, SA, TROA, TH, and FAGD-Lite, whereas Fig. 4 presents iteration-wise convergence trajectories of five stochastic iterative algorithms. All results are obtained under the same benchmark setting over 90 independent runs. GSPSO and PSO both achieve a service acceptance rate of 1.000.

[1]https://www.unsw.edu.au/canberra/our-research/research-centres-institutes/unsw-canberra-space/our-research/australian-owned-satellite-database/quick-stats-and-graphs

[2]https://nanoavionics.com/blog/how-many-satellites-are-in-space/

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

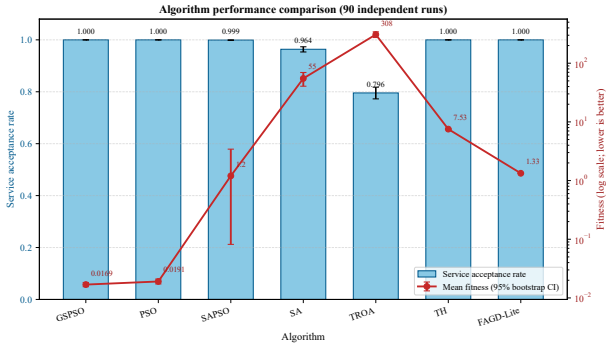

Fig. 3. Performance comparison of different algorithms.

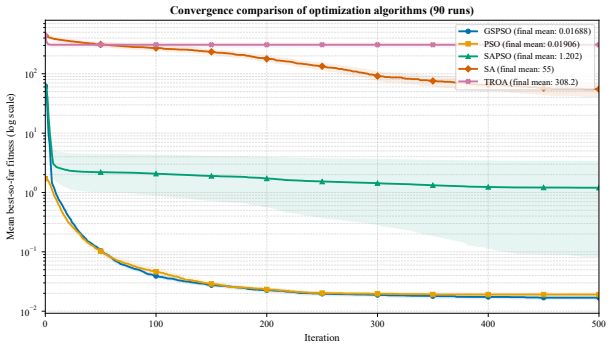

Fig. 4. Convergence comparison of stochastic iterative algorithms.

However, GSPSO obtains a lower mean fitness value of 0.016883, representing an 11.4% reduction relative to the 0.019061 obtained by PSO. The convergence curves also show that GSPSO and PSO rapidly enter the low-fitness region, with GSPSO ultimately reaching the lower final mean value.

Table VI summarizes the final fitness statistics. GSPSO achieves the lowest best-case, mean, and worst-case fitness values of 0.000598, 0.016883, and 0.038638, respectively. Its standard deviation of 0.006850 is also the lowest among the stochastic algorithms. Compared with PSO, SA, SAPSO, TROA, TH, and FAGD-Lite, GSPSO reduces the mean fitness by 11.4%, 99.97%, 98.60%, 99.99%, 99.78%, and 98.73%, respectively. Although deterministic TH and FAGD-Lite baselines exhibit approximately zero variance, their mean fitness values are considerably higher. These results demonstrate that GSPSO provides a favorable combination of solution quality and stability across repeated runs.

GSPSO achieves the lowest fitness value among metaheuristic algorithms and maintains a high service acceptance rate. The improvement is mainly attributed to the complementary roles of its three components. Genetic operations enhance population diversity, simulated annealing helps escape local optima, and particle swarm updating accelerates convergence toward promising regions. In contrast, PSO and SAPSO are more likely to converge around suboptimal deployment patterns due to insufficient exploration diversity, while SA converges more slowly because it lacks collective learning from the population. TROA is less suitable for the discrete and highly constrained SFC deployment problem, where small continuous perturbations may not lead to meaningful changes in VNF placement or service admission.

TABLE VI
COMPARISON OF FINAL FITNESS VALUES.

Algorithm	Best	Mean	Worst	Std.
GSPSO	**0.000598**	**0.016883**	**0.038638**	0.006850
PSO	0.000647	0.019061	0.045230	0.008836
SA	0.087497	54.996780	300.453670	70.565310
SAPSO	0.014980	1.202331	100.033900	10.535020
TROA	0.124151	308.198700	703.488067	167.200300
TH	7.531341	7.531341	7.531341	0.000000
FAGD-Lite	1.331265	1.331265	1.331265	0.000000

TABLE VII
ABLATION STUDY OF DIFFERENT ALGORITHMIC COMPONENTS.

