



## Article

# Service Function Chain Deployment Algorithm Based on Deep Reinforcement Learning in Space–Air–Ground Integrated Network

Xu Feng <sup>1</sup>, Mengyang He <sup>1,2</sup>, Lei Zhuang <sup>3,\*</sup>, Yanrui Song <sup>3</sup> and Rumeng Peng <sup>1</sup>

<sup>1</sup> The School of Cyber Science and Engineering, Zhengzhou University, Zhengzhou 450000, China; uxengf@126.com (X.F.); iemyhe@163.com (M.H.); prm1212@gs.zzu.edu.cn (R.P.)

<sup>2</sup> Song Shan Laboratory, Zhengzhou 450000, China

<sup>3</sup> School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou 450000, China; syr\_up@gs.zzu.edu.cn

\* Correspondence: ielzhuang@zzu.edu.cn

**Abstract:** SAGIN is formed by the fusion of ground networks and aircraft networks. It breaks through the limitation of communication, which cannot cover the whole world, bringing new opportunities for network communication in remote areas. However, many heterogeneous devices in SAGIN pose significant challenges in terms of end-to-end resource management, and the limited regional heterogeneous resources also threaten the QoS for users. In this regard, this paper proposes a hierarchical resource management structure for SAGIN, named SAGIN-MEC, based on a SDN, NFV, and MEC, aiming to facilitate the systematic management of heterogeneous network resources. Furthermore, to minimize the operator deployment costs while ensuring the QoS, this paper formulates a resource scheduling optimization model tailored to SAGIN scenarios to minimize energy consumption. Additionally, we propose a deployment algorithm, named DRL-G, which is based on heuristics and DRL, aiming to allocate heterogeneous network resources within SAGIN effectively. Experimental results showed that SAGIN-MEC can reduce the end-to-end delay by 6–15 ms compared to the terrestrial edge network, and compared to other algorithms, the DRL-G algorithm can improve the service request reception rate by up to 20%. In terms of energy consumption, it reduces the average energy consumption by 4.4% compared to the PG algorithm.

**Keywords:** space–air–ground integrated network; DRL; resource allocation; NFV



**Citation:** Feng, X.; He, M.; Zhuang, L.; Song, Y.; Peng, R. Service Function Chain Deployment Algorithm Based on Deep Reinforcement Learning in Space–Air–Ground Integrated Network. *Future Internet* **2024**, *16*, 27. <https://doi.org/10.3390/fi16010027>

Academic Editor: Symeon Papavassiliou

Received: 5 December 2023

Revised: 27 December 2023

Accepted: 8 January 2024

Published: 16 January 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The Space–Air–Ground Integrated Network (SAGIN) enables ubiquitous and seamless connectivity through the introduction of network devices such as Low Earth Orbit (LEO) satellites and High-Altitude Platforms (HAPs). SAGIN expands the coverage area of the existing network and is, therefore, capable of handling tasks that cannot be performed using terrestrial networks alone, and it has become an important direction in the development of the Internet in the future [1,2]. SAGIN, through the interaction of heterogeneous devices, can provide new solutions for processes in industry/agriculture that are not suitable for direct manual operation. For example, satellites can be used in the interaction of heterogeneous equipment in SAGIN to remotely send machine operation instructions which can realize the remote monitoring and operation of pesticide spraying, mechanical operation, terminal cargo transportation, etc. This can not only save much of the labor cost, but also promote the intelligentization of production.

However, SAGIN still faces two significant challenges. The first is the management of heterogeneous network resources. SAGIN includes satellite networks, aircraft networks, and ground networks, and these heterogeneous devices can cause complex end-to-end resource allocation due to differences in the configurations, standards, and performance. The second is the problem of regional network resource constraints. Although many studies have considered the application of Mobile Edge Computing (MEC) on satellites

and HAPs to reduce the computational pressure on terrestrial networks [3], the problem of the unbalanced allocation of regional computing resources still needs to be solved. For example, more service base stations are often deployed in economically developed regions, which tend to possess excess resources. In contrast, remote regions usually face the problem of resource shortages and need help even to achieve the expected delivery of user services.

Therefore, a reasonable resource management and scheduling structure is needed to access heterogeneous devices in SAGIN, which ensures accurate and fast access to resource information, different Qualities of Service (QoSs), and the efficient deployment of tasks. In addition, proper resource scheduling algorithms are needed to allocate different resources for various services, reducing the amount of computing nodes that are turned on and the deployment cost while scheduling inactive resources in other nearby areas on time to satisfy the network demand and ensure that the latency is within a tolerable range.

Network Function Virtualization (NFV) and Software-Defined Networking (SDN) provide the technical basis for managing SAGIN heterogeneous device resources. The introduction of the SDN and NFV releases SAGIN from the constraints of proprietary hardware, enabling efficient access to different devices and dynamic sharing of network infrastructure and resources between heterogeneous networks. The SDN separates the control and data planes, enabling the ability to defining and controlling the network in a software-programmable structure [4]. The SDN has been proposed to enable flexible resource management and resource allocation in Earth observation missions [5]. NFV enables the data plane virtualization and hardware implementation of communication devices in software [6–8]. In the SDN/NFV structure, a Service Function Chain (SFC) composed of multiple Virtual Network Functions (VNFs) can guide user traffic according to precise policies and efficiently utilize limited computing, storage, bandwidth, and other network resources. In SAGIN, the VNFs must be matched with heterogeneous resources, and the deployment strategies are further designed and optimized by resource scheduling algorithms to reduce the operator costs while ensuring tolerable communication latency.

Researchers have performed much work on these two issues and have produced many excellent results. However, the currently proposed resource management structures need to thoroughly consider SAGIN's complex network structure and multiple heterogeneous devices. Moreover, after the introduction of the SDN and NFV, SFC orchestration studies have focused on the load of the ground and air nodes, and there is no study on the energy consumption and delay of heterogeneous devices. However, the resources of the air devices are limited, and the air devices also introduce different degrees of delay when communicating with the ground devices, so it is necessary to study them further.

Therefore, to address the problem of heterogeneous resource management and scheduling in SAGIN scenarios, we propose a SAGIN resource management structure that introduces the SDN, NFV, and MEC to improve the utilization efficiency of the heterogeneous resources. In addition, we researched the resource scheduling and QoS optimization problem under this structure and designed a scheduling algorithm that takes into account both energy consumption and latency.

The primary contributions of this article are as follows:

1. We propose a SAGIN-MEC structure based on SDN/NFV and MEC. It has a multi-level distributed SDN control structure to achieve the harmonious scheduling of heterogeneous resources;
2. An optimization model is developed by comprehensively considering resource scheduling in SAGIN scenarios. The model is designed to reduce the system's energy consumption while meeting the constraints of the latency and network resources;
3. A hybrid algorithm DRL-G based on Deep Reinforcement Learning (DRL) and a greedy algorithm are proposed to optimize the SFC resource scheduling under SAGIN, reducing the energy consumption and cost while ensuring the efficient use of network resources and tolerable latency.

The other parts of this article are organized as follows: Section 2 introduces the related work of the SAGIN structure and SFC orchestration; Section 3 introduces the structure and topology of SAGIN; Section 4 sets up the mathematical model; Section 5 proposes the optimization algorithm; Section 6 carries out the experimental verification and analysis, and Section 7 finally summarizes this article.

## 2. Related Work

### 2.1. Network Structure

The development process of SAGIN can not avoid the challenges of network structure. In order to achieve the convergence of the heterogeneous devices, researchers have carried out some studies on structure design.

Early studies of SAGIN structure did not consider the change in satellites from merely providing forwarding functions to supporting information storage and processing, nor did they completely consider the application of SDN and NFV technologies in converged networks. For example, KOTA et al. proposed the concept and definition of an early satellite terrestrial converged network containing only satellite and terrestrial communication networks [9]. However, they proved the possibility of satellite and terrestrial networks communicating with each other only at the physical layer, failing to solve the resource management and allocation problems in the heterogeneous networks. WANG et al. introduced MEC into satellite networks and proposed a bilateral computation offloading of satellite terrestrial networks in the structure [10], which gives full play to the advantage of comprehensive satellite coverage and solves the problem of limited ground network services. The structure does not have a specific resource management model, although it proposes to unify the resources of satellite ground networks for management.

The SDN, NFV, and MEC technologies have recently received increasing attention from SAGIN researchers. For example, Li et al. proposed a SAGIN network structure consisting of LEO satellites and civil aircraft. However, the structure ignores the management role of Medium Earth Orbit (MEO) satellites in SAGIN. GIAMBENE et al. proposed a satellite access terrestrial network structure designed for eMBB scenarios [11], where the satellite backhaul network is connected to the 5G core network through a terrestrial gateway to realize satellite and 5G convergence, and user terminals can communicate with satellites either directly or via terrestrial satellite relay (satellite terminals only), but the auxiliary functions of the aircraft network have not been considered. Cao et al. proposed the SAGIN structure in the Internet of Vehicles scenario, but the centralized management of the whole network needs to be considered [12]. Although researchers have extensively studied SAGIN through the latest technological means and achieved specific results, they have yet to consider the complex structure of the converged network and the large number of heterogeneous devices comprehensively. In response to the above problems, this paper proposes a SAGIN network structure that uses SDN controllers for hierarchical management. This structure can efficiently manage heterogeneous devices in converged networks and achieve efficient and flexible collaborative work among heterogeneous networks.

