sensors

MDPI

Article

A Resource and Task Scheduling Based Multi-Objective Optimization Model and Algorithms in Elastic Optical Networks

Yuping Wang [1,*], Qingdong Yang [2] and Xiaofang Guo [1,*]

[1] School of Science, Xi'an Technological University, Xi'an 710021, China
[2] School of Computer Science and Technology, Xidian University, Xi'an 710071, China
* Correspondence: ywang@xidian.edu.cn (Y.W.); guoxiaofang@xatu.edu.cn (X.G.)

Abstract: The elastic optical network (EON) adopting virtual network function (VNF) is a new type of network, in which the routing, spectrum, and data center allocation are key and challenging problems, and solving these three problems simultaneously can not only improve the network efficiency for network providers, but also let users obtain better service. However, few existing works handle these three problems simultaneously. To tackle the three problems simultaneously, given a set of network function chains (i.e., a set of tasks), we set up a new multi-objective optimization model in which the total length of paths for all tasks is minimized, the totally occupied spectrums are minimized, and the loads on all data centers are most balanced, simultaneously. To solve the model, we design two new evolutionary algorithms. The experiments are conducted on 16 cases of 4 widely used types of networks, and the results indicate that the proposed model and algorithms are effective.

Keywords: elastic optical networks; data center allocation; routing; spectrum allocation; virtual network function; multi-objective optimization; evolutionary algorithm

Citation: Wang, Y.; Yang, Q.; Guo, X. A Resource and Task Scheduling Based Multi-Objective Optimization Model and Algorithms in Elastic Optical Networks. *Sensors* **2022**, *22*, 9579. https://doi.org/10.3390/s22249579

Academic Editor: Christian Haubelt

Received: 2 November 2022
Accepted: 2 December 2022
Published: 7 December 2022

1. Introduction

With the development of internet and network technology, a wavelength division multiplexing (WDM) network can not satisfy the requirement of real world problems due to its spectrum allocation mode of fixed grid and low spectrum utilization. Elastic optical network (EON) is a potential direction for developing the future network due to flexible spectrum allocation and high spectrum utilization. However, the routing and spectrum allocation (RSA) problem in EON is more complex. RSA has become one of the hottest and key problems in EON [1]. It needs to handle multiple issues simultaneously (e.g., select the best route, use the least spectrum resource, make the communication efficient, etc.). Although many researchers have studied these kinds of problems, most of them only can handle one or two issues simultaneously. For example, Ref. [2] proposed an adaptive scheme which only tackles the routing and spectrum allocation problems based on reducing the relative cost of network flow and the probability of communication block. Ref. [3] proposed a RSA strategy which only improves the quality of transmission (QoT) based on both the link state advertisement (LSA) and the risk reduction of quality of transmission (QoT). To improve the communication quality by tackling unbalanced distribution of network flows in EON, Ref. [4] proposed two algorithms based on the OTTM model to improve bandwidth efficiency. To handle routing and spectrum issues, Ref. [5] proposed an adaptive RSA algorithm which tried to select the routing of the lowest relative cost and design the efficient spectrum scheme by estimating both the transient effect of call admission on a given set of routing/spectrum and the relative cost of reaching the forward destination of the calls. To handle the spectrum problem, Ref. [6] set up an integer linear programming model for spectrum allocation problems by analyzing the characteristic of network data flow of three kinds of elastic optical networks, and proposed two meta

heuristic algorithms based on particle swarm optimization and tabu search. However, these works did not consider how to tackle more than two issues simultaneously and cannot complete a series of function service chains (FSC).

In addition, with the fast increase of the number of network users, it becomes more difficult to handle multiple issues simultaneously, and it is required to continuously update the network hardware with the increase of tasks and users. However, the hardware update speed can not catch the speed of increase of tasks and users. Therefore, it is impossible to solve this problem by only updating the hardware. The virtual network function (VNF) is to realize the function of special hardware by using software without the need for updating hardware. VNF can increase the number of tasks/users and enhance the network resource utilization without increasing network hardware. Thus, VNF is a potential technique in the environment of fast increase of users/tasks for elastic optical networks. Moreover, the elastic optical network with VNF provides an efficient tool to possibly handle multiple issues simultaneously in the RSA problem and complete a series of function service chains (FSC). Each FSC (can be seen as a task) consists of a sequential functions (e.g., data encryption, data decryption, deep packet inspection, data monitoring, etc.) to be successively completed. However, this flexibility and adaptability will result in more complexity for setting up models and designing algorithms. The existing models and algorithms for elastic optical networks can only handle one or two issues simultaneously, and cannot tackle multiple (more than two) issues simultaneously. In fact, the current models and algorithms for elastic optical networks face many challenges. One of the most difficult challenges is how to tackle the following four problems on EON simultaneously:

(1) How to reasonably make the routing (select reasonable path for each task/FSC) such that data can be quickly transmitted; (2) how to efficiently allocate spectrums such that the spectrum resource is conserved the most; (3) how to make the loads most balanced among the data centers such that the network communication is more efficient and stable; and (4) how to deploy VNF efficiently.

2. Related Works

Currently, there have been some research works in this field. For example, in order to handle two problems (i.e., reduce the cost and improve the quality of service (QoS)), Ref. [7] proposed an integrated algorithm based on VNF migration strategy, which aims at reducing the income loss of network service provider due to both data lost during the transmission and decreasing of quality of service (QoS), but it did not consider how to balance the loads among the data centers. To solve VNF and physical function allocation problems in network and cloud environments, Ref. [8] proposed a heuristic algorithm based on a self-defined greedy strategy to improve the performance and ability of feature decomposition method, but it did not consider the spectrum allocation and load balance. Ref. [9] surveyed the current research progress on VNF allocation problems. Ref. [10] proposed a new virtualization method on network functions, but it did not consider the load balance and routing. In order to balance the load of the data center nodes, Ref. [11] proposed a highly efficient switch migration (HESM) method for controlling load balance via minimizing migration cost, but it did not consider the routing and spectrum allocation. To increase spectral efficiency for the routing and spectrum assignment (RSA) in elastic optical networks (EON), Ref. [12] proposed a novel technique for resource planning and consumption estimation by the optimal utilization of already existing resources, but it did not consider the load balance. In order to enhance the efficiency of routing and resource allocation, Ref. [13] proposed a prediction-based dynamic slicing mechanism (PDSM) for heterogeneous elastic fiber wireless networks, but it also did not consider the load balance. To obtain a better resource allocation scheme, Ref. [14] proposed a novel constrained deep reinforcement learning algorithm for the resource allocation problem. Aiming at improving the efficiency of both wireless resource management and computation resource management within network slices in edge computing systems, Ref. [15] proposed an efficient approximation algorithm based on the game theory, which minimizes the