Algorithm	Avg. Fitness	Best Fitness	Avg. Service Acceptance Rate	Avg. Jain Index
GA-PSO	0.69	0.16	1.00	0.9862
GSPSO	**0.66**	**0.12**	**1.00**	**0.9868**
SA-PSO	1.33	1.03	1.00	0.9734

Table VII presents the ablation results of GSPSO. All three variants achieve full service acceptance, indicating their differences are mainly reflected in optimization quality and load balancing. GSPSO obtains the lowest average fitness of 0.66 and the lowest best fitness of 0.12. It achieves the highest average Jain index among three variants, demonstrating the complementary effects of genetic operators, simulated annealing mechanism, and particle swarm updating strategy.

Table IX compares the load balancing performance and running time of different algorithms over 90 independent runs. GSPSO achieves the highest average Jain index of 0.999662 and the lowest Jain-index standard deviation of 0.000137 among stochastic algorithms, indicating consistently balanced resource utilization. PSO obtains a comparable average Jain index of 0.999619 but requires a longer average running time of 273.9307 s, compared with 14.3243 s for GSPSO. Although SA, TROA, TH, and FAGD-Lite require less execution time, their average Jain indices are lower. These results indicate that GSPSO provides a favorable trade-off between load balancing quality, stability, and computational cost.

TABLE VIII
SCALABILITY EVALUATION OF GSPSO UNDER DIFFERENT NUMBERS OF SATELLITES.

Sat. Num.	Best Fit.	Avg. Fit.	Acc. Ratio	Jain Index
12	0.017993	0.042534	1.0000	0.999149
15	0.055438	0.096872	1.0000	0.998063
21	0.131127	0.229206	1.0000	0.995416
24	0.153827	0.289640	1.0000	0.994207
30	0.754855	1.440978	1.0000	0.971180

TABLE IX
COMPARISON OF JAIN INDEX AND RUNNING TIME OVER 90 INDEPENDENT RUNS.

Algorithm	Avg. Jain Index	Std. Jain Index	Avg. Time (s)	Std. Time (s)
GSPSO	**0.999662**	0.000137	14.3243	4.3377
PSO	0.999619	0.000177	273.9307	89.7925
SA	0.988953	0.008420	1.0389	0.1066
SAPSO	0.998176	0.001294	163.5470	21.5896
TROA	0.969360	0.021845	9.9693	1.5690
TH	0.849373	0.000000	0.0052	0.0021
FAGD-Lite	0.973375	0.000000	0.0234	0.0055

Table X evaluates the effects of population size and mutation probability on GSPSO over 30 independent runs. The mutation probability is fixed throughout each individual run, and all

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

evaluated configurations achieve a service acceptance rate of 1.0000. Increasing the population size generally increases the running time, whereas its effect on solution quality depends on the mutation probability. For population sizes of 20 and 80, the lowest mean fitness is yielded within the corresponding population group when $p_m = 0$. In contrast, when the population size is 40, the case with $p_m = 0.05$ achieves a substantially lower mean fitness than that with $p_m = 0$ and obtains the highest mean Jain index. Mutation probabilities of 0.20 and 0.50 consistently result in poorer fitness and load balancing performance. Among all tested configurations, a population size of 40 and a mutation probability of 0.05 achieve the lowest mean fitness of 0.016883 and the highest mean Jain index of 0.999662 while maintaining a moderate running time. Therefore, this configuration is adopted in the main experiments.