### 2.2. DRL

Deep Reinforcement Learning (DRL) is a method that combines deep learning with reinforcement learning. Deep learning is used for perceiving and representing things, while reinforcement learning focuses on learning strategies for problem solving. DRL builds predictive models of the environment and rewards through neural networks and trains this model through interaction with the environment to select the best action to maximize the expected reward. In recent years, with the rapid development of DRL technology, many researchers have begun to use DRL to solve resource optimization problems. Giannopoulos et al. proposed a resource allocation algorithm based on Deep Q-Learning to actively adjust the power of network transmitters to improve total throughput allocation [13]. Lyu et al. proposed a multi-agent deep learning algorithm for task offloading to reduce task latency [14]. Trakadas et al. proposed embedding technologies such as

Federated Learning and Supervised Learning into the algorithm framework to meet the needs of decentralized edge computing and local privacy protection [15]. Due to the high computational cost of traditional heuristic algorithms and the inability to generalize their solutions, DRL algorithms are more promising in generating solutions in large-scale networks. These studies on DRL provide a solid theoretical foundation for the algorithms studied in this paper.

### 2.3. Resource Scheduling

The current problem for resource scheduling and QoS optimization aspects in SAGIN has proved to be an NP-hard problem [16], and most of the existing solutions use Exact Solution Algorithms and Heuristic Algorithms to solve it.

There are many excellent results in existing research on SFC scheduling. For example, Zhou et al. conducted simulation experiments in small-scale scenarios using Matlab SCIP [17], but this approach could be more effective in large-scale environments. Li et al. proposed a VNF remapping and scheduling algorithm based on Tabu Search to improve the request acceptance rate of SFC [18]. However, the only experiments were conducted in small simulation scenarios.

Although there are many studies on SFC orchestration in other scenarios and excellent results have been achieved, there needs to be more studies on resource scheduling and QoS optimization in SAGIN. Zhang et al. proposed a joint learning-based algorithm to deploy SFC in SAGIN [19]. Li et al. proposed a heuristic SFC deployment algorithm with an inter-domain path calculation method based on surgical inter-domain path computation, aiming to reduce the load on the compute nodes [20]. Gao et al. proposed a Location-Aware Resource Allocation (LARA) algorithm based on Greedy and IBM CPLEX 12.10 to reduce the average utilization of compute resources in satellite and terrestrial networks [21]. Han et al. proposed a DRL-based SFC deployment algorithm to reduce end-to-end latency in large-scale LEO networks [22]. Qin et al. formulated the SFC embedding problem as a congestion game and proposed three algorithms suitable for different scenarios to meet users' latency requirements [23]. He et al. proposed a load-aware SFC orchestration algorithm to improve service capacity and load balancing. However, these studies should have considered the energy consumption and delay of the infrastructure in SAGIN [24]. Due to the complex structure and variable topology of the SAGIN, its problems in SFC are more complex and need more profound research. In order to meet the challenges of complex and changing environments in SAGIN, this paper is dedicated to minimizing the service energy consumption while satisfying the strict constraints on network resources and delays imposed by VNF in SFC. To this end, we employ a DRL-based algorithm to quickly obtain a solution set of potentially optimal deployment scenarios. Subsequently, through further filtering by heuristic algorithms, we can find the optimal deployment scheme precisely.

### 3. Network Structure

Regarding SAGIN structure, the current research has not fully considered the application of edge computing technology in converged networks. Those also have not well solved the problems of device heterogeneity, node effectiveness, and resource limitation in converged networks.

According to the difficulty of heterogeneous resource management in SAGIN, a SAGIN structure based on the SDN/NFV is proposed in this article, and its logic is shown in Figure 1. We put the abstract and application processing functions of physical devices on the MEC host so that the devices can be accessed according to procedures, free from hardware constraints. Therefore, the network structure proposed by us no longer divides the heterogeneous network into different layers, but into five layers: application layer, centralized control layer, control layer, data layer, and infrastructure layer.

The application layer contains the applications needed for the Industrial Internet of Things (IIoT) and Internet of Agriculture (IoA), such as machine operation, energy

extraction, and pesticide spraying, and provides the corresponding functions for the request of the centralized controller.

The centralized control layer is responsible for the collection of resource conditions; it develops management policies based on network resources and provides functions such as node mobility management, converged network topology reconstruction, network task scheduling, and heterogeneous resource management. This layer collects data from the control layer on the one hand and directs the control layer to perform its work on the other.

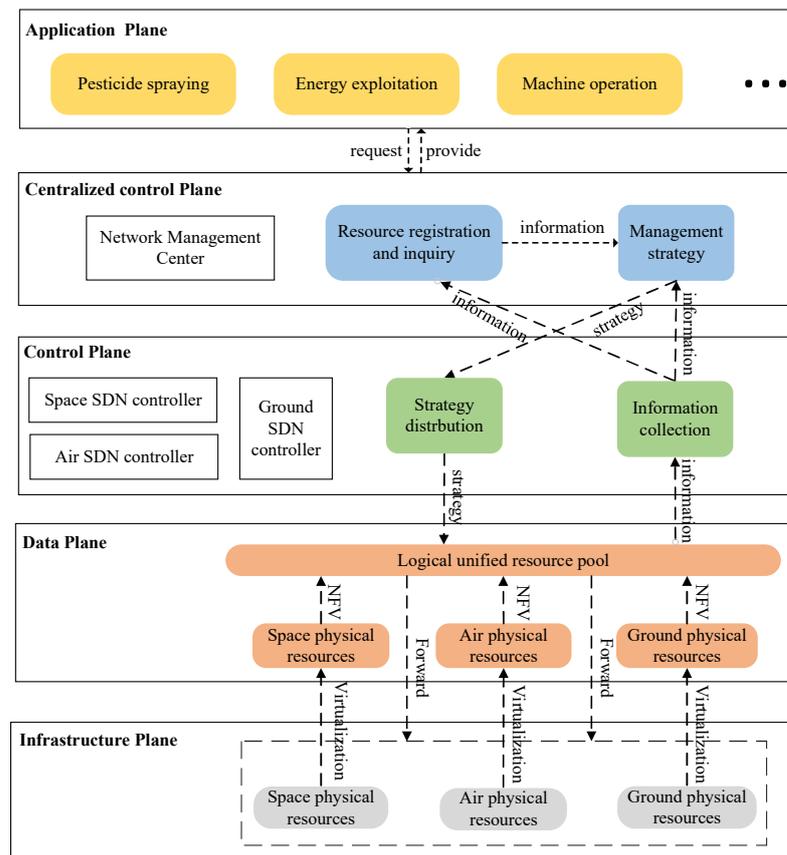


Figure 1. Logical structure of SAGIN based on the SDN/NFV for IoA/IIoT.

The control layer receives data from the data layer and feeds it back to the centralized control layer, allocating node resources and controlling data forwarding according to the policies of the centralized control layer. In order to solve the difficulties in the management of a large number of MEC servers and network nodes, the effective identification of new access network edge nodes is realized and the effectiveness of nodes is ensured. We divide SAGIN into three layers for control, namely the satellite network layer, the aircraft network layer, and the ground network layer. At the same time, to better manage SAGIN, a global SDN controller (SDNC) should be set to control all layers to complete tasks in a unified and coordinated manner. Meanwhile, primarily SDNCs and sub-SDNCs should be set at each layer. In the satellite network, the primary controllers are placed on Geostationary Earth Orbit (GEO) satellites, and sub-SDNCs are placed on the Medium Earth Orbit (MEO) satellites. In the aircraft network, one or a few High-Altitude Platforms (HAPs) are selected as the primary SDNCs, and the remaining HAPs are sub-control nodes. In the terrestrial network, servers are placed in the backbone network as primary SDNCs and edge servers are placed as sub-SDNCs.

The data layer integrates space-based, air-based, and ground-based computing resources into a unified resource pool through the NFV. Then, it forwards data to each substrate node according to the command of the control layer.

The infrastructure layer provides necessary resources for the upper layer, such as computing resources, storage resources, and bandwidth resources. The main facilities are GEO satellites, MEO satellites, LEO satellites, HAPs, UAPs, core servers, and edge servers, whose topologies are shown in Figure 2. Both ground base stations and MEO satellites can communicate directly with LEO satellites and GEO satellites. MEO satellites can communicate with other MEO satellites. In contrast, LEO satellites cannot communicate directly with GEO satellites or communicate with each other, and GEO satellites cannot communicate with other similar satellites.

The altitude of each orbit is shown in Table 1. LEO and GEO are 700 to 1500 km and 35,790 km away from the Earth’s surface, respectively, while MEO is between 2000 and 20,000 km away from the Earth’s surface, generally about 10,000 km away. Satellites transmit signals through radio waves, and the propagation speed is approximately the speed of light. The delay of data transmission by MEO satellites to the Earth’s surface is about 56 ms, and that by GEO satellites to the Earth’s surface is about 120 ms. As the delay is too long to meet user needs, MEC is not considered to be deployed on MEO and GEO satellites.