completion time of task execution in the system. These algorithms also did not consider the load balance. Ref. [16] investigated static routing, VNF, and spectrum allocation (RMSTA) of advanced reservation (AR) requests in elastic optical networks. They formulated a two-objective optimization model by minimizing the occupied spectrum and time resources, and proposed a heuristic algorithm based on three sorting strategies. However, this study focused only on sorting requests and neglected the effect of routing and spectrum allocation on the model and algorithm. Furthermore, solving the formulated multi-objective problem by minimizing the weighted sum of two objectives cannot obtain a set of solutions, which failed to satisfy users with different preferences. Ref. [17] investigated the routing and content replication placement in elastic optical networks. They built a mixed integer linear programming model and proposed a heuristic optimization method. However, this study focused on content replication and neglected the impact of routing policies on designing algorithms in large-scale networks and short-term re-optimization scenarios. Ref. [18] proposed an ILP model that takes into account the multicast routing, spectrum, and transceivers for multicast requests, with the goal of minimizing the spectrum consumption and transceivers required for the subtree in an elastic optical data center network (EODCNs). However, this study did not give a method to determine the weight coefficients for each optimization objective in the ILP model and did not consider the load balance. Ref. [19] used a physical slicing technique that supports an end-to-end optical path on a combination of multiple discontinuous spectral slots in order to enhance the spectrum allocation, and proposed a scheme to determine the splitting position and required number of slots for each slice component in an EON network. They built a mixed integer linear programming (MILP) model by making the trade-off among the slicers availability, spectrum utilization, and bandwidth blocking probability. Although the physical slicing technology has improved the spectrum utilization rate in EON, this study did not consider the network load balance. Ref. [20] studied the allocation of data center and spectrum for VNF service chain in inter-datacenter elastic optical networks. To minimize the total network cost and balance the utilization of resources, the authors set up a model with three optimization objectives first, and then transformed it into a single-objective optimization model by using the weight sum method. Finally, they proposed a heuristic algorithm to reduce the congestion in both data centers and links. However, this work ignored the communication quality (e.g., quality of service). To deploy VNF effectively, Ref. [21] proposed a deep reinforced learning based algorithm to minimize the spectrum utilization ratio and the number of deployed VNFs jointly. A trade-off between spectrum utilization ratio and number of deployed VNFs was made by assigning a weight to each of the objectives. However, this method did not carefully take into account both routing and load balance of data centers. Instead, it only used a simple heuristic strategy to make the routing.

By summary, the existing works mainly focus on tackling only one or two issues of the four aforementioned issues. There have been few works tackling the four problems above simultaneously. To overcome this shortcoming, the main contributions in this paper are as follows: (1) We set up a new multi-objective model for solving all these problems simultaneously, i.e., select reasonable routes for each task such that data can be quickly transmitted; efficiently allocate spectrums such that the spectrum resource is conserved the most; make the loads most balanced among the data centers such that the network communication is more efficient and stable; and deploy VNF efficiently. (2) We propose two new evolutionary algorithms for solving this model. (3) We conduct a lot of experiments (including 16 cases of 4 types of widely used networks) to test the effectiveness and efficiency of the proposed model and algorithms.

The rest of the paper is arranged as follows: In Section 3, we first briefly introduce the problem considered in this paper, and then set up a new optimization model for the problem. Finally, we design two new evolutionary algorithms for the model. In Section 4, we conduct the experiments and conduct the analysis of experimental results. In Section 5, we make the discussion on the paper and point out the future work. Conclusions are made in Section 6.

3. Problem Description, Optimization Model, and Algorithm

3.1. Problem Description

A scheduling problem of the elastic optical network using virtual network functions can be described as follows: The elastic optical network using virtual network functions can be represented by an undirected graph $G(V, E)$, where the set $V = \{v_1, v_2, \cdots, v_{N_v}\}$ of the vertexes in the graph represents the set of nodes of the network, the set of nodes of data centers is denoted by $V_D = \{v_1, v_2, \cdots, v_{N_d}\}$, and the set $E = \{L_{ij} | v_i, v_j \in V\}$ of edges of graphs represents the set of links of networks. The set of tasks is denoted by $\text{FSC} = \{r_1, r_2, \cdots, r_k, \cdots, r_{N_R}\}$, where each task is a function service chain (FSC), and the k-th function service chain (also called the k-th task) is represented as $r_k = (s_k, d_k, b_k, t_k)$, where s_k is source node, d_k is the destination node, t_k is the function sequence $t_k = \{t_k^1, t_k^2, \cdots, t_k^i, \cdots, t_k^{F_k}\}$ required to be realized sequentially by task r_k, where $t_k^i \in \{1, 2, \cdots, N_T\}$ is the index of the i-th virtual network function to be realized by the k-th task r_k (for convenience, the virtual network function is briefly called the function in the following). t_k^i are different each other in t_k. F_k is the number of the functions contained in the k-th FSC. N_T is the total number of all different functions in all N_R tasks. b_k is the number of slots to be occupied by the k-th task r_k. N_F is the maximal number of slots which are available in the path.

The problem considered in this paper is to tackle the following issues: (1) how to select a proper path from the source node to the destination node for each task such that the total length of all paths is minimized; (2) how to assign the slots in the selected path for each task such that the slot assumption is minimized; (3) how to make the loads most balanced among the data centers such that the network communication is more efficient and stable; and (4) how to deploy VNF efficiently.

3.2. A Multi-Objective Optimization Model

3.2.1. Determining Variables for the Model

A. Path variables:

Let $P_k = (s_k, x_k^1, x_k^2, \cdots, x_k^{l_k}, d_k)$ denote the path to be chosen for the task r_k, where $x_k^i \in \{1, 2, \cdots, N_v\}$, x_k^1 is the first node of the path to reach from the start node s_k, and x_k^i is the i-th node of the path to reach from the start node s_k for $i = 1, 2, \cdots, l_k$. Finally, node $x_k^{l_k}$ directly reaches the destination node d_k. That is, path P_k is as follows:

$$s_k \to x_k^1 \to x_k^2 \to \cdots \to x_k^{l_k} \to d_k$$

and has to pass through at lest one data center node. For example, path $P_k = (5, 3, 6, 1, 8, 7, 2)$ for task r_k means that the source node is node 5; then, the first node to reach from source node 5 is node 3, and the second node to reach from source node 5 is node 6. Similarly, the later nodes in the path are sequentially node 1, node 8, node 7, and finally to the destination node 2.