C. Scalability and Comprehensive Sensitivity Analysis

To evaluate the scalability of GSPSO, we increase the LEO network size from 12 to 30 satellites, with each setting independently evaluated over 30 runs. As shown in Table VIII, GSPSO maintains a service acceptance ratio of 1.0000 across all evaluated network scales. However, the average fitness increases from 0.042534 with 12 satellites to 1.440978 with 30 satellites, while the Jain index decreases moderately from 0.999149 to 0.971180. This trend indicates that increasing the number of satellites enlarges the placement search space and introduces more candidate VNF mapping combinations, making balanced resource allocation more difficult. Nevertheless, GSPSO continues to admit all service requests and retains a Jain index above 0.97 even in the 30-satellite scenario, demonstrating reasonable scalability for the evaluated small- and medium-scale constellations. We further evaluate GSPSO under five groups of LEO/SFC orchestration settings, including request density, SFC length, link capacity ratio, delay bound ratio, and computing resource ratio. The results are summarized in Table XI. Because these scenario groups involve different numbers of requests and different resource or QoS constraints, their absolute fitness values are interpreted within each group rather than directly compared across groups.

GSPSO maintains full service acceptance under all request-density, SFC-length, and computing-resource settings, with the Jain index remaining above 0.9955. Under stringent link-capacity settings of 0.05 and 0.08, the acceptance ratio decreases to 0.6800. When the link-capacity ratio reaches 0.10, full service acceptance is restored and the Jain index increases to 0.996051. The delay constraint has a more pronounced effect. As the delay bound increases from 5×10^{-5} to 9×10^{-5}, the acceptance ratio improves from 0.2000 to 0.6000 and the Jain index increases from 0.222221 to 0.650435. When the delay bound reaches 1.1×10^{-4}, GSPSO again achieves full service acceptance. These results show that GSPSO responds predictably to resource and QoS constraints and recovers high service acceptance and load fairness as the feasible deployment space becomes less restrictive. Overall, these results indicate that GSPSO maintains robust performance across a broad range of tested settings, while extremely restrictive link-capacity and delay constraints remain principal factors limiting service admission and load balance.

VI. Conclusions and Future Work

Optimizing the deployment of Virtual Network Functions (VNFs) in Low Earth Orbit (LEO) satellite networks through efficient allocation of computing and communication resources is a critical challenge because it is complicated by stringent constraints on satellite resources, inter-satellite communication capabilities, and service quality requirements. Existing approaches do not adequately address the resulting constrained binary nonlinear optimization problem in resource-limited LEO satellite networks. To tackle this challenge, this work proposes a hybrid algorithm called Genetic Simulated annealing-based Particle Swarm Optimization (GSPSO) to derive the optimal deployment strategy. The proposed approach takes into account the computing capacity of satellites, propagation delay between satellites, and service deadline requirements. By precisely modeling the decision variables for VNF placement and service acceptance indicators, this work maximizes the service acceptance rate while ensuring balanced resource utilization through effective resource allocation and VNF deployment strategies. Comparative experiments over 90 independent runs demonstrate that GSPSO achieves the lowest mean fitness value of 0.016883 and the lowest fitness standard deviation of 0.006850 among the stochastic algorithms while maintaining a service acceptance rate of 1.000. Compared with PSO, SAPSO, SA, TROA, TH, and FAGD-Lite, GSPSO reduces the mean fitness by 11.4%, 98.60%, 99.97%, 99.99%, 99.78%, and 98.73%, respectively. In the scalability experiments involving 12–30 satellites, GSPSO accepts all service requests and maintains a Jain index of 0.971180 in the largest evaluated scenario. The comprehensive sensitivity results further show that performance degradation mainly occurs under extremely restrictive link-capacity and delay constraints, while high service acceptance and load fairness are recovered as these constraints are relaxed. GSPSO provides a better overall trade-off among service admission, constraint satisfaction, load balancing, and solution quality under the tested small- and medium-scale LEO satellite network scenarios.

In the future, we will extend this work in three directions. **First**, we will integrate GSPSO with deep learning techniques to guide the search process and improve the adaptability of SFC deployment in LEO satellite networks. **Second**, we will consider a dynamic LEO satellite network by incorporating time-varying satellite topologies, intermittent link availability, service migration, and rolling-horizon SFC orchestration. **Third**, we will design more scalable algorithms that integrate deep reinforcement learning, distributed optimization, or parallel computing with GSPSO, reducing the search overhead, improving solution quality in larger-scale constellations, and enhancing load balancing in practical satellite Internet.

