electronics

Article

Discrete Artificial Fish Swarm Algorithm-Based One-Off Optimization Method for Multiple Co-Existing Application Layer Multicast Routing Trees

Ying Li [1,2], Ning Wang [1,2], Wei Zhang [3], Qing Liu [4] and Feng Liu [1,2,*]

[1] School of Computer Science, Xi'an Polytechnic University, Xi'an 710048, China; 20190503@xpu.edu.cn (Y.L.); wangning@stu.xpu.edu.cn (N.W.)
[2] Shaanxi Key Laboratory of Clothing Intelligence, Xi'an Polytechnic University, Xi'an 710048, China
[3] China Mobile System Integration Co., Ltd., Xi'an 710077, China; zhangwei1@cmict.chinamobile.com
[4] Department of Information and Control, Xi'an University of Technology, Xi'an 710018, China; liuqing@xaut.edu.cn
* Correspondence: liufeng@xpu.edu.cn

Abstract: As an effective multicast application mechanism, the application layer multicast (ALM) determines the path of data transmission through a routing tree. In practical applications, multiple multicast sessions often occur simultaneously; however, few studies have considered this situation. A feasible solution is to sequentially optimize each co-existing ALM routing tree. However, this approach can lead to node congestion, and, even if the node out-degree reservation strategy is adopted, an optimal solution may not be obtained. In this study, to solve the problem of routing tree construction for multiple co-existing application layer multicast sessions, an optimization model that minimizes the overall delay and instability is constructed, and a one-off optimization method based on the discrete artificial fish swarm algorithm (DAFSA) is proposed. First, Steiner node sets corresponding to the multicast sessions are selected. Then, the routing trees for each multicast session are obtained through the improved spanning tree algorithm based on the complete graph composed of Steiner node sets. The experimental results show that the proposed method can simultaneously obtain multiple co-existing ALM routing trees with a low total delay and low instability. Even if the input is a single multicast session, it can lead to ALM routing trees with a lower delay and less instability than other algorithms, and the introduction of a penalty function can effectively avoid the problem of excessive replication and forwarding loads on some end-hosts. In addition, the proposed algorithm is insensitive to parameter changes and exhibits good stability and convergence properties for networks of different sizes.

Keywords: multiple co-existing ALM routing trees; node congestion; one-off optimization; DAFSA

Citation: Li, Y.; Wang, N.; Zhang, W.; Liu, Q.; Liu, F. Discrete Artificial Fish Swarm Algorithm-Based One-Off Optimization Method for Multiple Co-Existing Application Layer Multicast Routing Trees. *Electronics* **2024**, *13*, 894. https://doi.org/10.3390/electronics13050894

Academic Editor: Christos J. Bouras

Received: 4 February 2024
Revised: 21 February 2024
Accepted: 23 February 2024
Published: 26 February 2024

1. Introduction

With the increasing number of Internet users and the constantly updating and evolving forms of Internet, the proportion of real-time multimedia transmission application scenarios has increased significantly, leading to higher requirements for information transmission. Under the current application requirements, IP multicast technology has developed rapidly. As a one-to-many communication mode, IP multicast technology can effectively save network bandwidth and reduce the network load. It is suitable for applications that are centralized in time and distributed in space, such as video conferencing, streaming media, and so on. However, due to the charging mechanisms and technical limitations of Internet service providers (ISPs), the popularity of IP multicasting [1,2] on the Internet is restricted. In contrast, the application layer multicast (ALM) [3] migrates multicast data transmission from the IP layer to the application layer; data are replicated and forwarded through end-hosts. Furthermore, such approaches have the advantages of being easy to deploy

and economical, as communication between the underlying layers of ALM sessions is still based on the very widespread unicast technology.

The key in application layer multicast communication is the construction of an ALM routing tree, which is mainly used to determine the tree structure in which data are delivered from the sender to all the receivers in the group. ALM routing trees are composed of user nodes, which may exit or fail. This uncontrollability can lead to instability in the ALM routing tree, thus affecting the ability of users to receive multicast data [4]. Many researchers have attempted to reduce the instability caused by user nodes' behavior by optimizing the topology of ALM routing trees [5]. End-hosts with high stability are more easily used as core nodes to transmit the data based on the behavior and attributes of the user nodes. To optimize the ALM routing tree topology, Cao et al. have established an instantaneous stability model for the application layer multicast [6] and successfully addressed the bounded-delay and high-stability model challenges [7]. In application layer multicast optimization, the delay is also an important optimization objective. Huo et al. proposed an algorithm based on the stability probability and contribution link of nodes (CL-S) [8]. This approach incorporates considerations for node out-degree and edge delay. Mercan et al. proposed the virtual direction multicast (VDM) [9] and noted that, as long as the virtual distance is based on the delay and the stability, the VDM can construct a stable ALM routing tree with a low transmission delay. Li et al. have noted that in the coverage network, apart from the link delay, the replication delay of user nodes in processing messages should also be considered [10]. Liao et al. have proposed an ALM model based on the node potential (NP) and a topological index (TI), which is suitable for applications in large-scale, real-time multimedia environments [11]. Li et al. have proposed a class of algorithms that create a greedy multicast tree based on the ratio of fan-out to delay (RFD) and the probability of terminal stability to obtain a high performance in multicast sessions [12]. This problem belongs to the class of combinatorial optimization problems, which is characterized by a high degree of complexity and computational difficulty. However, intelligent algorithms have some significant advantages in this regard. Some scholars have utilized neural networks to solve similar problems [13,14]. Some scholars have used evolutionary algorithms to solve it. For example, Pan et al. have designed a genetic algorithm to minimize the end-to-end delay under the out-degree constraint [15]. In addition to the delay, Ma et al. have considered the average path stretch and used the artificial fish swarm algorithm to solve the problem [16]. Based on previous research, Liu et al. have further considered the instability index of an ALM routing tree and designed an encoding-free non-dominated sorting genetic algorithm to simultaneously optimize the total delay and instability of the ALM routing tree [5].