Let $x_k = (x_k^1, x_k^2, \cdots, x_k^{l_k})$ denote path variables, and let V_{v_i} denote the set of nodes connected to node v_i in the network. Then, $x_k^1 \in V_{s_k}, x_k^2 \in V_{x_k^1}, x_k^3 \in V_{x_k^2}, \cdots, x_k^{l_k} \in V_{x_k^{l_k-1}}, d_k \in V_{x_k^{l_k}}$, and $D_c \cap \{s_k, x_k^1, x_k^2, \cdots, x_k^{l_k}, d_k\} \neq \emptyset$, where D_c is the set of nodes of data centers.

In the following, we shall design the following Algorithm 1 to look for such a path

$$P_k = (s_k, x_k^1, \cdots, x_k^{l_k}, d_k)$$

in which at least one node is a data center node, and to determine the path variables.

Algorithm 1 Path search algorithm

Step 1: Select x_k^1. For s_k, if V_{s_k} contains any data center node, randomly select one data center node as x_k^1; otherwise, randomly select a node in V_{s_k} as x_k^1, and let $i = 1$.

Step 2: Select x_k^{i+1} in $V_k^i / \{s_k, x_k^1, \cdots, x_k^i\}$ according to the following two cases, where V_k^i represents $V_{x_k^i}$ for notation convenience:
Case I: $\{s_k, x_k^1, \cdots, x_k^i\}$ contains a data center node. If $d_k \in V_k^i$, select d_k as x_k^{i+1}; otherwise, randomly select a node in V_k^i as x_k^{i+1}.
Case II: $\{s_k, x_k^1, \cdots, x_k^i\}$ does not contain any data center node. If V_k^i contains any data center node, randomly select one data center node in V_k^i as x_k^{i+1}, let $i = i + 1$, go to step 2; otherwise, randomly select one node in V_k^i as x_k^{i+1}, go to step 3.

Step 3: If $x_k^{i+1} \neq d_k$, let $i = i + 1$, go to step 2 (the current path neither contains any data center node nor reaches d_k), otherwise, go to step 4 (the current path does not contain any data center node but reaches d_k. This is not a feasible path, and we have to select another path).

Step 4: Re-select the nodes after node x_k^i until selecting node d_k by using the previous steps. If the new path is still an infeasible path (containing no data center node), re-select nodes after x_k^{i-1} until selecting d_k. Repeat this process until one feasible path is found.

B. Slot variables:

Let $y_k = (y_k^1, y_k^2, \cdots, y_k^{N_F})$ denote the slot assignment on path P_k for task r_k, where each y_k^i is a boolean variable. $y_k^i = 1$ represents that task r_k uses the i-th slot of each edge of path P_k, and $y_k^i = 0$ means that task r_k does not use the i-th slot on path P_k. Let U_k represent the set of tasks whose paths share a common link with path y_k, i.e.,

$$U_k = \{v \mid P_k \cap P_v \neq \emptyset, \ \forall 1 \leq v \leq N_R\}.$$

y_k must satisfy the following conditions: (1) The number of slots used in slot assignment y_k should be equal to the number of slots required by task r_k; (2) For the same slot, it should be used by at most one task. That is,

$$\begin{cases} \sum_{i=1}^{N_F} y_k^i = b_k, k = 1, 2, \cdots, N_R \\ \sum\limits_{v \in U_k} y_v^i \leq 1, i = 1, 2, \cdots, N_F \end{cases} \tag{1}$$

C. Function variables:

Suppose the data center nodes passed successively through by P_k are:

$$V_{k_1}, V_{k_2}, \cdots, V_{k_{i_k}}$$

The numbers of functions in r_k assigned to data center nodes $V_{k_1}, V_{k_2}, \cdots, V_{k_{i_k}}$ on path P_k are

$$z_k = (z_k^1, z_k^2, \cdots, z_k^{i_k}),$$

respectively, where z_k^j is the number of functions assigned to data center V_{k_j} to be realized, and the sequence of these functions assigned to the data center nodes is the same as that in r_k. We call z_k the function variables, which should satisfy the following conditions: The total number of assigned functions should be equal to the number of functions required to be realized in r_k, i.e.,

$$\sum_{j=1}^{i_k} z_k^j = F_k, k = 1, \cdots, N_R$$

where F_k is the number of functions required to be realized by r_k.

3.2.2. Objective Function Determination

A. Minimize the total length of all paths:

Note that path P_k corresponding to task r_k has $(l_k + 1)$ edges, and its length is $(l_k + 1)$. Then, the total length of all paths is $\sum_{k=1}^{N_R}(l_k + 1)$.

Thus, the first objective is to minimize the following total length:

$$f_1(x_1, \cdots, x_{N_R}) = \sum_{k=1}^{N_R}(l_k + 1)$$

Denote $x = (x_1, \cdots, x_{N_R})$. Then, we have $f_1(x) = \sum_{k=1}^{N_R}(l_k + 1)$.

B. Minimize the total number of slots used:

The number of slots in each link of path P_k for task r_k is b_k, and the number of links of path P_k is $l_k^{'}$ (excluding the shared common links and the shared links are not computed repeatedly). Thus, the total number of slots used by all tasks is

$$f_2(x, y_1, y_2, \cdots, y_{N_R}) = \sum_{k=1}^{N_R} b_k * l_k^{'}$$

Denote $y = (y_1, \cdots, y_{N_R})$. Then, the second objective is to minimize the following function $f_2(x, y) = \sum_{k=1}^{N_R} b_k * l_k^{'}$.

C. Optimize the balance of loads on all data center nodes:

To optimize the balance of loads on all data center nodes, the number of functions of all N_R tasks should be assigned to these data center nodes as equally as possible. The index set of the data center nodes passed through by path P_k is denoted by $\{k_1, k_2, \cdots, k_{i_k}\}$. Let

$$\widetilde{D_c} = \bigcup_{k=1}^{N_R} \{k_1, k_2, \cdots, k_{i_k}\}$$

denote the index set of data center nodes used by all tasks.