References

[1] G. Wang, S. Zhou, S. Zhang, Z. Niu, and X. Shen, "SFC-Based Service Provisioning for Reconfigurable Space-Air-Ground Integrated Networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 7, pp. 1478–1489, 2020.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

TABLE X
HYPERPARAMETER SENSITIVITY ANALYSIS OF GSPSO OVER 30 INDEPENDENT RUNS.

Population Size	Mutation Rate	Mean Fit.	Service Acceptance Rate	Mean Jain Index	Avg. Time (s)	Std. Time (s)
20	0.00	0.499608	1.0000	0.990008	6.8133	1.7213
20	0.05	0.647232	1.0000	0.987055	6.2970	1.2057
20	0.20	1.010532	1.0000	0.979789	5.8047	1.1265
20	0.50	1.564328	1.0000	0.968713	5.3275	1.1673
40	0.00	0.375278	1.0000	0.992494	14.5514	4.3377
40	**0.05**	0.016883	1.0000	0.999662	14.3243	6.1634
40	0.20	0.865698	1.0000	0.982686	11.7553	1.9870
40	0.50	1.363964	1.0000	0.972721	9.9531	2.3408
80	0.00	0.325600	1.0000	0.993488	29.3965	7.2658
80	0.05	0.420279	1.0000	0.991594	27.6919	5.8659
80	0.20	0.662964	1.0000	0.986741	23.2859	4.9485
80	0.50	1.079179	1.0000	0.978416	20.0984	3.6053

TABLE XI
COMPREHENSIVE SENSITIVITY ANALYSIS OF GSPSO UNDER DIFFERENT LEO/SFC ORCHESTRATION SETTINGS.

Scenario	Value	Avg. Fit.	Acc. Ratio	Jain's Index
Request Density	5	0.173334	1.0000	0.996533
	10	0.054754	1.0000	0.998905
	15	0.023407	1.0000	0.999532
	20	0.012125	1.0000	0.999757
	25	0.009982	1.0000	0.999800
SFC Length	3	0.224582	1.0000	0.995508
	4	0.083126	1.0000	0.998337
	5	0.054754	1.0000	0.998905
	6	0.037300	1.0000	0.999254
	7	0.029276	1.0000	0.999414
Link Cap. Ratio	0.05	321.707100	0.6800	0.965859
	0.08	321.707100	0.6800	0.965859
	0.10	0.197439	1.0000	0.996051
	0.12	0.197439	1.0000	0.996051
	0.15	0.053888	1.0000	0.998922
Delay Bound Ratio	5×10^{-5}	838.889000	0.2000	0.222221
	7×10^{-5}	633.360700	0.4000	0.332786
	9×10^{-5}	417.478200	0.6000	0.650435
	1.1×10^{-4}	3.846251	1.0000	0.923075
	1.5×10^{-4}	0.978497	1.0000	0.980430
Comp. Res. Ratio	0.6	0.047538	1.0000	0.999049
	0.8	0.051668	1.0000	0.998967
	1.0	0.054754	1.0000	0.998905
	1.2	0.054754	1.0000	0.998905
	1.5	0.054754	1.0000	0.998905

[2] X. Qin, T. Ma, Z. Tang, X. Zhang, H. Zhou, and L. Zhao, "Service-Aware Resource Orchestration in Ultra-Dense LEO Satellite-Terrestrial Integrated 6G: A Service Function Chain Approach," *IEEE Transactions on Wireless Communications*, vol. 22, no. 9, pp. 6003–6017, 2023.

[3] Y. Hu, M. Chen, W. Saad, H. V. Poor, and S. Cui, "Distributed Multi-Agent Meta Learning for Trajectory Design in Wireless Drone Networks," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 10, pp. 3177–3192, 2021.

[4] W. Liu, H. Yang, and J. Li, "Multi-Functional Time Expanded Graph: A Unified Graph Model for Communication, Storage, and Computation for Dynamic Networks Over Time," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 2, pp. 418–431, 2023.

[5] P. Zhang, Y. Zhang, N. Kumar, and M. Guizani, "Dynamic SFC Embedding Algorithm Assisted by Federated Learning in Space–Air–Ground-Integrated Network Resource Allocation Scenario," *IEEE Internet of Things Journal*, vol. 10, no. 11, pp. 9308–9318, 2022.