The above algorithms mainly optimize the delay and stability of ALM routing trees; however, several problems remain to be solved. The existing research has been optimized under a single conversation scenario. However, multiple multicast sessions existing simultaneously is fairly common. At present, studies on the simultaneous optimization of multiple co-existing ALM routing trees are rare. One feasible method for achieving this is to use a single ALM routing tree construction method multiple times; that is, the algorithms are used sequentially to construct each ALM routing tree. It is worth noting that, to improve the stability of data transmission, when constructing the ALM routing tree, the user nodes with a higher stability are preferentially selected as the core nodes for data forwarding. However, if these user nodes appear in multiple co-existing ALM routing trees at the same time, these user nodes' out-degree (the number of times end-hosts copy and forward the data) significantly increases. Due to the limitations in the ability of end-hosts to copy and forward data, when the out-degree of user nodes is too large, node congestion will occur. This is especially relevant for forwarding nodes that are close to the source and may experience massive stress issues [17], further affecting the stability of the ALM routing tree. Therefore, when multiple ALM routing trees are optimized at the same time, the out-degree of the user nodes in each ALM routing tree needs to be reasonably distributed to ensure that the total out-degree of each end-host does not exceed their capability.

This study aims to obtain multiple co-existing ALM routing trees based on multiple co-existing multicast sessions while striking a balance between minimizing the total delay and instability of these ALM routing trees. We introduce the node out-degree as a constraint to prevent the instability of multicast sessions caused by node congestion. First, a low delay and low instability model of multiple co-existing ALM routing trees is established. To achieve the optimization goal, a one-off solution method is proposed in this study. In this method, the encoding of the DAFSA represents the selection scheme of Steiner node sets for multiple multicast sessions, and then multiple ALM routing trees are obtained from the complete graph corresponding to the multiple Steiner node sets through the use of the spanning tree algorithm. The fitness function in the DAFSA is used to evaluate the generated ALM routing tree, which is iterated continuously to find the optimal ALM routing tree. Node congestion analysis is performed on the designed algorithm to verify the effectiveness of the algorithm in dealing with the node out-degree constraints, and the performance of the algorithm is verified through detailed simulation experiments. Due to the large difference in the importance of the two objective functions—namely, the delay and the instability—a weight selection method is used to assist in decision making.

The rest of this paper is organized as follows. In Section 2, the constructed application layer multicast stability model is introduced. In Section 3, the idea to solve the model of the problem is introduced, which is divided into two parts: selecting the Steiner point sets and improving the spanning tree algorithm. In Section 4, the design of the DAFSA and the improvement of Prim's spanning tree algorithm are described in detail. In Section 5, exhaustive simulation experiments are shown, and the obtained results are analyzed. In Section 6, the experimental results and the design approach of this paper are discussed. In Section 7, a summary is given.

2. Optimization Model for Multiple Co-Existing ALM Routing Trees

The application layer network can be expressed as $G = (V, E)$, consisting of a vertex set V and an edge set E. $v \in V$ represents a user node and $e \in E$ represents the communication channel between two user nodes. For a communication channel e, the transmission delay is denoted as $d(e)$, and the delay caused by message processing in the user node is denoted as $d(v)$. The user node v has a probability $p(v)$ of leaving from graph G. For a user node v, the out-degree is denoted as Od_v (which cannot exceed D_v), and the number of its descendants is denoted as Nd_v. In this paper, we mainly optimize the delay and instability of ALM routing trees. The routing tree for a single multicast session, including one source and multiple destinations, can be denoted as $T_k = \{V^{T_k}, E^{T_k}\}$. The optimization model for multiple co-existing ALM routing trees needs to be based on K groups as the source and M destinations, generating K ALM routing trees, which are denoted as $T_1, T_2, \cdots T_k, \cdots T_K$. The out-degree of user node $v_i^{T_k}$ in ALM routing tree T_k is denoted as $Od_{v_i^{T_k}}$.

2.1. Delay

Delay refers to the time required for data to travel from a source node to a destination node. In an application layer multicast session, the intermediate nodes that forward data are the end-hosts. The equipment of the end-hosts has a limited forwarding capability, so the processing delay cannot be ignored. Therefore, the delay in this paper includes two parts: the transmission delay and the processing delay in end-hosts. The delay of the ALM routing tree T_k is denoted as $f_1(T_k)$, and the total delay is calculated as shown in Equation (1).

$$min \sum_{k=1}^{K} f_1(T_k) = \sum_{k=1}^{K} \left(\sum_{e_i^{T_k} \in E^{T_k}} d(e_i^{T_k}) + \sum_{v_i^{T_k} \in V^{T_k}} d(v_i^{T_k}) Od_{v_i^{T_k}} \right) \quad (1)$$

2.2. Instability

Instability mainly focuses on the exit and failure of user nodes. Node exiting means that a user node voluntarily leaves the application layer multicast session, while user node failure means that a user node leaves the application layer multicast session without notifying any other user nodes. In the ALM routing tree, the exit and failure behaviors of non-leaf nodes cause their descendant nodes to lose connectivity with the root node of the multicast tree.