For $\forall c \in \widetilde{D_c}$, the number of functions handled by data center node c is

$$q_c(z_1, \cdots, z_{N_R}) = \sum_{k \in \widetilde{D_c}} z_k^c$$

The standard deviation of the numbers of functions handled by all data center nodes is

$$f_3(x, y, z_1, z_2, \cdots, z_{N_R}) = \sqrt{\frac{1}{|\widetilde{D_c}|} \sum_{c \in \widetilde{D_c}} [q_c(z_1, \cdots, z_{N_R}) - \frac{N_T}{|\widetilde{D_c}|}]^2}$$

Denote $z = (z_1, \cdots, z_{N_R})$. Then, optimizing the balance of load of data centers is equal to minimizing the following objective:

$$f_3(x, y, z) = \sqrt{\frac{1}{|\widetilde{D_c}|} \sum_{c \in \widetilde{D_c}} [q_c(z) - \frac{N_T}{|\widetilde{D_c}|}]^2}$$

Now our goal is to minimize the above three objectives simultaneously, i.e.,

$$\min \{f_1(x), f_2(x, y), f_3(x, y, z)\}$$

3.2.3. Constraints

A. The path variable constraints:

Path P_k selected for task r_k should pass through at least one data center node, i.e.,

$$D_c \bigcap \{s_k, x_k^1, x_k^2, \cdots, x_k^{l_k}, d_k\} \neq \emptyset, k = 1, \cdots, N_R$$

B. Slot constraints:

$$\sum_{i=1}^{N_F} y_k^i = b_k, k = 1, \cdots, N_R$$

$$\sum_{v \in U_k} y_v^i \leq 1, i = 1, 2, \cdots, N_F$$

C. Function assignment constraints:

$$\sum_{j=1}^{i_k} z_k^j = F_k, k = 1, \cdots, N_R$$

3.2.4. Multi-Objective Optimization Model

By summary, we can set up the following multi-objective optimization model:

$$\begin{cases} \min \{f_1(x), f_2(x,y), f_3(x,y,z)\} \\ s.t. \\ (1) D_c \bigcap \{s_k, x_k^1, x_k^2, \cdots, x_k^{l_k}, d_k\} \neq \emptyset, k = 1, \cdots, N_R \\ (2) \sum_{i=1}^{N_F} y_k^i = b_k, k = 1, \cdots, N_R \\ (3) \sum\limits_{v \in U_k} y_v^i \leq 1, i = 1, 2, \cdots, N_F \\ (4) \sum_{j=1}^{i_k} z_k^j = F_k, k = 1, \cdots, N_R \end{cases} \tag{2}$$

3.3. Two Evolutionary Algorithms for Solving the Model

Note that the above model is a nonlinear integer multi-objective optimization model. There is no existing algorithm for this model. We shall design two evolutionary algorithms to solve this model in this section.

The framework of the first algorithm is as follows: First, design a strategy to generate the initial population. Second, design a mutation operator. Finally, evolve the population iteratively and obtain a set of the approximate solutions. In the following, we shall introduce each part of the algorithm, respectively.

3.3.1. A Strategy for Generating the Initial Population

Given the population size N_{pop}, any individual (x, y, z) in the initial population POP^0 is generated by the following Algorithm 2:

Algorithm 2 A strategy for generating the initial population

Step 1: Generate each x by using Algorithm 1.

Step 2: Generate each $y = (y_1, y_2, \cdots, y_{N_R})^T$ as follows: denote

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_{N_R} \end{pmatrix} = \begin{pmatrix} y_1^1 & y_1^2 & \cdots & y_1^{N_F} \\ y_2^1 & y_2^2 & \cdots & y_2^{N_F} \\ \cdots & \cdots & \cdots & \cdots \\ y_{N_R}^1 & y_{N_R}^2 & \cdots & y_{N_R}^{N_F} \end{pmatrix}$$

Algorithm 2 *Cont.*

Set initial matrix $y = 0$. Let $y_{w_1}, y_{w_2} \cdots, y_{w_{N_R}}$ be a random sequence of $y_1, y_2, \cdots, y_{N_R}$ and y_w denote the matrix whose rows are $y_{w_1}, y_{w_2} \cdots, y_{w_{N_R}}$, respectively. Set $y^1_{w_1} = y^2_{w_1} = \cdots = y^{b_{w_1}}_{w_1} = 1$, and $y^j_{w_1} = 0$ for $j \neq 1, 2, \cdots, b_{w_1}$. If path P_{w_2} for task r_{w_2} shares a link with path P_{w_1} for task r_{w_1}, set $y^j_{w_2} = 1$ for $b_{w_1} + 1 \leq j \leq b_{w_1} + b_{w_2}$ and $y^j_{w_2} = 0$ for other j. Otherwise, set $y^1_{w_2} = y^2_{w_2} = \cdots = y^{b_{w_2}}_{w_2} = 1$, and $y^j_{w_2} = 0$ for $j \neq 1, 2, \cdots, b_{w_2}$. Generally, for each $k \in \{2, 3, \cdots, N_R\}$, if path P_{w_k} for task r_{w_k} does not share a link with any previous path P_{w_j} for $1 \leq j \leq k-1$, set $y^1_{w_k} = y^2_{w_k} = \cdots = y^{b_{w_k}}_{w_k} = 1$, and $y^j_{w_k} = 0$ for $j \neq 1, 2, \cdots, b_{w_k}$. Otherwise, let U_{w_k} denote the set of tasks whose paths share a link with path P_{w_k} for task w_k. Compute

$$\bar{y}_{w_k} = \sum_{v \in U_{w_k}} y_v$$

Let

$$k_0 = \min\{k'_0 | \sum_{i=k'_0}^{k'_0+b_{w_k}-1} \bar{y}^i_{w_k} = 0 \; for \; k'_0 \in \{1, 2, \cdots, N_F - b_{w_k}\}\}$$

Assign the slots $k_0, k_0 + 1, \cdots, k_0 + b_{w_k} - 1$ to path P_{w_k} for task r_{w_k}, i.e, let $y^{k_0}_{w_k} = y^{k_0+1}_{w_k} = \cdots = y^{k_0+b_{w_k}-1} = 1$, and $y^j_{w_k} = 0$ for $j \neq k_0, k_0 + 1, \cdots, k_0 + b_{w_k} - 1$.
Then, y satisfies the constraints:

$$\sum_{i=1}^{N_F} y^i_k = b_k, k = 1, \cdots, N_R$$

$$\sum_{v \in U_k} y^i_v \leq 1, i = 1, 2, \cdots, N_F$$

Step 3: Suppose the data center nodes passed successively through by path P_k are:

$$V_{k_1}, V_{k_2}, \cdots, V_{k_{i_k}}$$

The numbers of functions in task r_k assigned to data center nodes $V_{k_1}, V_{k_2}, \cdots, V_{k_{i_k}}$ are denoted by $z^1_k, z^2_k, \cdots, z^{i_k}_k$, respectively, where z^j_k is the number of functions assigned to data center V_{k_j} for $j = 1, 2, \cdots, i_k$. Denote

$$z_k = (z^1_k, z^2_k, \cdots, z^{i_k}_k)$$

For each task r_k, we can generate z_k by randomly taking i_k non-negative integers $z^1_k, z^2_k, \cdots, z^{i_k}_k \in [0, F_k]$ such that $\sum_{j=1}^{i_k} z^j_k = F_k, k = 1, \cdots, N_R$.