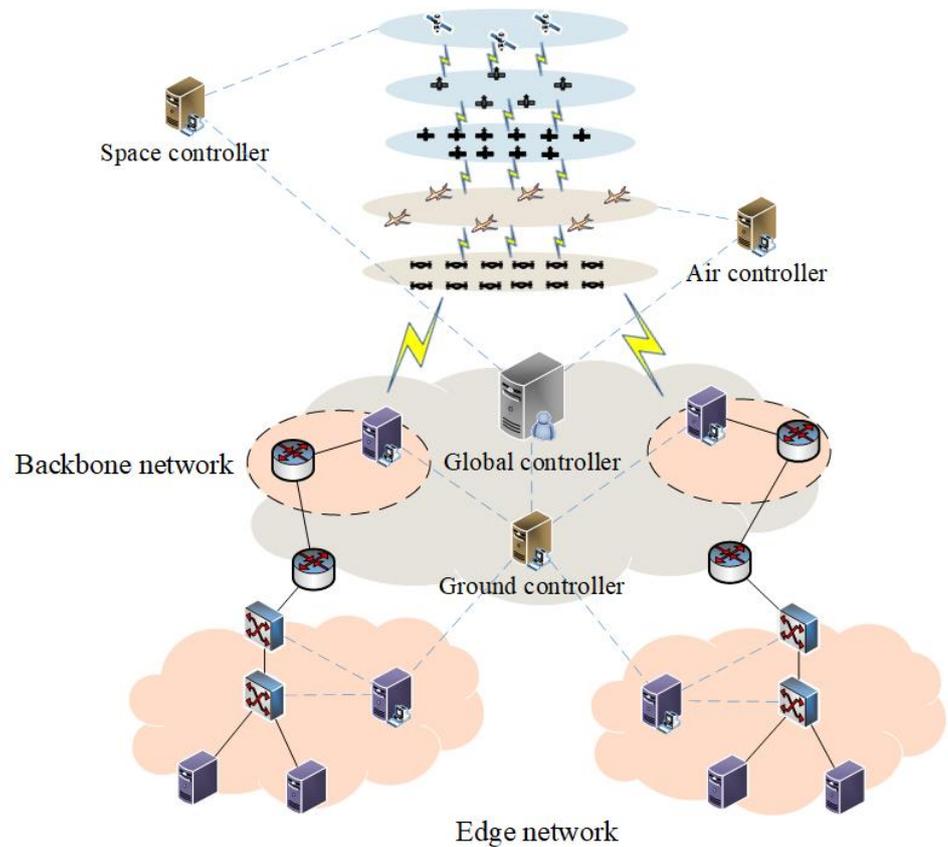


Figure 2. SAGIN topology.

Table 1. Distance of satellite from Earth’s surface.

Satellites	Orbit Height/km
GEO satellites	35,790
MEO satellites	2000~20,000 (10,000)
LEO satellites	700~1500

In view of the limited resources in SAGIN, an efficient dynamic resource management method is needed to deal with the unbalanced allocation of resources in converged networks to ensure stability of services. The dynamic resource management method under the SAGIN

structure based on the SDN and NFV is to combine VNFs into SFC orderly in the form of logical links so as to guide traffic through according to specific policies. As shown in Figure 3, there are two service requests, SFC-1 and SFC-2, where the traffic of SFC-1 passes through the VNFs carried by the non-ground node. Currently, the satellite nodes and the aircraft nodes have three functions. Firstly, the traffic can flow through the node without laying ground optical cables. In addition, it can balance the traffic load. For example, when some routes on the local network are congested, tasks are unloaded to the non-ground network. Furthermore, it shortens the delay by reducing the number of end-to-end hops.

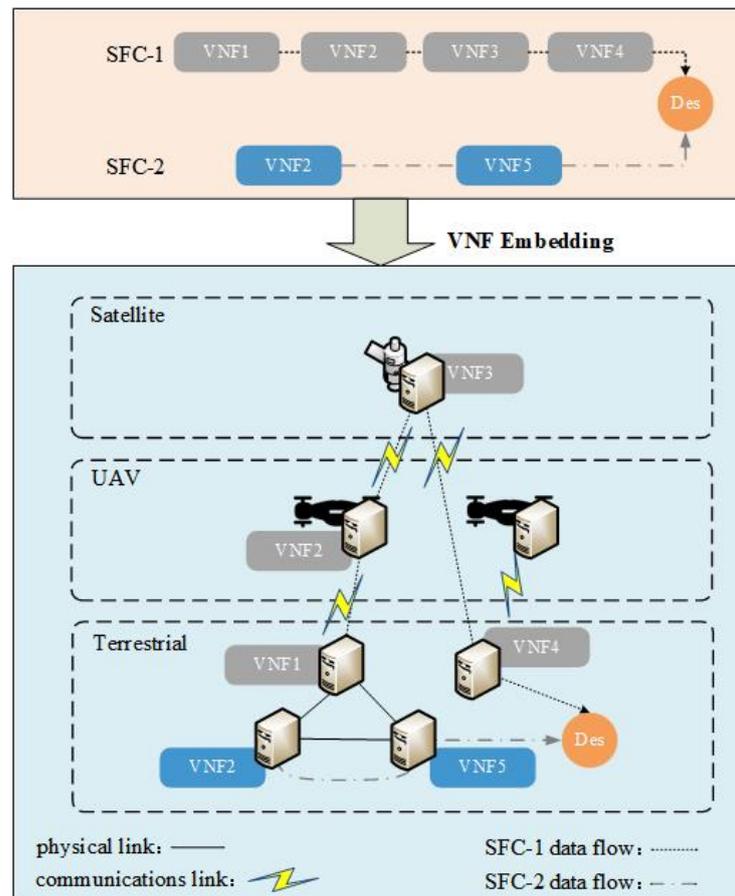


Figure 3. Example of SFC placement in SAGIN scenario.

Relative to terrestrial edge cloud networks, SAGIN-MEC utilizes SDN/NFV to achieve unified resource scheduling across different heterogeneous networks, which can improve the request acceptance rate and reduce the end-to-end latency of SFCs, and the specific results are presented in detail in the experiments and analyses in Section 4.

#### 4. Mathematical Model

In this section, we describe the VNF placement problem and related constraints in SAGIN scenarios in detail.

SFC:

We assume that the set of VNFs is  $V = \{v_1, v_2, \dots, v_n\}$ , which consists of  $n$  different VNFs. Network Service  $s = \{v_1, v_2, \dots, v_m\}$  is composed of  $m$  VNFs, where  $v \in V$ .

Network systems:

Firstly, we need to define a network system topology graph  $G = (H, L)$ , assuming that there are  $n$  host servers (containing ground, aircraft, and satellite servers). The hosts can be represented as  $H = \{h_1, h_2, \dots, h_n\}$ , the set of available computing resources owned by

these hosts is  $R$ , and the set of links between the hosts is defined as  $L, b_i$ , and  $l_i^s$  denotes the bandwidth and delay of link  $i$ , respectively.

Placement strategy:

The VNFs of the service chain need to be placed as optimally as possible in the host servers, and we use  $x_{vh}$  to indicate whether VNF  $v$  is placed in host  $h$ :

$$x_{vh} = \begin{cases} 1 & \text{VNF } v \text{ in host } h \\ 0 & \text{otherwise} \end{cases}$$

$y_h$  represents whether host  $h$  is occupied or not:

$$y_h = \begin{cases} 1 & \text{existence VNF in host } h \\ 0 & \text{otherwise} \end{cases}$$

The usage of link  $i$  is denoted as

$$z_i = \begin{cases} 1 & \text{data packet passes through link } i \\ 0 & \text{otherwise} \end{cases}$$

Energy consumption:

Energy consumption is divided into two parts during VNF placement: one is required for the hosts to process tasks, and the other is the energy consumption for link data transmission. The energy consumption of hosts is related to the number of resources it uses. We assume that  $W_h^{min}$  is the minimum energy required by host  $h$ , and  $W_h^{cpu}$  is the energy needed for each  $cpu$  of  $h$ ; then, the total energy consumption of hosts can be expressed as

$$energy\_cpu = \sum_{h \in H} (W_h^{cpu} \cdot \sum_{v \in S} r_{rv} \cdot x_{vh} + W_h^{min} \cdot y_h), \quad (1)$$

where  $r_{rv}$  represents the computing resources requested by VNF  $v$ .

The energy consumption of the links can be expressed as

$$energy\_link = \sum_{i \in L} W_i^{bw} \cdot \sum_{v \in S} b_v^s \cdot x_{vh}, \quad (2)$$

where  $W_i^{bw}$  is the energy consumed per bandwidth unit flowing through link  $i$ . Then, the total energy consumed to provide service  $s$  can be expressed as

$$F_1 = energy\_cpu + energy\_link. \quad (3)$$

After all of the VNFs about the  $s$  are placed in the host servers, the resources occupied by each host can be expressed as Equation (4).

$$cpu\_occupy = \sum_{v \in S} r_{rv} \cdot x_{vh} \quad \forall h \in H, r \in R. \quad (4)$$

The available computing resources of the host servers are represented as

$$cpu\_available = y_h \cdot a_{rh} \quad \forall h \in H, r \in R, \quad (5)$$

where  $a_{rh}$  denotes the available computing resources of host  $h$ .