2.2.1. Reducing the Impact of User Nodes' Exiting Behavior

User nodes exiting is a spontaneous behavior. As the distribution of the online times for the end-hosts in multicast sessions shows a heavy-tailed phenomenon [7,18], this study pays more attention to the probability of user nodes exiting and uses the average number of descendant user nodes affected by the exit of the user nodes to measure the instability of ALM routing trees. The instability of ALM routing tree T_k is denoted as $f_2(T_k)$, and the total instability is calculated as shown in Equation (2).

$$min \sum_{k=1}^{K} f_2(T_k) = \sum_{k=1}^{K} \left(\frac{1}{1 + Nd_{source^{T_k}}} \sum_{v_i^{T_k} \in V^{T_k}} p(v_i^{T_k}) Nd_{v_i^{T_k}} \right) \tag{2}$$

2.2.2. Reducing the Risk of User Nodes' Failure

User node failure is a passive behavior, which usually occurs as user nodes lose the ability to forward data due to experiencing a heavy load. Therefore, in this study, the out-degree of a node is limited to reduce the load on the end-host. Equation (3) is the constraint.

$$s.t. \sum_{k=1}^{K} Od_{v_i^{T_k}} \leq D_v \tag{3}$$

In this study, the delay and instability are considered as the optimization objectives. However, these two objective functions may be in conflict. To find an appropriate trade-off in the multi-objective problem, weights for the objective functions are introduced to convert the multi-objective problem into a single-objective problem. Equation (4) is the specific formula.

$$min \quad w_1 \sum_{k=1}^{K} f_1(T_k) + w_2 \sum_{k=1}^{K} f_2(T_k) \tag{4}$$

3. One-Off Optimization

The problem of ALM routing tree construction is essentially the Steiner tree problem in graph theory [19,20]. This problem requires finding the optimal tree that contains specified terminal nodes. However, solving this problem is very complicated: it has been proven to be NP-complete [21], which means that there is no effective algorithm for solving it in polynomial time, and the solution space can be searched only with methods of exponential or even factorial complexity.

In the construction of multiple co-existing ALM routing trees, multiple co-existing application layer multicast sessions correspond to multiple Steiner trees. This further escalates the difficulty of solving the problem, as different multicast sessions may share nodes, and the out-degree of a node needs to be guaranteed not to exceed the performance limit of the node.

Although the co-existing Steiner tree optimization problem is difficult to solve, the spanning tree problem is relatively simple, which involves finding a single tree that contains all the vertices. This has been studied in depth and includes the minimum spanning tree problem [22,23], the degree-constrained minimum spanning tree problem [24], the multi-objective spanning tree problem [25], and so on.

In addition, it is very difficult to rationally allocate the out-degree of nodes between multiple co-existing ALM routing trees, which often results in an inability to obtain a feasible solution. However, the good adaptability and global search ability of the DAFSA enable it to perform well when dealing with problems involving complex constraints [26]. At present, the processing methods for infeasible solutions include the use of penalty functions, repair methods, and so on.

In this study, the problem is decomposed into the following two parts.

3.1. Evolution: Using the DAFSA, Based on the Actual Source Nodes and the Destination Nodes, an Appropriate Set of Steiner Nodes Is Selected through a Population Iteration

The key to solving the considered problem is selecting the other user nodes that are not the source and the destinations (Steiner nodes) instead of user nodes. These nodes serve as the core nodes that connect the destination nodes. The positions and numbers of these nodes usually vary, according to the nature of the problem and the optimization goal. A trade-off needs to be struck between low node instability and a low delay between the source and the destinations while also considering the out-degree constraints of the user nodes to rationally distribute the Steiner nodes in each tree. These nodes, the source nodes, and the destination nodes are combined into a complete subgraph.

The discrete artificial fish swarm algorithm is a swarm intelligence algorithm. The basic idea of this algorithm is to simulate the behavior of individual fish in a fish swarm, such that the whole swarm can cooperatively find an optimal solution in the solution space. Each artificial fish represents a candidate solution in the solution space, and they exchange information and adjust their positions to find an optimal solution. Owing to a number of salient properties, which include flexibility, a fast convergence, and insensitivity to the initial parameter settings, the AFSA family has emerged as an effective swarm intelligence (SI) methodology that has been widely applied to solving real-world optimization problems [27]. One of its main advantages is the ability to perform a global search in the search space and avoid becoming trapped in local optimal solutions.

The algorithm contains a series of behavior rules, such as foraging, following, randomly moving, and so on. These rules simulate the behavior of individual artificial fish when searching for food and avoiding danger:

(1) Randomly moving behavior: The individual randomly moves in various directions within its *step* limit.

(2) Foraging behavior: The individual randomly explores a new position within its *visual* limit. If the new position has a better fitness, it moves toward this position within its *step* limit; otherwise, if a position with a better fitness cannot be found within a limited number of *try_number* times, it will move randomly.

(3) Following behavior: The individual perceives the optimal individual within its *visual* limit and moves toward that individual if the surrounding area is not crowded; otherwise, the individual performs foraging.

In this study, the artificial fish school behavior strategy designed by Ma et al. [16] was used. First, whether the artificial fish (AF) are crowded or not is determined. If not, the fish perform the following behavior and the algorithm ends. Otherwise, the individual enters into foraging behavior.

3.2. Evaluation: Based on the Spanning Tree Algorithm, the Complete Subgraph Is Converted into an ALM Routing Tree, and the Fitness Value Is Calculated

For this part, an ALM routing tree must be constructed based on the obtained complete subgraph; that is, all of the terminal nodes are connected using Steiner nodes, ensuring that the objective function is optimized. This problem is similar to the minimum spanning tree problem.

Prim's algorithm [22] has the advantages of simplicity and efficiency in processing the minimum spanning tree problem, the basic idea of which is to start from an initial node and gradually select the shortest edge connected to the current spanning tree until all the nodes are covered. According to the objective function defined above, this study

improves Prim's algorithm to heuristically construct an ALM routing tree with a low delay and better stability.

4. One-Off Optimization Method for Multiple Co-Existing Application Layer Multicast Trees

In this study, the DAFSA is used as the core method for the optimization of multiple co-existing ALM routing trees. First, based on the input multicast session, multiple sets of suitable Steiner node sets are selected to form a complete subgraph, as shown in Figure 1. Then, multiple subgraphs are converted into ALM routing trees using the improved spanning tree algorithm. Subsequently, evaluation and updating of the bulletin board (used to store the set of optimal routing trees) was performed. The optimal ALM routing trees were ultimately obtained through continuous iteration. It is worth noting that the improved spanning tree algorithm is a deterministic algorithm, and the selected Steiner node set directly affects the fitness function used to evaluate the ALM routing tree.