[6] Z. Chen, N. Cheng, Z. Yin, J. He, and N. Lu, "Service-Oriented Topology Reconfiguration of UAV Networks with Deep Reinforcement Learning," in *2022 14th International Conference on Wireless Communications and Signal Processing*, Nanjing, China, 2022, pp. 753–758.

[7] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart Resource Allocation for Mobile Edge Computing: A Deep Reinforcement Learning Approach," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 3, pp. 1529–1541, 2019.

[8] R. Shi, J. Zhang, W. Chu, Q. Bao, X. Jin, C. Gong, Q. Zhu, C. Yu, and S. Rosenberg, "MDP and Machine Learning-Based Cost-Optimization of Dynamic Resource Allocation for Network Function Virtualization," in *2015 IEEE International Conference on Services Computing*, New York, NY, USA, 2015, pp. 65–73.

[9] C. Sun, J. Bi, Z. Zheng, and H. Hu, "SLA-NFV: an SLA-aware High Performance Framework for Network Function Virtualization," in *Proceedings of the 2016 ACM SIGCOMM Conference*, Florianopolis, Brazil, 2016, pp. 581–582.

[10] Y. Xiao, Q. Zhang, F. Liu, J. Wang, M. Zhao, Z. Zhang, and J. Zhang, "NFVdeep: Adaptive Online Service Function Chain Deployment with Deep Reinforcement Learning," in *IEEE/ACM 27th International Symposium on Quality of Service* , Phoenix, AZ, USA, 2019, pp. 1–10.

[11] Y. Liu, Y. Lu, X. Li, Z. Yao, and D. Zhao, "On Dynamic Service Function Chain Reconfiguration in IoT Networks," *IEEE Internet of Things Journal*, vol. 7, no. 11, pp. 10969–10984, Nov. 2020.

[12] H.-G. Kim, S. Park, S. Lange, D. Lee, D. Heo, H. Choi, J.-H. Yoo, and J. W.-K. Hong, "Graph Neural Network-based Virtual Network Function Deployment Optimization," *International Journal of Network Management*, vol. 31, no. 6, pp. 1–18, 2021.

[13] R. Wang, J. Zhang, Z. Gu, S. Yan, Y. Xiao, and Y. Ji, "Edge-enhanced Graph Neural Network for Du-cu Placement and Lightpath Provision in X-haul Networks," *Journal of Optical Communications and Networking*, vol. 14, no. 10, pp. 828–839, 2022.

[14] Y. Xie, L. Huang, Y. Kong, S. Wang, S. Xu, X. Wang, and J. Ren, "Virtualized Network Function Forwarding Graph Placing in SDN and NFV-Enabled IoT Networks: A Graph Neural Network Assisted Deep Reinforcement Learning Method," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 524–537, 2021.

[15] Q. Liu, *et al*., "A Multi-Swarm PSO Approach to Large-Scale Task Scheduling in a Sustainable Supply Chain Datacenter," *IEEE Trans. on Green Communications and Networking*, vol. 7, no. 4, pp. 1667–1677, Dec. 2023.

[16] Y. Liu, F. Zhou, T. Shang, and J. M. Torres-Moreno, "Disaster Protection for Service Function Chain Provisioning in EO-DCNs," *IEEE Transactions on Network and Service Management*, vol. 20, no. 3, pp. 2356–2369, Sept. 2023.

[17] M. Niu, *et al*., "HARS: A High-Available and Resource-Saving Service Function Chain Placement Approach in Data Center Networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 829–847, Jun. 2022.

[18] Y. C. Wang, R. H. Hwang, and Y. D. Lin, "Low-Latency Service Chaining With Predefined NSH-Based Multipath Across Multiple Datacenters," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 4721–4733, Dec. 2022.

[19] M. Wang, B. Cheng, S. Wang, and J. Chen, "Availability- and Traffic-Aware Placement of Parallelized SFC in Data Center Networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 182–194, Mar. 2021.