During storage, the computing resources occupied by service  $s$  cannot exceed the available resources of the host. The specific equation is shown as follows:

$$F_2 : cpu\_occupy \leq cpu\_available. \quad (6)$$

Bandwidth:

Similarly, the deployment of VNFs also needs to meet the bandwidth constraints of the link. The bandwidth occupied by SFC  $s$  can be expressed as

$$bw\_occupy = \sum_{v \in s} b_v^s \cdot x_{vh}. \quad (7)$$

The bandwidth owned by link  $i$  is expressed as

$$bw\_available = z_i \cdot b_i \quad \forall i \in L. \quad (8)$$

The bandwidth occupied by SFC  $s$  must not exceed the available bandwidth of each link during the placement. Therefore, the constraint is expressed as

$$F_3 : bw\_occupy \leq bw\_available. \quad (9)$$

End-to-end latency:

End-to-end delay is divided into two parts: one is the time required by the host to process the VNF, and the other is the time required by the packet during transmission. The delay in processing VNF is expressed as

$$delay\_cpu = \sum_{h \in H} \sum_{v \in s} d_v^{cpu} \cdot x_{vh} \quad \forall h \in H, \quad (10)$$

where  $d_v^{cpu}$  represents the delay of VNF  $v$  processing by the host.

We let  $d_i^s$  represent the time required by link  $i$  to transmit SFCs data packets. The delay of this process is expressed as

$$delay\_link = \sum_{i \in L} \sum_{v \in s} d_i^s \cdot z_i \quad \forall h \in H. \quad (11)$$

The end-to-end delay of  $s$  must meet its maximum allowable time,  $T$ , which can be expressed as

$$F_4 : delay\_cpu + delay\_link \leq T. \quad (12)$$

Deployment constraints:

VNF in SFC  $s$  can only be placed in one host at a time during the placement:

$$F_5 : \sum_{h \in H} x_{vh} = 1 \quad \forall v \in s. \quad (13)$$

Summary of objectives:

Considering energy consumption, computing resources, bandwidth, and service delay comprehensively, we defined an optimization model as follows:

$$\min(F_1) \quad s.t. \quad F_2, F_3, F_4, F_5. \quad (14)$$

## 5. Algorithm Design

The resource allocation problem in the SAGIN scenario is essentially a multi-objective optimization problem (MaOP), which proves to be an NP-hard problem. This scenario results in a much more complex environment than a ground network alone due to the highly dynamic nature of satellites and vehicles.

To solve this problem, we propose DRL-G, which is a hybrid algorithm combining deep reinforcement learning and the heuristic algorithm. At present, the effect of deep reinforcement learning in solving combinatorial optimization problems is as good as that of high-performance heuristic algorithms [25]. However, due to the fact that the deep reinforcement learning algorithm cannot fully explore all actions, we use the heuristic algorithm to conduct a local search based on its results so as to obtain better results than the original algorithm. The specific algorithm flow is shown in Figure 4. After receiving

the service request, DRL-G first predicts the placement scheme through the sequence-to-sequence model within the allowed response time and then finds the best strategy through the greedy algorithm.

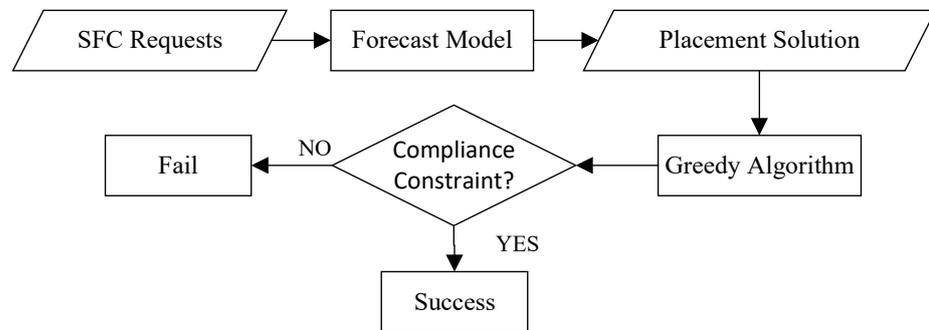


Figure 4. Workflow of proposed DRL-G.

5.1. Predictive Models

The prediction model is used to predict server and link occupancy when the SFC request arrives, and it is the core part of the whole method. It consists of four parts, namely the input, the encoder, the decoder, and the output layer, and its structure is shown in Figure 5.

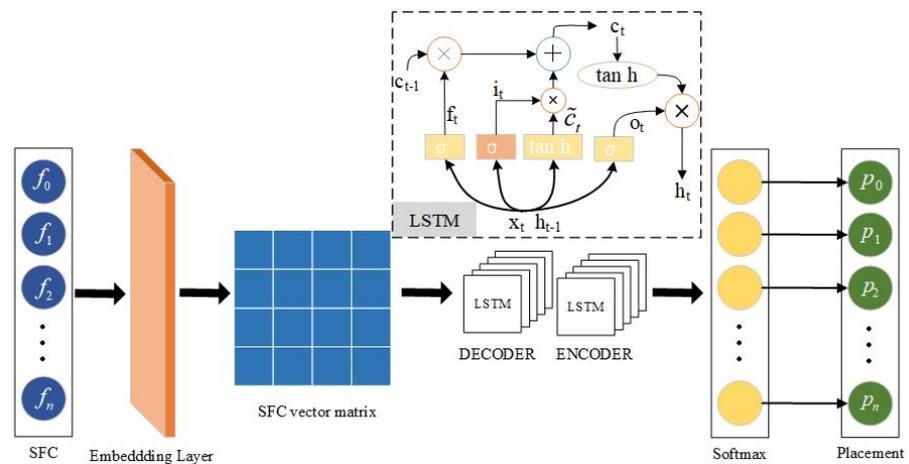


Figure 5. SFC deployment prediction model based on sequence-to-sequence model.

The input layer processes and normalizes the computational and bandwidth resource data required to extract the SFC into a  $2 \times n$  feature matrix  $x = [c, bw]$ , where  $n$  is the chain length of the longest SFC,  $c$  is the computational resource normalized data, and  $bw$  is the bandwidth resource normalized data. The normalized formula is as follows:

$$c_i = \frac{C_i - \min(C_i)}{\max(C_i) - \min(C_i)}, \tag{15}$$

$$bw_i = \frac{BW_i - \min(BW_i)}{\max(BW_i) - \min(BW_i)}, \tag{16}$$

where  $C$  and  $B$  are the amount of computing and bandwidth resources required by VNF, respectively.

The core of the neural network model employs encoders and decoders for sequence-to-sequence models. Its most important feature is that the length of the input and output sequences is variable, which makes it ideal for use in situations where the SFC chain length is uncertain.

The encoder and the decoder consist of several Long Short-Term Memory Networks (LSTMs). LSTM is a special kind of Recurrent Neural Network (RNN). Each LSTM input consists of the previous LSTM output, cell states  $h_{t-1}$ ,  $c_{t-1}$ , and current input  $x_t$ , where  $x_t$  is the data of the  $t$ th row of feature matrix  $x$ . The formulae for the output of the LSTMs,  $h$ , and the cell state,  $c_t$ , are as follows:

$$c_t = \sigma(\theta_f \cdot [h_{t-1}, x_t] + b_f) \cdot c_{t-1} + \sigma(\theta_i \cdot [h_{t-1}, x_t] + b_i) \cdot \tanh(\theta_c \cdot [h_{t-1}, x_t] + b_c), \quad (17)$$

$$h_t = \sigma(\theta_o \cdot [h_{t-1}, x_t] + b_o) \cdot \tanh(c_t), \quad (18)$$

$$f_t = \sigma(\theta_f \cdot [h_{t-1}, x_t] + b_f), \quad (19)$$

where  $\theta_f$ ,  $\theta_i$ ,  $\theta_c$ ,  $\theta_o$ ,  $b_f$ ,  $b_i$ ,  $b_c$ , and  $b_o$  are the parameters on which the neural network needs to be trained.

The output layer is a sampling function that changes continuous actions into discrete actions. This is achieved by predicting the occupied servers by sampling according to the output probabilities in the decoder. The higher the probability, the greater the chance that a server will be drawn.

## 5.2. The Trained Algorithm

We adopt the policy gradient to train the prediction model, and the training algorithm is shown in Algorithm 1.

---

### Algorithm 1 Train Agent Network

---

**Input:** action network  $\pi(a|s, w)$ , critic network  $\hat{v}(s, w_v)$

- 1: **Random** policy parameter  $w$  and critic parameter  $w_v$
  - 2: **for** epoch = 1, 2... **do**
  - 3:   reset  $d^* \leftarrow 0$
  - 4:    $s_j \sim \text{SampleInput}(S)$  for  $j \in \{1, \dots, B\}$
  - 5:    $p_j \sim \text{SampleSolution}(\pi_w(\cdot|s))$  for  $j \in \{1, \dots, B\}$
  - 6:    $b_j \leftarrow b_{w_v}(s_j)$  for  $j \in \{1, \dots, B\}$
  - 7:   compute cost function :  $L(p_j)$  for  $j \in \{1, \dots, B\}$
  - 8:    $g_w = 1/B \cdot \sum_{j=1}^B (L(p_j) - b(s_j)) \cdot \nabla_w \log \pi_w(p_j|s_j)$
  - 9:    $\mathcal{L}(w_v) = 1/B \cdot \sum_{j=1}^B \|b_{w_v}(s_j) - L(p_j|s_j)\|^2$
  - 10:    $w \leftarrow \text{Adam}(w, g_w)$
  - 11:    $w \leftarrow \text{Adam}(w_v, \mathcal{L}(w_v))$
  - 12: **end for**
  - 13: **return**  $w$  and  $w_v$
- 

The algorithm model is divided into three parts: agent, environment, and state. The agent includes the sequence-to-sequence model and value evaluator. The environment is the network system constructed in this article, and the state includes SFC and the resource situation of each host and link. After receiving the SFC, the agent selects the VNF placement policy according to the current state. Then, the environment places the VNF according to the specific location and generates a new state, and gives back the corresponding reward value to the agent so as to guide the agent to constantly explore a better placement policy. The specific algorithm model is shown in Figure 6.