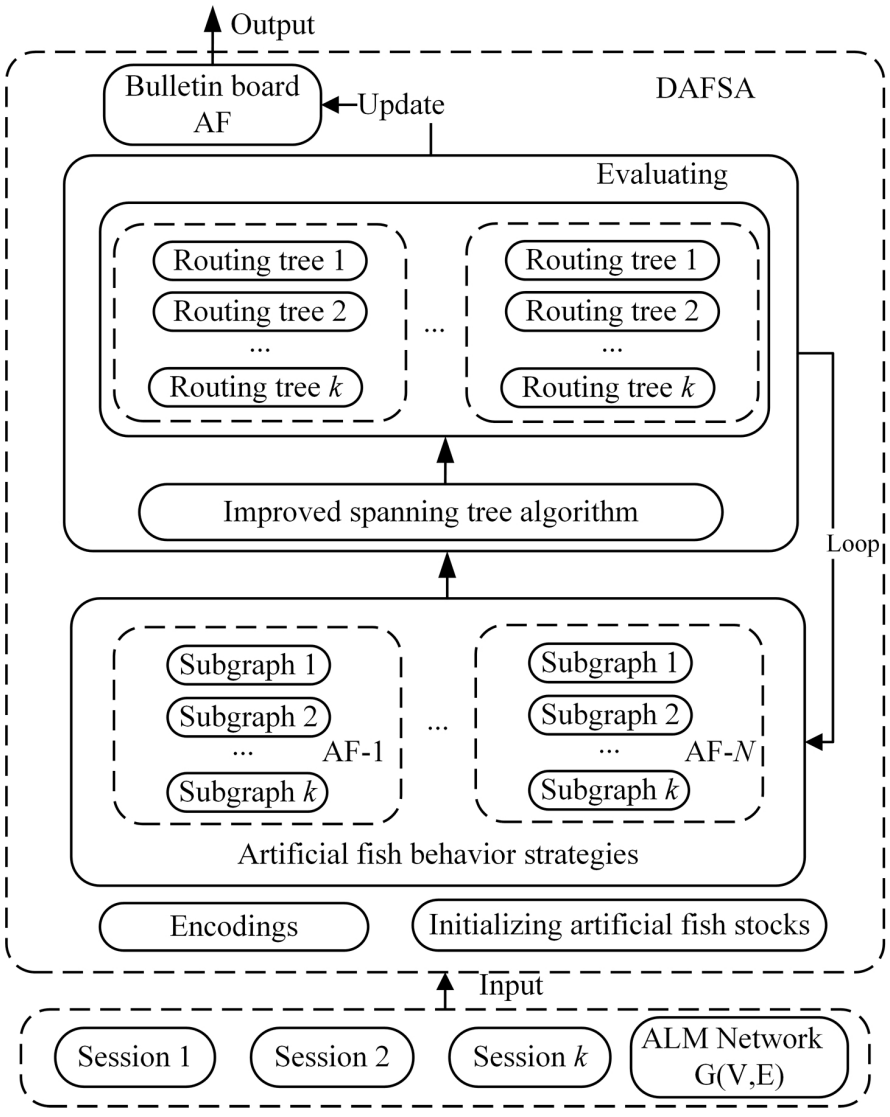


Figure 1. Optimization framework for multiple co-existing ALM routing trees.

4.1. Application of DAFSA in Multiple Co-Existing ALM Routing Trees

4.1.1. Encoding

The genotypes of the artificial fish are represented using matrix coding, where each row represents a Steiner node selection scheme for a multicast session, and this set of nodes

forms a complete subgraph $subG_j$. Equation (5) represents the code for artificial fish X (AF-X).

$$X = \begin{bmatrix} subG_1 \\ \cdots \\ subG_j \\ \cdots \\ subG_K \end{bmatrix} = \begin{bmatrix} x_{1,1} \cdots x_{1,i} \cdots x_{1,|V|} \\ \cdots \\ x_{j,1} \cdots x_{j,i} \cdots x_{j,|V|} \\ \cdots \\ x_{K,1} \cdots x_{K,i} \cdots x_{K,|V|} \end{bmatrix} \tag{5}$$

where each row has $|V|$ elements and each element can only be 0 or 1. If the complete subgraph $subG_j$ contains vertex i, then $x_{j,i} = 1$; otherwise, $x_{j,i} = 0$. All the elements in any $subG_j$ corresponding to the source and destinations should always be 1, as all potential complete subgraphs must contain the source and destinations.

4.1.2. Fitness Function

The fitness function is used to evaluate the quality of the artificial fish. To address the artificial fish that do not satisfy the constraints, a penalty value is introduced into the fitness function. The artificial fish that do not meet the constraints are eliminated in the iterative process when the fitness function takes a large value. This strategy helps to emphasize the importance of satisfying the constraint conditions and guides the algorithm to find suitable solutions in the search space. The formula for the *Fitness* is as follows:

$$Fitness = w_1 \sum_{k=1}^{K} f_1(T_k) + w_2 \sum_{k=1}^{K} f_2(T_k) + p \cdot \sum_{v_i \in V} Q(v_i) \tag{6}$$

$$Q(v_i) = \begin{cases} \sum_{k=1}^{K} Od_{v_i^{T_k}} - D_v, \sum_{k=1}^{K} Od_{v_i^{T_k}} > D_v \\ 0, \sum_{k=1}^{K} Od_{v_i^{T_k}} < D_v \end{cases} \tag{7}$$

where p in Equation (6) is the penalty factor and $Q(v_i)$ in Equation (7) represents the number of out-degree of node v_i that exceeds the degree constraint.

4.1.3. Behavior of Artificial Fish

The artificial fishes cooperatively search the solution space through the execution of behaviors. Specifically, optimal behavior is realized through a change in spatial position. As the solution space is discrete, the Hamming distance [28] is used to measure the distance between two artificial fishes. In this study, the behaviors used in the DAFSA were designed as follows:

(1) Randomly moving behavior

The encoding method used in this study is binary encoding. To implement this behavior, we only need to randomly flip the elements that do not correspond to the source and destinations used in the encoding matrix of AF-X, in the manner of $x_{j,i} = 1 - x_{j,i}$.