[20] M. Zhu, J. Gu, T. Shen, J. Zhang, and P. Gu, "Delay-aware and resource-efficient service function chain mapping in inter-datacenter elastic optical networks," *Journal of Optical Communications and Networking*, vol. 14, no. 10, pp. 757–770, Oct. 2022.

[21] J. Li, *et al*., "Digital Twin-Assisted, SFC-Enabled Service Provisioning in Mobile Edge Computing," *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 393–408, Jan. 2024.

[22] L. Rui, *et al*., "SFC Orchestration Method for Edge Cloud and Central Cloud Collaboration: QoS and Energy Consumption Joint Optimization Combined With Reputation Assessment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 10, pp. 2735–2748, Oct. 2023.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246

[23] Y. Liu, H. Lu, X. Li, Y. Zhang, L. Xi, and D. Zhao, "Dynamic Service Function Chain Orchestration for NFV/MEC-Enabled IoT Networks: A Deep Reinforcement Learning Approach," *IEEE Internet of Things Journal*, vol. 8, no. 9, pp. 7450–7465, May 2021.

[24] P. K. Thiruvasagam, A. Chakraborty, A. Mathew and C. S. R. Murthy, "Reliable Placement of Service Function Chains and Virtual Monitoring Functions With Minimal Cost in Softwarized 5G Networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1491–1507, Jun. 2021.

[25] J. He, N. Cheng, Z. Yin, C. Zhou, H. Zhou, W. Quan, and X.-H. Lin, "Service-Oriented Network Resource Orchestration in Space-Air-Ground Integrated Network," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 1, pp. 1162–1174, 2024.

[26] Z. Jia, Y. Cao, L. He, G. Li, F. Zhou, Q. Wu, and Z. Han, "NFV-Enabled Service Recovery in Space–Air–Ground Integrated Networks: A Matching Game-Based Approach," *IEEE Transactions on Network Science and Engineering*, vol. 12, no. 3, pp. 1732–1744, 2025.

[27] Z. Jia, Y. Cao, L. He, Q. Wu, Q. Zhu, D. Niyato, and Z. Han, "Service Function Chain Dynamic Scheduling in Space-Air-Ground Integrated Networks," *IEEE Transactions on Vehicular Technology*, pp. 1–15, 2025.

[28] J. Li, W. Shi, H. Wu, S. Zhang, and X. Shen, "Cost-Aware Dynamic SFC Mapping and Scheduling in SDN/NFV-Enabled Space–Air–Ground-Integrated Networks for Internet of Vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 5824–5838, 2022.

[29] P. Zhang, P. Yang, N. Kumar, and M. Guizani, "Space-Air-Ground Integrated Network Resource Allocation Based on Service Function Chain," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 7, pp. 7730–7738, 2022.

[30] J. Zhang, Q. Ye, K. Qu, Y. Sun, Y. Tang, D. Zhao, and T. Ye, "Learning-Assisted Dynamic VNF Selection and Chaining for 6G Satellite-Ground Integrated Networks," *IEEE Transactions on Vehicular Technology*, vol. 74, no. 1, pp. 1504–1519, 2025.

[31] J. He, N. Cheng, Z. Yin, H. Zhou, C. Zhou, K. Aldubaikhy, A. Alqasir, and X. Shen, "Load-Aware Network Resource Orchestration in LEO Satellite Network: A GAT-Based Approach," *IEEE Internet of Things Journal*, vol. 11, no. 9, pp. 15 969–15 984, 2024.

[32] O. Bouachir, M. Aloqaily, L. Tseng, and A. Boukerche, "Blockchain and Fog Computing for Cyberphysical Systems: The Case of Smart Industry," *Computer*, vol. 53, no. 9, pp. 36–45, 2020.

[33] H. Peng, D. Li, K. Abboud, H. Zhou, H. Zhao, W. Zhuang, and X. Shen, "Performance Analysis of IEEE 802.11p DCF for Multiplatooning Communications With Autonomous Vehicles," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 2485–2498, 2016.

[34] X. Zhang, Z. Xu, L. Fan, S. Yu, and Y. Qu, "Near-Optimal Energy-Efficient Algorithm for Virtual Network Function Placement," *IEEE Trans. on Cloud Computing*, vol. 10, no. 1, pp. 553–567, Mar. 2022.