We suppose that the VNF placement problem in SAGIN has  $n$  states, and its state space is  $S = \{s_1, s_2, \dots, s_n\}$ . The size of the state space is closely related to the number of underlying network resources and the type of VNF. When  $n$  compute nodes are considered, the set of compute resources of these compute nodes is  $R_c = \{r_1, r_2, \dots, r_n\}$ . Meanwhile, the bandwidth resources connecting these compute nodes are  $R_b = \{b_1, b_2, \dots, b_n\}$ . If there are  $m$  VNF types, then  $m \cdot \sum_{i=0}^n r_i \cdot b_i$  states are formed. The action space is related to the

number of computing nodes, assuming that the underlying network has  $n$  computing nodes. Then, the agent has  $n$  actions, and its action space is  $A = \{a_1, a_2, \dots, a_n\}$ . The agent interacts with the environment  $t$  times to obtain the trajectory and deployment scenario can be represented as:  $\tau = \{(s_1, a_1), (s_2, a_2), \dots, (s_t, a_t)\}$ ,  $p = \{a_1, a_2, \dots, t_t\}$ .

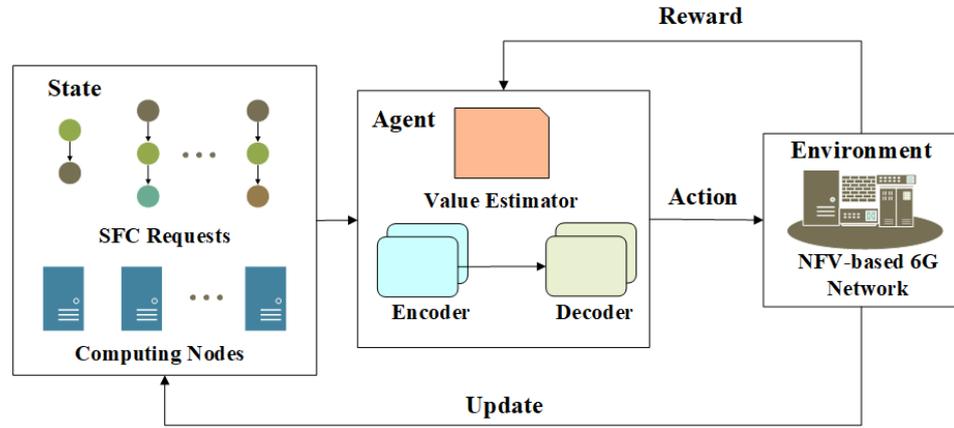


Figure 6. RL training process of prediction model.

After the configuration according to deployment scenario  $p$ , the environment evaluates whether the network resource constraints, delay constraints, and energy consumption magnitude are met and offers a feedback value. Subsequently, the neural network updates the policy based on this feedback value.

We need to find strategy  $\pi(a|s)$  that enables optimal placement of the SFC. Therefore, this strategy cannot be used directly; we set policy function  $\pi_\theta(a|s)$ , and its relationship with  $\pi(a|s)$  is shown below.

$$\pi(a|s) \approx \pi_\theta(a|s) = P(a|s, \theta). \tag{20}$$

$\pi_\theta(a|s)$  approximate to strategy  $\pi(a|s)$  using a function with  $\theta$  to present the probability of performing action  $a$  in state  $s$ . Then, the optimal strategy is constantly approximated by updating  $\theta$ .

Therefore, according to the guidelines of this function, we need to set an objective function so that policy function  $\pi_\theta(a|s)$  can optimize  $\theta$ . When  $\theta$  is determined, we can express the energy consumption of SFC as Equation (15).

$$J_E^\pi(\theta|s) = \sum_{\tau} E(\tau)P_\theta(\tau) = \mathbb{E}_{p \sim \pi(\cdot|s)} [E(\tau)], \tag{21}$$

where  $E(\tau)$  is the energy consumption of the network system at trajectory  $\tau$ . The agent needs to infer an approximately optimal solution from all possible combination schemes by expectation, so we define the expected energy consumption expectation as

$$J_E^\pi(\theta) = \mathbb{E}_{s \sim \pi(\cdot|S)} [J_E^\pi(\theta|s)]. \tag{22}$$

Similarly, the expectations that do not conform to the constraints are expressed as follows:

$$J_F^\pi(\theta) = \mathbb{E}_{s \sim \pi(\cdot|S)} [J_F^\pi(\theta|s)] \tag{23}$$

To sum up, the goal can be expressed as

$$\min J_E^\pi(\theta) \quad s.t. \quad J_F^\pi(\theta) \leq 0. \tag{24}$$

However, optimization problems with constraints are difficult to solve, so we assign penalty value  $C_i(s|a)$  to the cases that do not conform to the constraints; the expectation is

$J_C^\tau(\theta)$ , and the final objective function is defined as the sum of energy consumption and penalty values that do not conform to restrictions:

$$\bar{R}(\theta) = \min(J_E^\tau(\theta) + J_C^\tau(\theta)). \tag{25}$$

To reduce the value of the objective function, the method of gradient descent is adopted to optimize the function, and the gradient is obtained:

$$\begin{aligned} \nabla \bar{R}(\tau) &= \sum_{\tau} R(\tau) \cdot \nabla P_{\theta}(\tau) \\ &= \sum_{\tau} R(\tau) \cdot P_{\theta}(\tau) \frac{\nabla P_{\theta}(\tau)}{P_{\theta}(\tau)} \\ &= \sum_{\tau} R(\tau) \cdot \nabla \log P_{\theta}(\tau) \\ &= \mathbb{E}_{\tau \sim P_{\theta}(\tau)} \sum_{\tau} R(\tau) \cdot \nabla \log P_{\theta}(\tau) \\ &= \mathbb{E}_{(a) \sim \pi_a(\cdot|s)} \sum_{\tau} R(a|s) \cdot \nabla \log \pi_{\theta}(a|s), \end{aligned} \tag{26}$$

where  $R(\tau) = E(\tau) + \sum_i \lambda_i \cdot C_i(s|a)$ . However, this expected value cannot be calculated directly, so it is necessary to approximate the gradient by sampling. The expected value of sampling  $N$   $\tau$  can be expressed as

$$\begin{aligned} \mathbb{E}_{(a) \sim \pi_a(\cdot|s)} \sum_{\tau} R(a|s) \cdot \nabla \log \pi_{\theta}(a|s) &\approx \\ \frac{1}{N} \sum_{n=1}^N R(a_n|s_n) \cdot \nabla \log \pi_{\theta}(a_n|s_n). \end{aligned} \tag{27}$$

Some actions may never be sampled during the actual learning process, and it reduces the probability of a better deployment. We suppose the actions that state  $s$  can perform are  $a, b$ , and  $c$ , but only actions  $b$  or  $c$  are sampled. If these actions reduce the penalty value, we can know that the probability of each action being selected should rise according to  $\bar{R}(\tau)$ . But in this process, the best action  $a$  is not sampled, which leads to a decrease in its probability of being chosen. This is obviously problematic; we want the punishment to be able to judge the relative goodness of the action, so we need to introduce a baseline that depends on the state, and auxiliary network  $b_{\theta_v}(s)$  is needed to predict the penalty value based on the state. The gradient after its introduction is

$$\nabla \bar{R}(\tau) \approx \frac{1}{N} \sum_{n=1}^N [R(a_n|s_n) - b_{\theta_v}(s_n)] \cdot \nabla \log \pi_{\theta}(a_n|s_n). \tag{28}$$

### 5.3. Greedy Deployment

However, only using the above algorithm has two disadvantages: one is that random gradient descent is easy to fall into the saddle points; the other is that all cases cannot be sampled in the training process, and it is impossible to judge which method is the best solution.

For this purpose, we trained multiple models to avoid the situation where a single model could not escape the saddle point. In training, the probability of a single model falling into saddle points is relatively high, but training multiple models can reduce this probability to a certain extent. Even if all models fall into the saddle points, the saddle points approaching the optimal solution can be obtained eventually.

In addition, to alleviate the second problem raised, the prediction model is sampled  $n$  times after passing through the softmax layer, and the best action is greedily selected from it, instead of sampling the action once according to the action probability. The greedy algorithm's flow is shown in Algorithm 2.