(2) Foraging behavior

Suppose the current position of an AF is X. Then, the AF randomly moves to a new position $X^{'}$. If the foraging behavior is successful (i.e., $Fitness(X^{'}) < Fitness(X)$), then the AF will randomly select $r(r \in [1, step])$ different elements between X and $X^{'}$ in X to cover the corresponding elements in $X^{'}$; otherwise, the AF will perform random movement.

(3) Following behavior

Following (or tail-chasing) is a behavior that imitates other AFs, especially those that perform well. Suppose that, within the visual range of AF-X, there are n AFs and X_p is the solution with the optimal fitness. Assume that the Hamming distance between X and X_p is equal to N_d, which means there are N_d elements in the encoding matrix of X that differ from the corresponding ones in X_p. The fitness function is satisfied if and only if $Fitness(X) > Fitness(X_p)$ and $n < N \times \delta$, in which case the following behav-

ior will be executed. The specific way in which this is executed is to randomly select $r(r \in [1, min(step, N_d)])$ elements from the above N_d elements in X_p to cover the corresponding elements in the AF, such that the distance between the two AFs will decrease and the similarity will increase.

4.2. Improved Spanning Tree Algorithm

During the decoding of an individual artificial fish, a tree that connects all the nodes needs to be obtained based on a complete graph. To make the constructed tree more stable with less delay under the condition that the out-degree constraint of the user node is satisfied, this study improves Prim's algorithm by comprehensively considering the delay and the instability, instead of using the edge weights, to weigh the order of joining in the minimum spanning tree. We used the contributions of the delay and the instability (*DIC*), calculated as follows:

$$DIC = min\{w_1(d(e_{v_i}) + d(v_i)) + w_2 \cdot l_{v_i} \cdot p(v_i)\} \tag{8}$$

In Equation (8), l_{v_i} represents the corresponding depth when node v_i joins the tree, $d(e_{v_i})$ represents the corresponding edge delay after node v_i is added to the tree, $d(v_i)$ represents the replication delay of node v_i, and $p(v_i)$ represents the probability of node v_i leaving a multicast session.

The node depth refers to the number of nodes that pass from the source node to a given node. The greater the depth of a node, the more unstable the data transmission path is, as the departure of any of the node's ancestor nodes will cause it to receive no data. Therefore, to increase the stability of the entire tree, the depth of each node should be kept as small as possible.

When the delays of the end-hosts are the same, the preference is to choose the end-hosts with a low leaving rate, as the nodes that are preferentially added to the tree are more likely to serve as transit nodes for data forwarding. In this way, the overall stability of the multicast tree can be increased. Similarly, when nodes have the same probability of leaving, the node with the shortest delay is selected first, which can reduce the overall delay. Smaller *DIC* nodes should be at the upper level of the multicast tree, in order to take full advantage of their low delay and low instability, thus improving the two target values of the ALM routing tree.

By borrowing ideas from Prim's algorithm, a preliminary ALM routing tree can be obtained that connects all the nodes in the complete graph. However, in the process of generating the tree, the phenomenon of node redundancy may occur due to improper selection of the Steiner node set; that is, non-destination nodes may appear at leaf nodes and are only involved in receiving data, not in forwarding it. The data transmission corresponding to this part has no practical significance and will only increase the delay and instability. These redundant branches need to be pruned, in order to ensure that the leaf nodes only contain the destination nodes of the session.

The improved spanning tree algorithm based on Prim's algorithm is constructed in Algorithm 1.

Algorithm 1: *DIC*-based tree generation algorithm

Data: Complete subgraph $subG(V, E)$
Result: ALM routing tree
Initialize an empty tree T, add the source node to T ;
while $|T|$!= $|V|$ **do**
  Generate alternative edges sets according to node in T;
  Calculate *DIC* of nodes in $\overline{T}$ and sort the them;
  Choose the *DIC* smallest node v_i in $\overline{T}$, add it and its corresponding edge e_{v_i} to T;
  The available out-degree of node v_i minus one;
  Update the collection T, $\overline{T}$;
Prune the tree T.

4.3. Algorithm Process

(1) The application layer network $G = (V, E)$ is input, and the relevant sources and destinations in K co-existing multicast sessions are specified;
(2) The algorithm-related parameters, such as the $popsize$, $visual$, $step$, try_number, δ, and p are set;
(3) Individual artificial fish execute the behavior strategy and obtain multiple Steiner node sets;
(4) The improved spanning tree algorithm is used to obtain the co-existing ALM routing trees corresponding to the multiple Steiner node sets obtained for the AF;
(5) The fitness of the AF individuals are evaluated by calculating the delay and instability of multiple co-existing ALM routing trees. The current best AF individual is compared with those recorded on the bulletin board, and if its fitness is better, the bulletin is updated;
(6) It is determined whether the algorithm termination condition has been met. If not, steps (3)–(6) are repeated; otherwise, the ALM routing tree corresponding to the multicast sessions is output.

5. Simulation Experiment Analysis

The DAFSA approach designed in this paper was written and tested in C++. The simulations were run on a computer (AMD Ryzen 7 5700U) with an 1.80 GHz Radeon GPU, 16.00 GB of RAM, and the Windows 7 (x64) operating system. The parameter settings were as follows: $popsize = 20$, $visual = 20$, $step = 6$, $try_number = 100$, $\delta = 0.5$, $p = 1$, $iteration = 200$, $D_v = 5$, $w_1 = 1$, and $w_2 = 0.0001$. These parameters are chosen experimentally. The detailed discussion on parameter settings will be given in Sections 5.4 and 5.5.