[35] L. Liu, S. Guo, G. Liu, and Y. Yang, "Joint Dynamical VNF Placement and SFC Routing in NFV-Enabled SDNs," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4263–4276, 2021.

[36] M. A. Tahmasbi Nejad, S. Parsaeefard, M. A. Maddah-Ali, T. Mahmoodi, and B. H. Khalaj, "vSPACE: VNF Simultaneous Placement, Admission Control and Embedding," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 542–557, 2018.

[37] A. Pentelas, G. Papathanail, I. Fotoglou, and P. Papadimitriou, "Network Service Embedding Across Multiple Resource Dimensions," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 209–223, Mar. 2021.

[38] S. Moreno, A. M. Mora, P. Padilla, J. Carmona-Murillo, and P. A. Castillo, "Applying Ant Colony optimization for Service Function Chaining in a 5G Network," in *2019 Sixth International Conference on Internet of Things: Systems, Management and Security*, 2019, pp. 567–574.

[39] Y. Yue, B. Cheng, X. Liu, M. Wang, B. Li, and J. Chen, "Resource Optimization and Delay Guarantee Virtual Network Function Placement for Mapping SFC Requests in Cloud Networks," *IEEE Trans. on Network and Service Management*, vol. 18, no. 2, pp. 1508–1523, Jun. 2021.

[40] H. Li, Z. Kong, Y. Chen, L. Wang, Z. Lu, X. Wen, W. Jing, and W. Xiang, "Slice-Based Service Function Chain Embedding for End-to-End Network Slice Deployment," *IEEE Transactions on Network and Service Management*, vol. 20, no. 3, pp. 3652–3672, Sept. 2023.

[41] J. Sun, F. Liu, H. Wang, M. Ahmed, Y. Li, L. Zhang, and H. Zeng, "Network Function Placement Under Randomly Arrived Networking Traffic," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 4, pp. 1382–1398, Dec. 2021.

[42] H. Huang, C. Zeng, Y. Zhao, G. Min, Y. Zhu, W. Miao, and J. Hu, "Scalable Orchestration of Service Function Chains in NFV-Enabled Networks: A Federated Reinforcement Learning Approach," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2558–2571, Aug. 2021.

[43] H. Huang, J. Tian, G. Min, H. Yin, C. Zeng, Y. Zhao, and D. O. Wu, "Parallel Placement of Virtualized Network Functions via Federated Deep Reinforcement Learning," *IEEE/ACM Transactions on Networking*, vol. 32, no. 4, pp. 2936–2949, Aug. 2024.

[44] Z. Wang, C.-H. Lung, and C. Huang, "ML-Assistant Service Function Chaining Workload Scheduler for Cloud-Native inFrastructure," in *2025 IEEE Network Operations and Management Symposium*, 2025, pp. 1–6.

[45] G. Zheng, N. Wang, P. Qian, D. Griffin, and R. Tafazolli, "SDN-based Service Function Chaining in Integrated Terrestrial and LEO Satellite based Space Internet," *IEEE Journal on Selected Areas in Communications*, vol. 43, no. 2, pp. 537–550, Feb. 2025.

[46] Z. Zhang, H. Hu, and Y. Wu, "DRL-Based Dynamic SFC Orchestration Algorithm for LEO Satellite Networks," *Electronics*, vol. 14, no. 1, 2025.

[47] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: New insights," *Struct. Multidisc. Optim.*, vol. 41, no. 6, pp. 853–862, Jun. 2010, doi: 10.1007/s00158-009-0460-7.

[48] D. Wu, N. Jiang, W. Du, K. Tang, and X. Cao, "Particle Swarm Optimization with Moving Particles on Scale-Free Networks," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 497–506, 2018.

[49] Q. Li and W. Wang, "AVO Inversion in Orthotropic Media Based on SA-PSO," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 10, pp. 8903–8912, 2021.

[50] V. S. D. M. Sahu, P. Samal, and C. K. Panigrahi, "Tyrannosaurus Optimization Algorithm: A New Nature-inspired Meta-heuristic Algorithm for Solving Optimal Control Problems," *e-Prime-Advances in Electrical Engineering, Electronics and Energy*, vol. 5, pp. 1–21, 2023.