Step 4: Repeat Step 1 to Step 3 for N_{pop} times to obtain POP^0.

3.3.2. Mutation Operator

For any individual (x, y, z) in the current population POP^t, the mutation operator for it is as follows:

(1) Scheme of mutation for x:

1. For each x_k, generate $\bar{x}^1_k$. For s_k, randomly select a node in V_{s_k}/x^1_k as $\bar{x}^1_k$.

2. Generate $\bar{x}^{i+1}_k$ in $V^i_k/\{s_k, \bar{x}^1_k, \cdots, \bar{x}^i_k\}$ according to the following two cases:

Case I: $\{s_k, \bar{x}^1_k, \cdots, \bar{x}^i_k\}$ contains a data center node. If

$$\{\{x^1_k, x^2_k, \cdots, d_k\} - \{\bar{x}^1_k, \cdots, \bar{x}^i_k\}\} \cap V^i_k \neq \emptyset,$$

select one node

$$x_k^{i'} \in \{\{x_k^1, x_k^2, \cdots, d_k\} - \{\bar{x}_k^1, \cdots, \bar{x}_k^i\}\} \cap V_k^i,$$

and let $\bar{x}_k = (s_k, \bar{x}_k^1, \cdots, \bar{x}_k^i, x_k^{i'}, \cdots, d_k)$. We obtain the mutation offspring $\bar{x}_k$ of x_k; otherwise, randomly select a node in V_k^i as $\bar{x}_k^{i+1}$. Go to step 3.

Case II: $\{s_k, \bar{x}_k^1, \cdots, \bar{x}_k^i\}$ does not contain any data center node. If V_k^i contains any data center node, randomly select one data center node in V_k^i as $\bar{x}_k^{i+1}$. Let $i = i + 1$, go to step 2; otherwise, randomly select one node in V_k^i as $\bar{x}_k^{i+1}$, and go to step 3.

3. If $\bar{x}_k^{i+1} \notin \{x_k^1, x_k^2, \cdots, d_k\}$, let $i = i + 1$, and go to step 2; otherwise, go to step 4.

4. Select

$$i' \in \{v | x_k^v = \bar{x}_k^{i+1}, x_k^v \in \{x_k^1, x_k^2, \cdots, d_k\}\},$$

and let $\bar{x}_k = (s_k, \bar{x}_k^1, \cdots, \bar{x}_k^{i+1}, x_k^{i'+1}, \cdots, d_k)$. If $\bar{x}_k$ does not contain any data center node, go to step 5; otherwise, we obtain the mutation offspring $\bar{x}_k$ of x_k.

5. Re-select the nodes after node $\bar{x}_k^i$ until selecting node d_k by using the previous steps. If the new path is still an infeasible path (containing no data center node), re-select nodes after x_k^{i-1} until selecting d_k. Repeat this process until one feasible path is found.

(2) Scheme of mutation for y: Randomly change two elements of $y_{w_1}, y_{w_2} \cdots, y_{w_{N_R}}$, and obtain the mutation offspring $\bar{y}$ of y by step 2 of Algorithm 2.

(3) Scheme of mutation for z: Generate the mutation offspring $\bar{z}$ of z by step 3 of Algorithm 2.

Note that the offspring $(\bar{x}, \bar{y}, \bar{z})$ of (x, y, z) generated by the above schemes satisfies all constraints of the model. Let O^t denote the set of offspring of POP^t.

3.3.3. Selection Operator

We select the next generation population POP^{t+1} among $POP^t \cup O^t$ by using the selection operator in NSGA-II [22].

3.3.4. The First Evolutionary Algorithm for the Model

The first proposed algorithm is the following Algorithm 3:

Algorithm 3 An evolutionary algorithm for the model

Step 1: Given the maximum generation number N_{gen} and population size N_{pop}.

Step 2: Generate initial population POP^0 by using Algorithm 2, and set $t = 0$.

Step 3: Mutation. For each individual (x, y, z) in the current population POP^t, execute the mutation operator on it to generate its offspring $(\bar{x}, \bar{y}, \bar{z})$. All offspring $(\bar{x}, \bar{y}, \bar{z})$ form an offspring population O^t.

Step 4: Selection. Select the best N_{pop} individuals among $POP^t \cup O^t$ to form the next generation population POP^{t+1} by selection operator. Set $t = t + 1$.

Step 5: If $t \geq N_{gen}$, the non-dominated set of POP^t is the approximate Pareto optimal solution set. Stop. Otherwise, go to Step 2.

3.3.5. The Second Evolutionary Algorithm for the Model

The second proposed algorithm is the following Algorithm 4:

Algorithm 4 Another evolutionary algorithm for the model

The steps of Algorithm 4 are the same as those of Algorithm 3 except for the replacement of Algorithm 1 in Algorithm 3 by the following Algorithm 5 to generate path variable

The proposed algorithm for generating path variavles in Algorithm 4 is as follows:

Algorithm 5 Strategy for generating path variable x

Step 1: For each task r_k, randomly select i_k data center nodes, and their random sequence is denoted by $D_k = k_1^{'}, k_2^{'}, \cdots, k_{i_k}^{'}$.

Step 2: Use the Dijkstra algorithm [23] to obtain the shortest path from s_k to $k_1^{'}$, recorded as $Path1 : s_k \to \cdots \to k_1^{'}$.

Step 3: Use the Dijkstra algorithm to obtain the shortest path from $k_{i_k}^{'}$ to d_k, recorded as $Path3 : k_{i_k}^{'} \to \cdots \to d_k$.

Step 4: Use the Dijkstra algorithm to obtain the shortest path between each pair of two adjacent data centers $k_i^{'}$ and $k_{i+1}^{'}$ for $i = 1, 2, \cdots, i_k - 1$. This path is denoted by $p_{(k_i^{'}, k_{i+1}^{'})}$ for $i = 1, 2, \cdots, i_{k-1}$. Connect these paths successively to obtain a path through data center node sequence $D_k = k_1^{'}, k_2^{'}, \cdots, k_{i_k}^{'}$:

$$Path2 = \{p_{(k_1^{'}, k_2^{'})} \to p_{(k_2^{'}, k_3^{'})}, \cdots, \to p_{(k_i^{'}, k_{i+1}^{'})}, \cdots, \to p_{(k_{i_k-1}^{'}, k_{i_k}^{'})}\}$$

Step 5: Connect the tail node of $Path1$ with the head node of $Path2$, connect the tail node of $Path2$ with the head node of $Path3$, and then obtain the path $\{Path1 \to Path2 \to Path3\}$ of task r_k. Denote this path by $(s_k, x_k^1, x_k^2, \cdots, x_k^{n_k}, d_k)$, and briefly denoted as $x_k = (x_k^1, x_k^2, \cdots, x_k^{n_k})^T$ by removing start node s_k and end node d_k. Let $x = (x_1, x_2, \cdots, x_{N_R})^T$ denote the matrix whose rows are all paths for all tasks.