Figure 2 shows the IP network diagram. The circles in the diagram represent the user nodes, and the squares represent the router nodes. Each user node has two transmission parameters: the node replication delay and the departure probability. The weights between nodes represent the data transfer delays. Although the application layer multicast approach uses user nodes to transmit data, the underlying layer was still propagated through a routing node unicast approach. The edge delay between each pair of user nodes was obtained using the Dijkstra shortest path algorithm.

The session results for the optimization of four co-existing multicast sessions, each with one source node and eight destination nodes, are shown in Table 1, and the ALM routing trees obtained using the proposed algorithm are shown in Figure 3. For each ALM multicast tree corresponding to a multicast session, the out-degrees of all the nodes in Figure 3 satisfied the constraint. The out-degrees of nodes 8, 30, 38, and 24 were all 5, as the instability probabilities of nodes 8, 30, and 38 were very low (i.e., two orders of magnitude lower than those of the other nodes). Therefore, when constructing the ALM routing tree, these three nodes were preferentially selected as the transfer nodes for data transmission. The out-degree of node 24 was also 5, as the out-degrees of nodes 8, 30,

and 38 were allocated and because the data could only be forwarded through other nodes. However, the other nodes had a high probability of instability and, thus, were not suitable as transfer nodes. Therefore, the root node was directly used to transmit data to reduce the depth of the entire tree, thereby reducing the instability of the ALM routing tree.

Table 2 lists the delay and instability of the ALM routing tree for the four multicast sessions. As analyzed above, ALM trees a, b, and c used nodes 8, 30, and 38 as the transit nodes, respectively, which effectively reduced the instability. However, to satisfy the node out-degree constraint, the algorithm eventually selected some transit nodes (i.e., non-source and non-destination Steiner nodes), resulting in an increase in the link delay. In contrast, although ALM tree d (corresponding to multicast session 4) achieved a lower delay, it paid a higher price with its instability, which further illustrates that the algorithm made a certain trade-off between delay and stability.

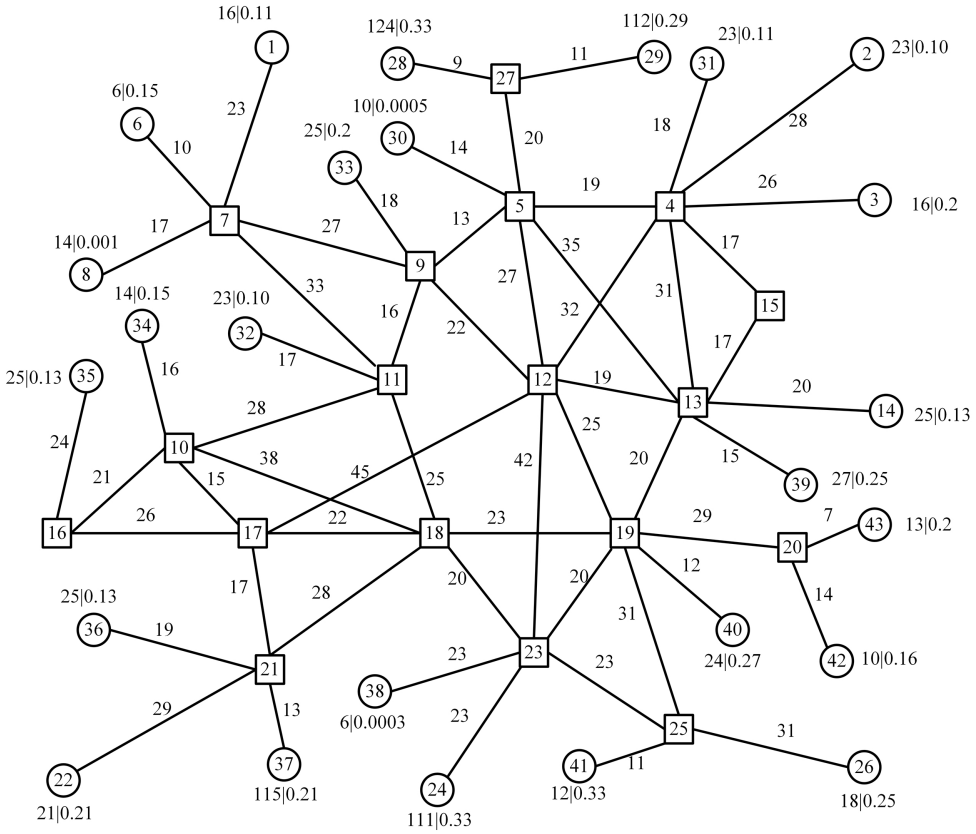


Figure 2. IP network instance.

Table 1. Multicast session information.

Multicast Session	Source	Destination
1	8	2, 14, 22, 24, 26, 28, 29, 31
2	2	1, 29, 31, 32, 41, 37, 36, 42
3	14	2, 6, 21, 31, 33, 35, 36, 40
4	24	28, 29, 32, 34, 36, 39, 40, 41