---

**Algorithm 2 Greedy Choice Placement**


---

**Input:** Number of models  $N$ ; Number of samplings  $M$ ;

**Output:** Optimal scheme  $P$ 

```

1:  $temp, temp_m \leftarrow E(1)$ 
2: for  $i = 1, 2, \dots, N$  do
3:   for  $j = 1, 2, \dots, M$  do
4:     if  $temp < E(i)$  then
5:        $temp \leftarrow j$ th Sampling penalty of the  $i$ th model  $E(j)$ 
6:     end if
7:   if  $temp_m < temp$  then
8:      $temp_m \leftarrow temp$ 
9:   end if
10: end for
11: end for
12: return the placement  $P$ 

```

---

## 6. Experiment and Analysis

### 6.1. Experimental Setting

In the simulation experiment, the topology of the ground network uses the real NTT global network topology from the Topology Zoo [26]. There are 47 hosts in the network which links North America, Asia, Europe, and Australia, but there are 15 isolated hosts in the Middle East, Africa, and some small islands.

Referring to the network topology in the experiment of Li et al. [27], we simulated a satellite network composed of 58 satellites whose physical properties are shown in Table 2. Firstly, there is an orbit in the top layer with an angle of  $0^\circ$  from the equator, there are 3 GEO satellites in total. In addition, the mesosphere has 2 orbits with an angle of  $45^\circ$  from the equator, each with 5 MEO satellites. Furthermore, there are 5 orbits in the lowest layer, and the angle between the orbits and the equator is  $90^\circ$ ; there are 9 satellites in each orbit, and 3 LEO satellites are randomly selected to be introduced into the MEC. The communication delay between satellites is shown in Table 3.

**Table 2.** Running period, number and orbital inclination of different satellites.

Satellites	Running Period ( $^\circ$ /h)	Orbital Inclination ( $^\circ$ )	
GEO	15	$1 \times 3$	0
MEO	60	$2 \times 5$	45
LEO	210	$5 \times 9$	90

**Table 3.** Satellite-to-satellite and satellite-to-ground latency.

	GEO	MEO	LEO	SGs
GEO	-	86 ms	-	-
MEO	86 ms	66 ms	50 ms	-
LEO	-	50 ms	-	3 ms
SGs	-	-	3 ms	-

UAVs are characterized by versatility and high mobility and can provide communication services by installing a communication transceiver as an aerial communication platform [28,29]. On the other hand, UAVs can also be used as aerial hosts to realize various applications from cargo delivery to surveillance [30,31]. In this experiment, UAVs are used to assist ground networks and satellite networks in achieving full coverage of the network [32].

The resource situation of the servers is shown in Table 4. A host in the ground network is selected as the cloud server. Compared with other edge hosts, it has more computing and bandwidth resources.

**Table 4.** Edge computing Server Resource Settings.

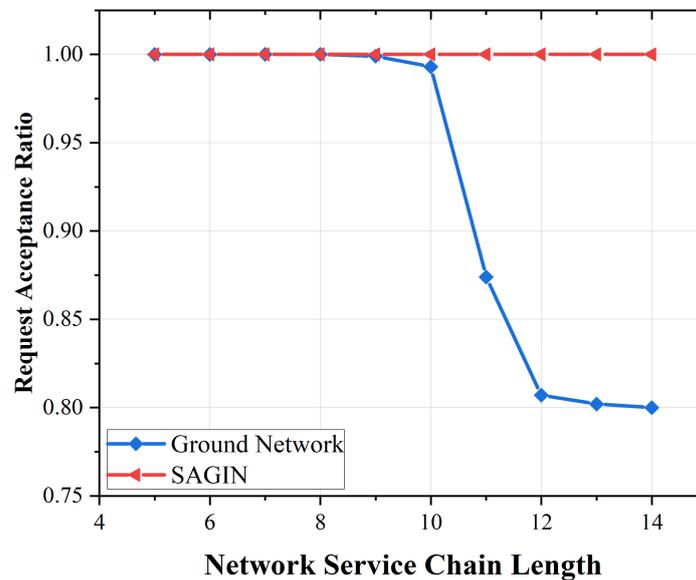
	Number	CPU	BW
MEC	47	[4, 6]	[400, 500]
LEO-MEC	3	2	[200, 300]

6.2. Comparative Experimental Results and Analysis

We design two experiments to verify the advantages of using SAGIN-MEC versus terrestrial edge cloud networks and the advantages of DRL-G versus other algorithms.

SAGIN’s use of satellites and aircraft for cable-free communication, as opposed to a ground-based network alone, provides a significant resource advantage that significantly improves the success rate of SFC deployments. Therefore, 15 isolated hosts in the ground network are eliminated, and the remaining 32 ground hosts are used for the experiment.

In total, 1000 SFCs are deployed in SAGIN-MEC and terrestrial edge cloud networks, respectively, and the comparison of their request acceptance rates is shown in Figure 7. After the chain length exceeds 8, the acceptance rate using only the ground network begins to decrease, while SAGIN-MEC is able to maintain the original acceptance level. In the case of successful deployment of SFC, the comparison of the average delay between them is shown in Figure 8. Therefore, the delay using SAGIN-MEC is always lower than that using only ground networks, with a difference of 6–15 ms. For this result, the reason is that the link that has the lowest latency can be automatically selected among terrestrial and non-terrestrial networks in SAGIN. At an altitude of 895 km, the coverage diameter of LEO is 3000 km. In this case, the delays of one-way and two-way communications to the ground are 3 ms and 6 ms, respectively, and the communication distance of the ground network is 1800 km at 6 ms. In SAGIN, DRL-G can choose the best solution for each situation.



**Figure 7.** The request acceptance ratio in SAGIN and terrestrial network.

Compared with terrestrial edge cloud networks, SAGIN-MEC can significantly reduce the network service delay and improve the request acceptance rate by taking advantage of the low-latency characteristic of LEO satellites when transmitting data packets at medium distances.

In SAGIN, compared with First-Fit (FF) [33], Greedy algorithm guided by First-Fit (F-G) and Policy Gradient algorithm (PG) [34], it can prove the effectiveness of the hybrid algorithm DRL-G. Among them, FF is the classical baseline algorithm and PG is the widely adopted deep reinforcement learning algorithm.

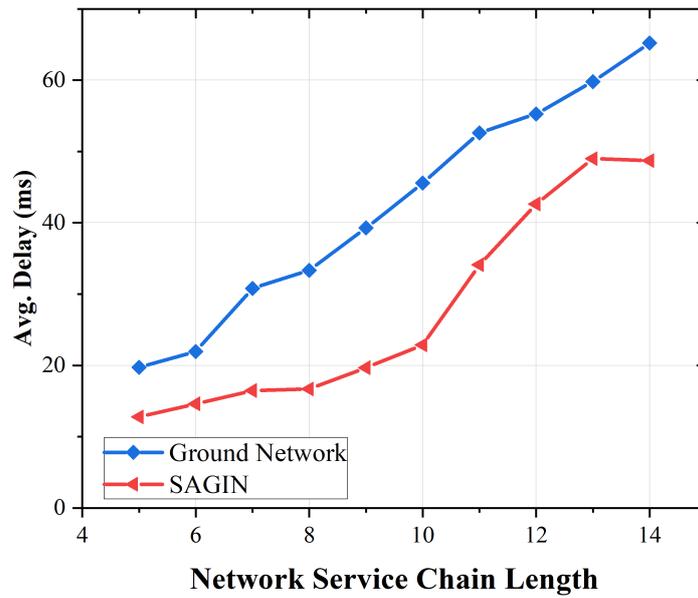


Figure 8. Average delay in SAGIN and terrestrial network.

With 1000 SFCs deployed using different algorithms, the request acceptance rate is shown in Figure 9. In the case of short chain length, DRL-G and PG are superior to the heuristic algorithm (FF and F-G). In the case of long-chain length and resource shortage, PG is slightly worse than the heuristic algorithm, while DRL-G is significantly better than other algorithms, which can improve the acceptance rate by up to 20% compared to the F\_G algorithm. The comparison results of energy consumption of different algorithms are shown in Figure 10. The average energy consumption of DRL-G is less than that of PG. However, the average energy consumption required by DRL-G is 6.6% higher than that of FF and F-G algorithms. When generating deployment policies, the DRL-G algorithm first ensures that the strict constraints on network resources imposed by VNF are met, and then works to reduce service energy consumption, while the FF and F-G algorithms optimize both resources and energy consumption. During the deployment process, DRL-G may sacrifice a certain amount of energy consumption to improve the deployment success rate.

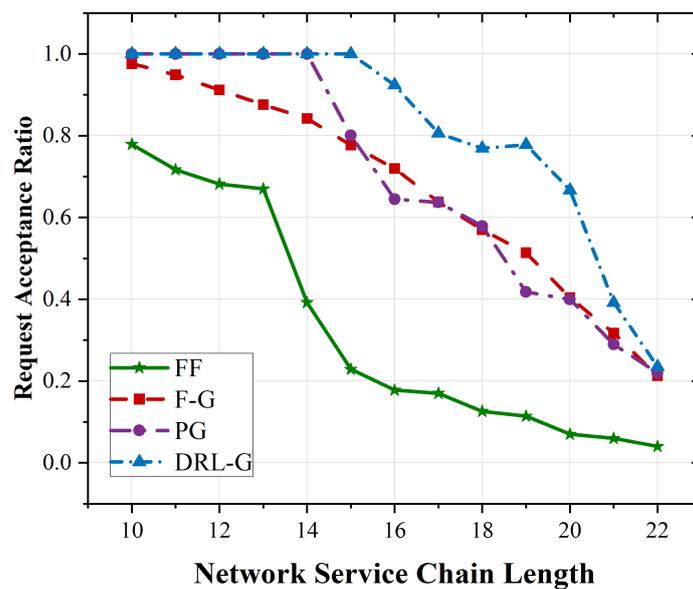


Figure 9. Comparison of average delay of different algorithms.