4. Experiments and Results

4.1. Experimental Environment and Parameters

All experiments are conducted on Windows 11 with 16 GB memory, R7-5800H 8 core CPU, 3.2 GHz by using Python 3.7. The experimental tool is PyCharm IDE. Four types of networks are used in the experiments: USANET [10], NSFNET [24], CHNNET [25], and ARPANET [25]. For the details of network topology and other information, please refer to [10,24,25]. The parameters are given in Table 1.

Table 1. Parameters for four types of networks.

Network Topology	NSFNET	CHNNET	ARPANET	USANET
Number of Nodes	14	15	20	24
Number of links	21	27	32	43
Number of spectrum	358	358	358	358

The experiments are divided into four groups. Each group is for one type network and contains four cases according to different numbers of tasks denoted by N_R and the number of VNF denoted by N_T:

Case 1. The number of tasks $N_R = 50$ and the number of VNF $N_T = 100$.
Case 2. $N_R = 100$ and $N_T = 100$.
Case 3. $N_R = 150$ and $N_T = 100$.
Case 4. $N_R = 200$ and $N_T = 100$.

To test the stability of the model and algorithms, for each case, the experiments are conducted 10 times, and the mean results are recorded and compared. In the experiments, the population size $N_{pop} = 50$ and the maximum generations $N_{gen} = 100$.

4.2. Comparison of Algorithms 3 and 4

Because the proposed model is a novel one, there is no existing algorithms for the proposed model, we cannot find any suitable existing algorithm as the compared algorithm. We compare the proposed Algorithms 3 and 4 in the experiments.

4.3. Performance Measures and Results

In the experiments, we adopt two performance measures: U-measure and C-measure.

1) U-measure [26]:

This metric represents the uniformity and wideness of the Pareto front, which is usually a surface in three-dimensional space. This measure can be calculated by the following three steps:

(1) Determine the boundary of Pareto front.

(2) Determine the nearest neighbors of each Pareto solution in objective space.

(3) Calculate the standard deviation of the distances among the nearest neighbors.

Note that the smaller the U-measure, the better the solution set. For the detail of U-measure, please refer to reference [26].

2) C-measure [27]:

This metric compares the convergent quality of two solution sets. It represents the percentage of solutions in solution set B dominated by solutions in solution set A. Its definition is as follows:

$$C(A,B) = \frac{|\{u \in B | \exists v \in A : v \ dominates \ u\}|}{|B|}$$

where the numerator represents the number of solutions in B dominated by at least one solution in A, and the denominator is the number of solutions in B.

$C(A,B) = 1$ means all solutions in B are dominated by at least one solution in A and $C(A,B) = 0$ means no solution in B is dominated by solutions in A. If $C(A,B) > C(B,A)$, then the solution set A has better convergence than solution set B.

In the C-measure, for notation convenience, we use A to represent the solution set obtained by Algorithm 3 and B the solution set obtained by Algorithm 4, respectively. In addition, use C_{AB} and C_{BA} to represent $C(A,B)$ and $C(B,A)$, respectively, where

$$C_{AB} = C(A,B) = \frac{|\{u \in B | \exists v \in A : v \ dominates \ u\}|}{|B|}$$

$$C_{BA} = C(B,A) = \frac{|\{u \in A | \exists v \in B : v \ dominates \ u\}|}{|A|}$$

For each case of one type network, the mean value and standard deviation of each metric in 10 independent runs are recorded. These results are given in Tables 2 and 3, where each table records the results for one measure on 16 cases of 4 types of networks.

Table 2. Comparison of U-measure.

Networks	(N_T, N_R)	Algorithm 3		Algorithm 4	
		U_{mean}	U_{std}	U_{mean}	U_{std}
NSFNET	(100,50)	1.0164	0.0073	1.0147	0.0078
	(100,100)	1.0093	0.0030	1.0119	0.0040
	(100,150)	1.0118	0.0060	1.0110	0.0029
	(100,200)	1.0065	0.0032	1.0079	0.0029

Table 2. *Cont.*

Networks	(N_T, N_R)	Algorithm 3		Algorithm 4	
		U_{mean}	U_{std}	U_{mean}	U_{std}
USANET	(100,50)	1.0168	0.0072	1.0181	0.0077
	(100,100)	1.0105	0.0089	1.0119	0.0045
	(100,150)	1.0042	0.0011	1.0077	0.0042
	(100,200)	1.0047	0.0024	1.0073	0.0029
ARPANET	(100,50)	1.0134	0.0065	1.0158	0.0064
	(100,100)	1.0078	0.0056	1.0102	0.0052
	(100,150)	1.0069	0.0035	1.0095	0.0043
	(100,200)	1.0059	0.0031	1.0072	0.0018
CHNNET	(100,50)	1.0191	0.0085	1.0151	0.0055
	(100,100)	1.0110	0.0048	1.0091	0.0042
	(100,150)	1.0096	0.0046	1.0080	0.0031
	(100,200)	1.0059	0.0022	1.0068	0.0035

Table 3. Comparison of C-measure.

Networks	(N_T, N_R)	C_{AB}		C_{BA}	
		C_{mean}	C_{std}	C_{mean}	C_{std}
NSFNET	(100,50)	0.7727	0.1449	0	0
	(100,100)	0.4725	0.1760	0	0
	(100,150)	0.5260	0.1671	0	0
	(100,200)	0.4705	0.1131	0	0
USANET	(100,50)	0.7039	0.2410	0	0
	(100,100)	0.6952	0.1279	0.0488	0.0804
	(100,150)	0.4216	0.2087	0.0995	0.0832
	(100,200)	0.5364	0.1452	0.0083	00264
ARPANET	(100,50)	0.8690	0.0872	0.0287	0.0607
	(100,100)	0.6871	0.1356	0.0500	0.1208
	(100,150)	0.5837	0.1544	0.0133	0.0422
	(100,200)	0.4462	0.1484	0.0214	0.0678
CHNNET	(100,50)	0.9332	0.0737	0	0
	(100,100)	0.8553	0.1135	0	0
	(100,150)	0.9536	0.0813	0	0
	(100,200)	0.9778	0.0703	0	0