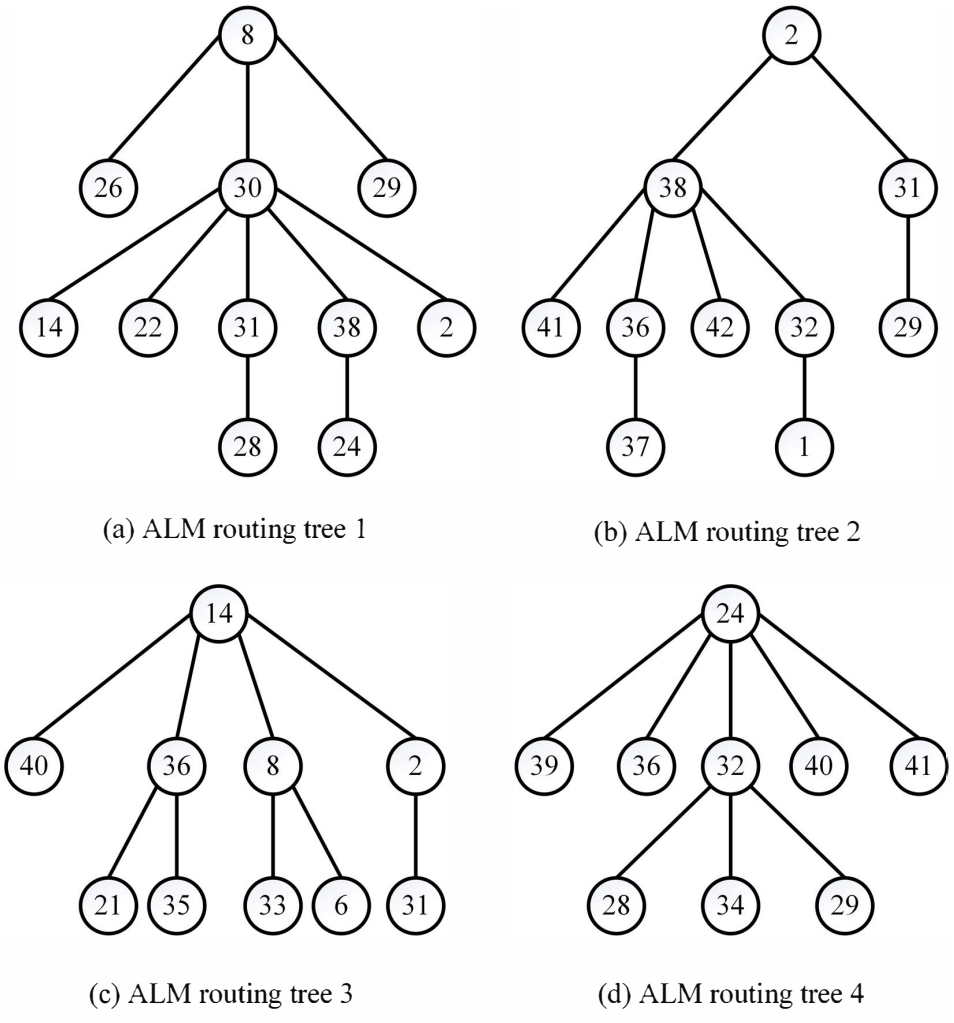


Figure 3. Obtained ALM routing trees.

Table 2. ALM routing tree information.

Multicast Session	Replication Delay (ms)	Link Delay (ms)	Instability	Total Delay (ms)
1	121	832	0.010	953
2	141	659	0.124	800
3	201	586	0.153	787
4	124	578	0.326	702
Total	587	2655	0.614	3242

In fact, the routing tree obtained with the algorithm was based on the application layer, and the actual data forwarding process used by the routing nodes to forward the data was in the form of an IP unicast. Taking ALM tree b from session 2 as an example, the actual data transmission process is shown in Figure 4. The transmission path between each pair of nodes was the transmission path with the lowest delay.

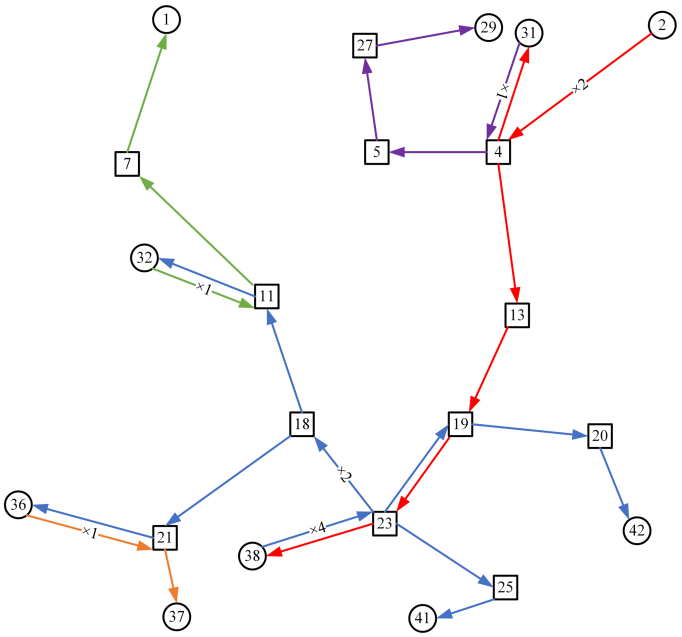


Figure 4. Actual data transmission path in multicast session 2. The red arrow indicates a path for the user node 2 to transmit data, the blue arrow indicates a path for the user node 38 to transmit data, the orange arrow indicates a path for the user node 36 to transmit data, and the green arrow indicates a path for the user node 32 to transmit data.

5.1. Comparison Between One-Off Optimization and Sequential Optimization

5.1.1. Comparison of Sequential Optimization That Does Not Consider the Out-Degree Constraint

In sequential optimization without considering the out-degree constraint, only one multicast session is optimized at a time, and the out-degree constraint on nodes is not considered. In one-off optimization, multiple multicast sessions are considered simultaneously to yield all multicast session transmission schemes. Different numbers of multicast sessions and destination nodes in each multicast session were set, and the node congestion under the two approaches described above was analyzed.

Figure 5 shows the fitting curves under sequential optimization and one-off optimization. The black dots indicate the out-degree violations under the two algorithms. The fitting surface shows that the node out-degree violation under sequential optimization increased exponentially with the number of multicast sessions and destination nodes, while one-off optimization presented no node constraint violations.

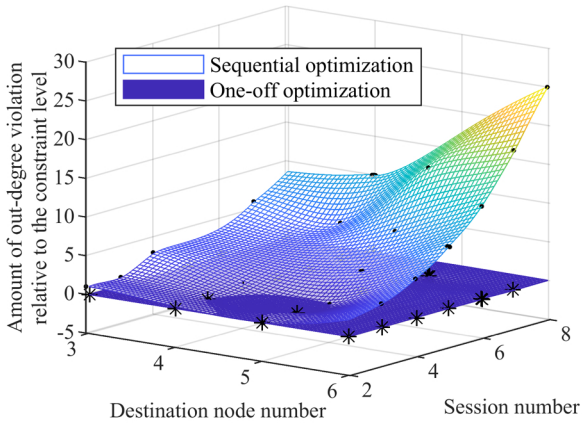


Figure 5. Violation of the out-degree constraint fitting surface under one-off optimization and sequential optimization approaches. The color represents the value that is out of bounds, the lighter the color the more out of bounds it is.