In order to further analyze the advantages of DRL-G in this environment and the reasons for the above results, we use different algorithms to deploy 100 SFCs and make a comparative analysis of the results. The number of SFCs exceeding the maximum delay limit is shown in Figure 11. In the deployment results of DRL-G, the number of SFCs exceeding the maximum allowable delay is less than ten, which is far lower than that of other algorithms, indicating that the proposed hybrid algorithm has a stable delay and can better meet the delay requirements of SFCs. Figure 12 shows the number of SFCs that do not meet resource constraints after deploying requests by different algorithms. Compared with other algorithms, DRL-G can better meet environmental resource constraints. For this, the reason is that other algorithms tend to use fewer hosts to save energy, but this approach easily fails to meet the resources required by the service, resulting in a decrease in the success rate of deployment. Although DRL-G increases the cost of deploying SFC, it can greatly improve the request acceptance ratio.

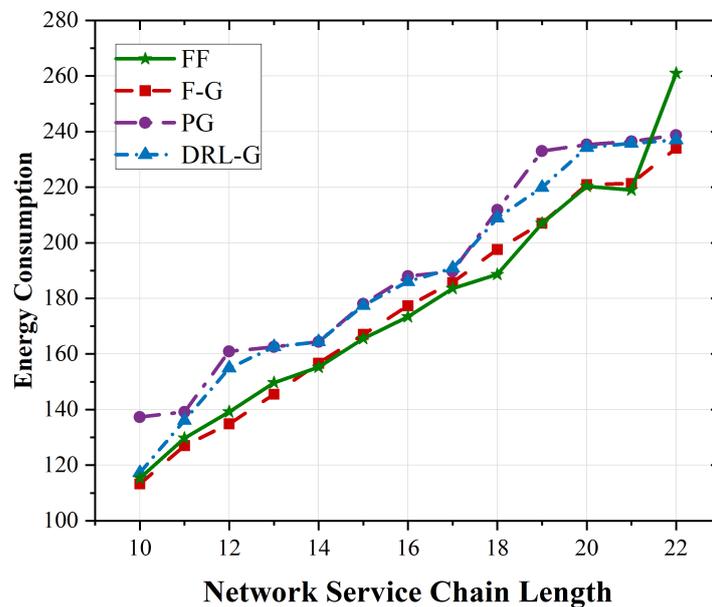


Figure 10. Comparison of energy consumption of different algorithms.

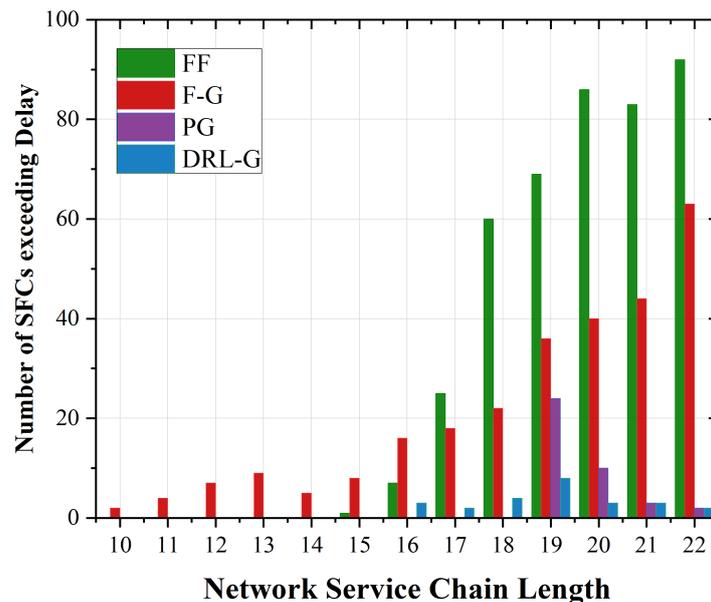
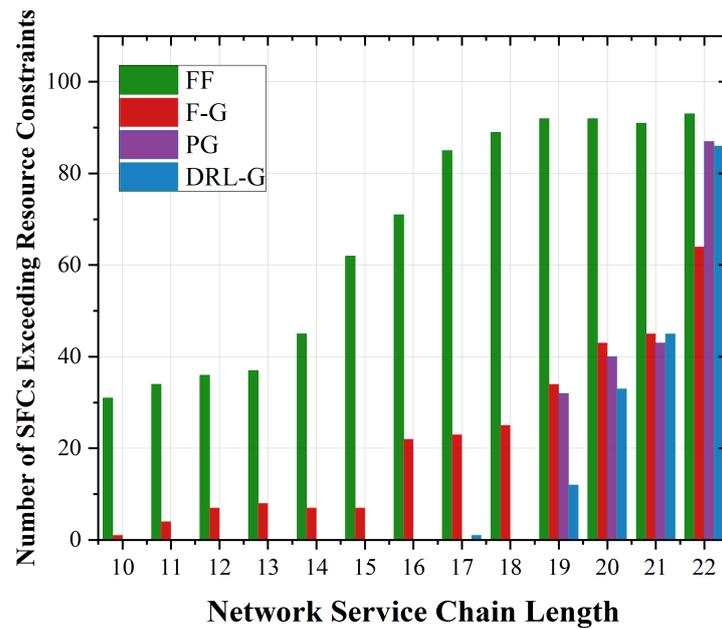


Figure 11. The number of SFCs exceeded the delay constraint with the different algorithms.



**Figure 12.** The number of SFCs exceeding energy constraints with different algorithms.

## 7. Conclusions

Overall, this article proposes SAGIN-MEC, a SAGIN structure for heterogeneous device management and resource allocation optimization in IIOT and IOA scenarios. The structure uses the SDN and NFV technology for distributed control of the whole network. It also introduces MECs in satellite, aircraft, and ground hosts near the destination to perform computing tasks so as to reduce service delay. Based on this structure, we design a hybrid algorithm DRL-G based on deep reinforcement learning and the heuristic algorithm to solve the resource allocation problem in SAGIN. Several simulation experiments show that service delay in SAGIN-MEC can be reduced by 6–15 ms, and DRL-G significantly improves the success rate and delay. The next phase will focus on computing resource scheduling in SAGIN.

**Author Contributions:** X.F.: Conceptualization, Investigation, Methodology, Software, Formal analysis, Writing—Original draft preparation. M.H.: Writing—review & editing, Supervision, Project administration, Funding acquisition. L.Z.: Writing—review & editing, Supervision, Project administration, Funding acquisition. Y.S.: Data curation, Visualization, Writing—review & editing. R.P.: Data curation, Software. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by Science and Technology Major Project of Henan Province, China (NO. 221100210900-03) & The Scientific and technological project in Henan Province (Project No. 232102210154) & Pre-research Project of Songshan Laboratory (Project No. YYJC022022001).

**Data Availability Statement:** The original contributions presented in the study are included in the article material, further inquiries can be directed to the corresponding author.

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

## Abbreviations

The following abbreviations are used in this manuscript:

SAGIN	Space–Air–Ground Integrated Network
QoS	Quality of Service
SDN	Software Defined Network
NFV	Network Function Virtualization
MEC	Mobile Edge Computing
DRL	Deep Reinforcement Learning

LEO	Low Earth Orbit
HAP	High-Altitude Platforms
SFC	Service Function Chain
VNF	Virtual Network Function
IIOT	Industrial Internet of Things
IOA	Internet of Agriculture
SDNC	SDN controller
GEO	Geostationary Earth Orbit
UAP	Unmanned Aerial Vehicle
MaOP	Multi-objective Optimization Problem
LSTM	Long Short-Term Memory Network
RNN	Recurrent Neural Network