[51] Z. Liu, F. Peng, C. Liang, et al., "A Two-Stage Heuristic SFC Deployment Approach in Software Defined Satellite Networks," in *Proc. International Conference on Communications and Networking in China*, Cham, Switzerland: Springer Nature Switzerland, 2023, pp. 36–49.

[52] H. Abreha, H. Chougrani, I. Maity, Y. Drif, C. Politis, and S. Chatzinotas, "Fairness-Aware VNF Mapping and Scheduling in Satellite Edge Networks for Mission-Critical Applications," *IEEE Transactions on Network and Service Management*, vol. 21, no. 6, pp. 6716–6730, 2024.



Haitao Yuan (S'15–M'17–SM'21) received the Ph.D. degree in Computer Engineering from New Jersey Institute of Technology (NJIT), Newark, NJ, USA in 2020. He is currently a Deputy Director in the Department of Strategic Development, Wenchang International Aerospace City, Hainan, China. He is currently an Associate Professor at the School of Automation Science and Electrical Engineering at Beihang University, Beijing, China, and he is named in the world's top 2% of Scientists List since 2022. His research interests include the Internet of Things, deep learning, data-driven optimization, and computational intelligence algorithms. He received the Chinese Government Award for Outstanding Self-Financed Students Abroad, the 2021 Hashimoto Prize from NJIT, the Best Work Award in the 17th International Conference on Networking, Sensing and Control, and the Best Student Work Award Nominee in the 2024 IEEE International Conference on Systems, Man, and Cybernetics. He received the Best Paper in Theory and Technology from IEEE Robotics & Automation Society and IEEE Systems, Man & Cybernetics Society. He is an associate editor for IEEE Transactions on Systems, Man, and Cybernetics: Systems, IEEE Internet of Things Journal, and Expert Systems With Applications.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3746246



Dengyu Guo received his B.S. degree in Automation from Beihang University, China, in 2024. He has been pursuing his M.S. degree in Control Science and Engineering at the same university since 2024. His research interests include intelligent optimization algorithms, deep learning, LEO satellite networks, and network optimization and management.



Jing Bi (Senior Member, IEEE) received her B.S., and Ph.D. degrees in Computer Science from Northeastern University, Shenyang, China, in 2003 and 2011, respectively. She is currently a Professor with the Faculty of Information Technology, Beijing University of Technology, Beijing, China. She has over 170 publications in international journals and conference proceedings. Her research interests include distributed computing, cloud & edge computing, large-scale data analytics, machine learning, industrial internet, and performance optimization. She is now an Associate Editor of IEEE Transactions on Systems Man and Cybernetics: Systems. She is a senior member of the IEEE.



Jia Zhang (Senior Member, IEEE) received a PhD degree in computer science from the University of Illinois at Chicago, Chicago, IL, USA in 2000. She is the Cruse C. and Marjorie F. Calahan Centennial Chair in Engineering, Professor of the Department of Computer Science in the Lyle School of Engineering at Southern Methodist University. Her research interests emphasize the application of machine learning and information retrieval methods to tackle data science infrastructure problems, with a recent focus on scientific workflows, provenance mining, software discovery, knowledge graphs, and interdisciplinary applications of all of these interests in earth science. She is a senior member of the IEEE.



Rajkumar Buyya (Fellow, ACM; Fellow, IEEE) is a Redmond Barry Distinguished Professor and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He is Fellow of ACM, IEEE, and Academia Europea. He received a B.E and M.E in Computer Science and Engineering from Mysore and Bangalore Universities in 1992 and 1995, respectively, and a Ph.D. in Computer Science and Software Engineering from Monash University, Melbourne, Australia, in 2002. He was a Future Fellow of the Australian Research Council from 2012 to 2016. He has authored over 1,00 publications and seven textbooks. He is one of the highly cited authors in computer science and software engineering worldwide, with over 174,880 citations and an h-index of 183. He was recognized as a "Web of Science Highly Cited Researcher" from 2016 to 2021 by Thomson Reuters.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 13,2026 at 05:23:22 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.