For U-measure, it can be seen from Table 2 that Algorithm 3 generally performs better than Algorithm 4. Among 16 cases of 4 types of networks, Algorithm 3 has the better performance than Algorithm 4 on 12 cases in which Algorithm 3 obtains the smaller values of U-measure. For three cases $(N_T, N_R) = (100, 100)$, $(N_T, N_R) = (100, 150)$ and $(N_T, N_R) = (100, 200)$ of NSFNET, the values of U-measure of Algorithm 3 are 1.0093, 1.0118, and 1.0065, respectively, which are smaller than the values 1.0119, 1.011, and 1.0079 of U-measure of Algorithm 4. For all four cases of USANET, the values of U-measure of Algorithm 3 are 1.0168, 1.0105, 1.0042, and 1.0047, respectively, while the values of U-measure of Algorithm 4 are 1.0181, 1.0119, 1.0077, and 1.0073, respectively. Thus, the values of U-measure of Algorithm 3 are smaller than those of Algorithm 4 for all cases of USANET. Similarly, it can be seen from Table 2 that, for all four cases of ARPANET, the values of U-measure of Algorithm 3 are smaller than those of Algorithm 4, which indicates that Algorithm 3 performs better than Algorithm 4 on U-measure. Only for four cases (one case of NSFNET and three cases of CHNNET) does Algorithm 4 obtain smaller values of U-measure. For the case $(N_T, N_R) = (100, 50)$ of NSFNET, the value of U-measure of Algorithm 3 is 1.0164 which is larger than that (i.e., 1.0147) of Algorithm 4. For three cases of CHNNET with a smaller number of tasks, the values of U-measure of Algorithm 3 are a little bit larger than the corresponding values of U-measure of Algorithm 4, while for the larger number of tasks (i.e., $NR = 200$), Algorithm 3 outperforms Algorithm 4. Anyhow, both algorithms obtain relatively small values of U-measure on all 16 cases of all types of

networks. This illustrates that both algorithms perform well. In addition, note that the standard deviation values for two algorithms are relatively small. This indicates that both algorithms are robust and perform stably.

For C-measure, it can be seen from Table 3 that Algorithm 3 performs completely better than Algorithm 4 with respect to the convergent ability. For all four cases of NSFNET, about 77.27%, 47.25%, 52.6%, and 47.05% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, respectively, while no solution obtained by Algorithm 3 is dominated by solutions obtained by Algorithm 4. For USANET, for case $(N_T, N_R) = (100, 50)$, 70.39% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while no solution obtained by Algorithm 3 is dominated by solutions obtained by Algorithm 4. For case $(N_T, N_R) = (100, 100)$, 69.52% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while 4.88% solutions obtained by Algorithm 3 are dominated by solutions obtained by Algorithm 4. For case $(N_T, N_R) = (100, 150)$, 42.16% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while 9.95% solutions obtained by Algorithm 3 are dominated by solutions obtained by Algorithm 4. For case $(N_T, N_R) = (100, 200)$, 53.64% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while 0.83% solutions obtained by Algorithm 3 are dominated by solutions obtained by Algorithm 4. For ARPANET, for case $(N_T, N_R) = (100, 50)$, 86.9% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while 2.87% solutions obtained by Algorithm 3 are dominated by solutions obtained by Algorithm 4. For case $(N_T, N_R) = (100, 100)$, 68.71% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while 5.0% solutions obtained by Algorithm 3 are dominated by solutions obtained by Algorithm 4. For case $(N_T, N_R) = (100, 150)$, 58.37% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while 1.33% solutions obtained by Algorithm 3 are dominated by solutions obtained by Algorithm 4. For case $(N_T, N_R) = (100, 200)$, 44.62% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while 2.14% solutions obtained by Algorithm 3 are dominated by solutions obtained by Algorithm 4. For CHNNET, for case $(N_T, N_R) = (100, 50)$, 93.32% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while no solution obtained by Algorithm 3 is dominated by solutions obtained by Algorithm 4. For case $(N_T, N_R) = (100, 100)$, 85.53% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while no solution obtained by Algorithm 3 is dominated by solutions obtained by Algorithm 4. For case $(N_T, N_R) = (100, 150)$, 95.36% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while no solution obtained by Algorithm 3 is dominated by solutions obtained by Algorithm 4. For case $(N_T, N_R) = (100, 200)$, 97.78% solutions obtained by Algorithm 4 are dominated by solutions obtained by Algorithm 3, while no solution obtained by Algorithm 3 is dominated by solutions obtained by Algorithm 4.

By summary, Algorithm 3 performs better than Algorithm 4 in most cases for all types of networks by using two performance metrics. However, both algorithms can obtain well distributed solution sets.

5. Discussion and Future Works

Task scheduling and resource allocation in an elastic optical network is an important and challenging problem. Different providers and customers may have different requirements. With the increase of numbers of tasks and VNF, the problems will become more and more difficult. This is very challenging to design efficient algorithm for the model. However, the proposed multi-objective optimization model in this paper can well satisfy these multiple requirements of both providers and customers as most as possible, and the designed algorithms can solve the model well. In addition, it seems that the proposed algorithm is still effective and efficient with the increase of the numbers of tasks and VNF. However, the limitation of this work is mainly that, when the links of networks are sparse and there are too many tasks, the performance of the proposed model and algorithm will

decrease. In addition, the task scheduling and resource allocation in the elastic optical network involves too many issues except for routing, spectrum allocation, VNF deployment, and load balance of data centers. In the future work, it is necessary to study how to set up a new model with more objectives (many objective optimization model) to tackle more issues and satisfy more requirements for service providers and customers in addition to how to design more efficient algorithms for the multi-objective optimization model.

6. Conclusions

In this paper, we set up a new multi-objective optimization model for task scheduling and resource allocation problem in elastic optical networks. By using this model, both the total length of paths for all tasks and the totally occupied spectrums can be minimized, which illustrates that the lowest amount of resources are consumed. In addition, the loads on all data centers can be mostly balanced, which illustrates that the efficiency of the network is the highest. Thus, the new model is effective and efficient. To solve the model, we design two new effective evolutionary algorithms. The experiments conducted on up to 16 cases of 4 types of widely used networks have indicated that the proposed algorithms are effective.

Author Contributions: Conceptualization, Y.W. and X.G.; methodology, Y.W. and X.G.; software, Q.Y.; validation, Q.Y.; formal analysis, Y.W. and X.G.; data curation, Q.Y.; writing—original draft preparation, Y.W.; writing—review and editing, X.G. and Q.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NSFC of Grant Nos. 62102304 and 62272367.

Institutional Review Board Statement: Not applicable

Informed Consent Statement: Not applicable.