## References

1. Ray, P.P. A review on 6G for space-air-ground integrated network: Key enablers, open challenges, and future direction. *J. King Saud-Univ.-Comput. Inf. Sci.* **2022**, *34*, 6949–6976. [\[CrossRef\]](#)
2. Cheng, N.; He, J.; Yin, Z.; Zhou, C.; Wu, H.; Lyu, F.; Zhou, H.; Shen, X. 6G service-oriented space-air-ground integrated network: A survey. *Chin. J. Aeronaut.* **2022**, *35*, 1–18. [\[CrossRef\]](#)
3. Hou, X.; Wang, J.; Fang, Z.; Ren, Y.; Chen, K.C.; Hanzo, L. Edge Intelligence for Mission-Critical 6G Services in Space-Air-Ground Integrated Networks. *IEEE Netw.* **2022**, *36*, 181–189. [\[CrossRef\]](#)
4. Chen, C.; Liao, Z.; Ju, Y.; He, C.; Yu, K.; Wan, S. Hierarchical Domain-Based Multicontroller Deployment Strategy in SDN-Enabled Space–Air–Ground Integrated Network. *IEEE Trans. Aerosp. Electron. Syst.* **2022**, *58*, 4864–4879. [\[CrossRef\]](#)
5. Liao, Z.; Chen, C.; Ju, Y.; He, C.; Jiang, J.; Pei, Q. Multi-Controller Deployment in SDN-Enabled 6G Space–Air–Ground Integrated Network. *Remote. Sens.* **2022**, *14*, 1076. [\[CrossRef\]](#)
6. Alhussain, O.; Zhuang, W. Dynamic Topology Design of NFV-Enabled Services Using Deep Reinforcement Learning. *IEEE Trans. Cogn. Commun. Netw.* **2022**, *8*, 1228–1238. [\[CrossRef\]](#)
7. Khatiri, A.; Mirjalily, G. A cost-efficient, load-balanced and fragmentation-aware approach for deployment of VNF service chains in Elastic Optical Networks. *Comput. Commun.* **2022**, *188*, 156–166. [\[CrossRef\]](#)
8. Cisneros, J.C.; Yangui, S.; Hernandez, S.E.P.; Drira, K. A Survey on Distributed NFV Multi-Domain Orchestration from an Algorithmic Functional Perspective. *IEEE Commun. Mag.* **2022**, *60*, 60–65. [\[CrossRef\]](#)
9. Kota, S.; Giambene, G.; Kim, S. Satellite component of NGN: Integrated and hybrid networks: SATELLITE COMPONENT OF NGN. *Int. J. Satell. Commun. Netw.* **2011**, *29*, 191–208. [\[CrossRef\]](#)
10. Wang, Y.; Zhang, J.; Zhang, X.; Wang, P.; Liu, L. A Computation Offloading Strategy in Satellite Terrestrial Networks with Double Edge Computing. In Proceedings of the 2018 IEEE International Conference on Communication Systems (ICCS), Chengdu, China, 19–21 December 2018; pp. 450–455. [\[CrossRef\]](#)
11. Giambene, G.; Kota, S.; Pillai, P. Satellite-5G Integration: A Network Perspective. *IEEE Netw.* **2018**, *32*, 25–31. [\[CrossRef\]](#)
12. Cao, B.; Zhang, J.; Liu, X.; Sun, Z.; Cao, W.; Nowak, R.M.; Lv, Z. Edge–Cloud Resource Scheduling in Space–Air–Ground-Integrated Networks for Internet of Vehicles. *IEEE Internet Things J.* **2022**, *9*, 5765–5772. [\[CrossRef\]](#)
13. Giannopoulos, A.; Spantideas, S.; Capsalis, N.; Gkonis, P.; Karkazis, P.; Sarakis, L.; Trakadas, P.; Capsalis, C. WIP: Demand-Driven Power Allocation in Wireless Networks with Deep Q-Learning. In Proceedings of the 2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Pisa, Italy, 7–11 June 2021; pp. 248–251. [\[CrossRef\]](#)
14. Lyu, Y.; Liu, Z.; Fan, R.; Zhan, C.; Hu, H.; An, J. Optimal Computation Offloading in Collaborative LEO-IoT Enabled MEC: A Multiagent Deep Reinforcement Learning Approach. *IEEE Trans. Green Commun. Netw.* **2023**, *7*, 996–1011. [\[CrossRef\]](#)
15. Trakadas, P.; Masip-Bruin, X.; Facca, F.M.; Spantideas, S.T.; Giannopoulos, A.E.; Kapsalis, N.C.; Martins, R.; Bosani, E.; Ramon, J.; Prats, R.G.; et al. A Reference Architecture for Cloud–Edge Meta-Operating Systems Enabling Cross-Domain, Data-Intensive, ML-Assisted Applications: Architectural Overview and Key Concepts. *Sensors* **2022**, *22*, 9003. [\[CrossRef\]](#) [\[PubMed\]](#)
16. Rost, M.; Schmid, S. On the Hardness and Inapproximability of Virtual Network Embeddings. *IEEE/Acm Trans. Netw.* **2020**, *28*, 791–803. [\[CrossRef\]](#)
17. Zhou, S.; Wang, G.; Zhang, S.; Niu, Z.; Shen, X.S. Bidirectional Mission Offloading for Agile Space-Air-Ground Integrated Networks. *IEEE Wirel. Commun.* **2019**, *26*, 38–45. [\[CrossRef\]](#)
18. Li, J.; Shi, W.; Wu, H.; Zhang, S.; Shen, X. Cost-Aware Dynamic SFC Mapping and Scheduling in SDN/NFV-Enabled Space–Air–Ground-Integrated Networks for Internet of Vehicles. *IEEE Internet Things J.* **2022**, *9*, 5824–5838. [\[CrossRef\]](#)
19. Zhang, P.; Zhang, Y.; Kumar, N.; Guizani, M. Dynamic SFC Embedding Algorithm Assisted by Federated Learning in Space-Air-Ground Integrated Network Resource Allocation Scenario. *IEEE Internet Things J.* **2022**, *10*, 9308–9318. [\[CrossRef\]](#)
20. Li, G.; Zhou, H.; Feng, B.; Li, G.; Xu, Q. Horizontal-based orchestration for multi-domain SFC in SDN/NFV-enabled satellite/terrestrial networks. *China Commun.* **2018**, *15*, 77–91. [\[CrossRef\]](#)
21. Gao, X.; Liu, R.; Kaushik, A. Service Chaining Placement Based on Satellite Mission Planning in Ground Station Networks. *IEEE Trans. Netw. Serv. Manag.* **2021**, *18*, 3049–3063. [\[CrossRef\]](#)

22. Han, C.; Li, X.; Ji, H.; Zhang, H. Adaptive Online Service Function Chain Deployment in Large-scale LEO Satellite Networks. In Proceedings of the 2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Toronto, ON, Canada, 5–8 September 2023; pp. 1–6. [\[CrossRef\]](#)
23. Qin, X.; Ma, T.; Tang, Z.; Zhang, X.; Zhou, H.; Zhao, L. Service-Aware Resource Orchestration in Ultra-Dense LEO Satellite-Terrestrial Integrated 6G: A Service Function Chain Approach. *Trans. Wireless. Comm.* **2023**, *22*, 6003–6017. [\[CrossRef\]](#)
24. He, J.; Cheng, N.; Yin, Z.; Zhou, H.; Xu, W.; Peng, H.; Zhou, C.; Zhang, R. Service-Oriented Resource Allocation in SDN Enabled LEO Satellite Networks. In Proceedings of the 2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Toronto, ON, Canada, 5–8 September 2023; pp. 1–6. [\[CrossRef\]](#)
25. Deudon, M.; Cournot, P.; Lacoste, A.; Adulyasak, Y.; Rousseau, L.M. Learning Heuristics for the TSP by Policy Gradient. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*; van Hoeve, W.J., Ed.; Series Title: Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2018; Volume 10848, pp. 170–181. [\[CrossRef\]](#)
26. Knight, S.; Nguyen, H.X.; Falkner, N.; Bowden, R.; Roughan, M. The Internet Topology Zoo. *IEEE J. Sel. Areas Commun.* **2011**, *29*, 1765–1775. [\[CrossRef\]](#)
27. Li, T.; Zhou, H.; Luo, H.; Yu, S. SERvICE: A Software Defined Framework for Integrated Space-Terrestrial Satellite Communication. *IEEE Trans. Mob. Comput.* **2018**, *17*, 703–716. [\[CrossRef\]](#)
28. Bai, L.; Liu, J.; Wang, J.; Han, R.; Choi, J. Data Aggregation in UAV-Aided Random Access for Internet of Vehicles. *IEEE Internet Things J.* **2022**, *9*, 5755–5764. [\[CrossRef\]](#)
29. Nguyen, T.V.; Le, H.D.; Pham, A.T. On the Design of RIS-UAV Relay-Assisted Hybrid FSO/RF Satellite-Aerial-Ground Integrated Network. *IEEE Trans. Aerosp. Electron. Syst.* **2022**, *59*, 757–771. [\[CrossRef\]](#)
30. Chen, Q.; Meng, W.; Li, S.; Li, C.; Chen, H.H. Civil Aircrafts Augmented Space–Air–Ground-Integrated Vehicular Networks: Motivation, Breakthrough, and Challenges. *IEEE Internet Things J.* **2022**, *9*, 5670–5683. [\[CrossRef\]](#)
31. Chen, Q.; Meng, W.; Han, S.; Li, C.; Chen, H.H. Effect of Intelligent Multi-Association in Civil Aircraft-Augmented SAGIN. *IEEE Trans. Cogn. Commun. Netw.* **2023**, *9*, 223–238. [\[CrossRef\]](#)
32. Cao, X.; Yang, B.; Yuen, C.; Han, Z. HAP-Reserved Communications in Space-Air-Ground Integrated Networks. *IEEE Trans. Veh. Technol.* **2021**, *70*, 8286–8291. [\[CrossRef\]](#)
33. Békési, J.; Dósa, G.; Galambos, G. A first Fit type algorithm for the coupled task scheduling problem with unit execution time and two exact delays. *Eur. J. Oper. Res.* **2022**, *297*, 844–852. [\[CrossRef\]](#)
34. Li, G.; Zhou, H.; Feng, B.; Zhang, Y.; Yu, S. Efficient Provision of Service Function Chains in Overlay Networks Using Reinforcement Learning. *IEEE Trans. Cloud Comput.* **2022**, *10*, 383–395. [\[CrossRef\]](#)

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.