Data Availability Statement: For the network framework data, please refer to [18–20].

Acknowledgments: This research was supported by the National Natural Science Foundation of China with Grant Nos. 62102304 and 62272367.

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

References

1. Abkenar, F.S.; Rahbar, A.G. Study and analysis of routing and spectrum allocation (RSA) and routing, modulation and spectrum allocation (RMSA) algorithms in elastic optical networks (EONs). *Opt. Switch. Netw.* **2017**, *23*, 5–39. [CrossRef]
2. Alyatama, A. Multi-path routing based on relative cost in elastic optical networks. In Proceedings of the 2020 7th International Conference on Electrical and Electronics Engineering (ICEEE), Virtual Conference, 14–16 April 2020.
3. Zhou, Y.; Sun, Q.; Lin, S. Link state aware dynamic routing and spectrum allocation strategy in elastic optical networks. *IEEE Access* **2020**, *8*, 45071–45083. [CrossRef]
4. Yan, B.; Zhao, Y.; Yu, X.; Wang, W.; Wu, Y.; Wang, Y.; Zhang, J. Tidal-traffic-aware routing and spectrum allocation in elastic optical networks. *J. Opt. Commun. Netw.* **2018**, *10*, 832–842. [CrossRef]
5. Al-Yatama, A. Relative cost routing and spectrum allocation in elastic optical networks. *J. Opt. Commun. Netw.* **2020**, *12*, 38–49. [CrossRef]
6. Gocień, R. Two metaheuristics for routing and spectrum allocation in cloud-ready survivable elastic optical networks. *Swarm Evol. Comput.* **2019**, *44*, 388–403. [CrossRef]
7. Eramo, V.; Miucci, E.; Ammar, M.; Lavacca, F.G. An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures. *IEEE/ACM Trans. Netw.* **2017**, *25*, 2008–2025. [CrossRef]
8. Mechtri, M.; Ghribi, C.; Zeghlache, D. A scalable algorithm for the placement of service function chains. *IEEE Trans. Netw. Serv. Manag.* **2016**, *13*, 533–546. [CrossRef]
9. Herrera, J.G.; Botero, J.F. Resource allocation in NFV: A comprehensive survey. *IEEE Trans. Netw. Serv. Manag.* **2016**, *13*, 518–532. [CrossRef]
10. Assis, K.D.R.; Santos, A.F.; Almeida, R.C.; Reed, M.J.; Jaumard, B.; Simeonidou, D. Virtualization of elastic optical networks and regenerators with traffic grooming. *J. Opt. Commun. Netw.* **2020**, *12*, 428–442. [CrossRef]
11. Liu, Y.; Gu, H.; Yan, F.; Calabretta, N. Highly-efficient switch migration for controller load balancing in elastic optical inter-datacenter networks. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 2748–2761. [CrossRef]

12. Mesquita, L.; Assis, K.; Pinho, L.; Almeida, R.; Filho, E.; Alencar, M. Resource planning on elastic optical networks using traffic matrix prediction. *Int. J. Electron. Commun.* **2021**, *134*, 153615. [CrossRef]
13. Yin, S.; Zhang, Z.; Yang, C.; Chu, Y.; Huang, S. Prediction-based end-to-end dynamic network slicing in hybrid elastic fiber-wireless networks. *J. Light. Technol.* **2021**, *39*, 1889–1899. [CrossRef]
14. Xu, Y.; Zhao, Z.; Cheng, P.; Chen, Z.; Ding, M.; Vucetic, B.; Li, Y. Constrained reinforcement learning for resource allocation in network slicing. *IEEE Commun. Lett.* **2021**, *25*, 1554–1558. [CrossRef]
15. Josilo, S.; Dsn, G. Joint wireless and edge computing resource management with dynamic network slice selection. *IEEE-ACM Trans. Netw.* **2022**, *30*, 1865–1878. [CrossRef]
16. Zhao, Y.; Zhang, Q.; Xin, X.; Li, Y.; Gao, R.; Tao, Y.; Tian, Q.; Tian, F.; Chen, D.; Cao, G. Static resource allocation of advanced reservation requests in elastic optical networks. *Appl. Opt.* **2020**, *59*, 1420–1429. [CrossRef] [PubMed]
17. Miyamura, T.; Misawa, A. Improving Efficiency of Network Resources in Elastic Optical Transport Network by Using In-Network Cache Functions. *Opt. Switch. Netw.* **2021**, *42*, 100629. [CrossRef]
18. Tang, Y.; Li, X.; Gao, T.; Zhang, L.; Huang, S. Cost-adaptive multi-class multicast service aggregation based on distributed sub-trees in elastic optical data center networks. *Opt. Fiber Technol.* **2021**, *66*, 102661. [CrossRef]
19. Kitsuwan, N.; Pavarangkoon, P.; Nag, A. Elastic optical network with spectrum slicing for fragmented bandwidth allocation. *Opt. Switch. Netw.* **2020**, *38*, 100583. [CrossRef]
20. Khatiri, A.; Mirjalily, G.; Luo, Z.Q. Balanced resource allocation for VNF service chain provisioning in inter-datacenter elastic optical networks. *Comput. Netw.* **2022**, *203*, 108717. [CrossRef]
21. Zhu, M.; Chen, Q.; Gu, J.; Gu, P. Deep Reinforcement Learning for Provisioning Virtualized Network Function in Inter-Datacenter Elastic Optical Networks. *IEEE Trans. Netw. Serv. Manag.* **2022**, *19*, 3341–3351. [CrossRef]
22. Kalyanmoy, D.; Pratap, A.; Agrawal, S. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197.
23. Hershberger, J. Finding the k shortest simple paths: A new algorithm and its implementation. *ACM Trans. Algorithms* **2007**, *3*, 1–5. [CrossRef]
24. Gong, L.; Zhou, X.; Liu, X.; Zhao, W.; Lu, W.; Zhu, Z. Efficient resource allocation for all-optical multicasting over spectrum-sliced elastic optical networks. *IEEE/OSA J. Opt. Commun. Netw.* **2013**, *5*, 836–847. [CrossRef]
25. Xuan, H.; Wang, Y.; Xu, Z.; Hao, S.; Wang, X. New bi-level programming model for routing and spectrum assignmentin elastic optical network. *Opt. Quantum Electron.* **2017**, *49*, 186. [CrossRef]
26. Leung, Y.W.; Wang, Y. U-measure: A quality measure for multiobjective programming. *IEEE Trans. Syst. Man Cybern* **2003**, *33*, 337–343. [CrossRef]
27. Zitzler, E.; Deb, K.; Thiele, L. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evol. Comput.* **2000**, *8*, 173–195. [CrossRef]