From the above analysis, as nodes 8, 30, and 38 were suitable transit nodes for forwarding data, their out-degree easily exceeded the constraint. For further analysis, we designed each session to contain five destination nodes and tested the out-degree of these three nodes under different numbers of multicast sessions.

Figure 6 shows that, in sequential optimization, when the number of multicast sessions was greater than four, the out-degree of node 30 exceeded the constraint, and when the number of multicast sessions was greater than six, the out-degree of node 38 exceeded the constraint; meanwhile, for node 8, the out-degree was basically maintained at 3 and was within the constraint. With an increase in the number of sessions, the out-degree of nodes 30 and 38 increased significantly. In addition, we found that the sum of the out-degree violation levels for nodes 30 and 38 and of all the nodes was equal, indicating that, under the considered experimental conditions, these two nodes caused the ALM routing tree to fail to satisfy the constraints.

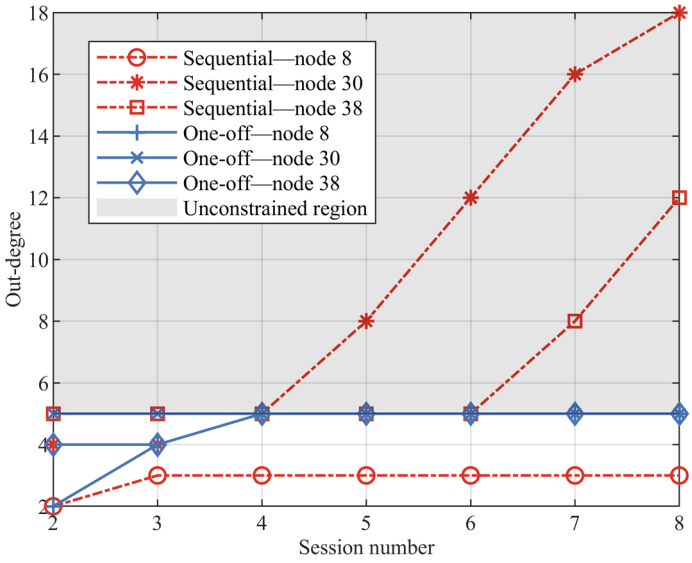


Figure 6. The out-degree in multiple multicast sessions with five destination nodes.

In contrast, in one-off optimization, when the number of sessions reached four, the out-degrees of nodes 8, 30, and 38 were all 5, equal to the critical constraint value. However, as the number of multicast sessions increased, the out-degree of these three nodes did not exceed the constraint. This indicates that the one-off optimization method can make full use of the out-degree of core nodes and obtain an optimal solution under the constraint conditions.

5.1.2. Comparison with Sequential Optimization While Considering the Out-Degree Constraint

The above experiments provided in-depth information on the impact of not introducing constraint processing technology in sequential optimization. Notably, sequential optimization can also consider the out-degree of a node as a constraint condition. We adopted the node out-degree reservation strategy; that is, each time the optimization of an ALM routing tree is completed, the out-degree of the corresponding node is purposefully reduced. In the next optimization of the ALM routing tree, we can choose only those nodes that still have a valid out-degree. However, this strategy may trap the entire ALM routing tree in a local optimal solution.

This occurs because, during the construction of the ALM routing tree, better nodes are initially selected. As the out-degree of such core nodes is exhausted, the subsequent ALM routing tree can use only other nodes with a greater delay and a greater instability, resulting in a sharp increase in the instability and delay of the whole tree.

Table 3 shows the optimization results obtained for four multicast sessions. The number of destination nodes for each session was five, and the out-degree of each node was

two. Although the one-off optimization method was not as good as the sequential optimization method in the construction of the first ALM routing tree, the results of the one-off optimization method showed a lower delay and instability when constructing the third and fourth ALM routing trees. When considering multiple co-existing ALM routing trees, the overall delay and stability were significantly better than those of the trees constructed using the sequential optimization method.

Table 3. Comparison of the one-off optimization and sequential optimization results.

Sessions	Sequentially Optimizing		One-Off Optimizing	
	Delay (ms)	Instability	Delay (ms)	Instability
1	671	0.0167	732	0.0514
2	501	0.135	501	0.135
3	588	0.235	494	0.17
4	580	0.376	555	0.326
total	2340	0.762	2282	0.682

As can be seen from Figure 7, in the first ALM routing tree, nodes 8, 30, and 38 were used, which decreased the delay and instability. However, in the third and fourth ALM routing trees, as the out-degrees of the selected core nodes 8, 30, and 38 had been used up, the other nodes were selected only to transmit data, resulting in significant increases in delay and instability in the two routing trees.

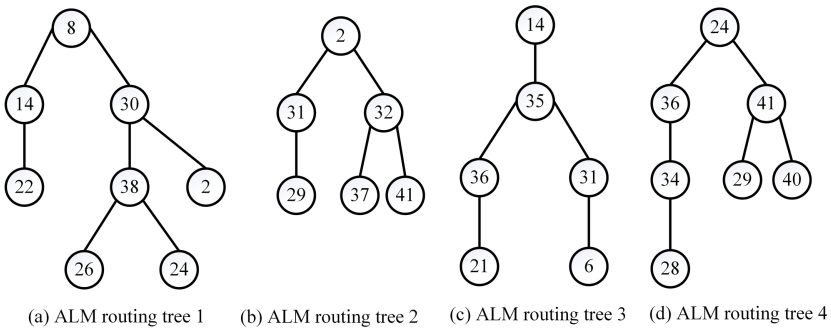


Figure 7. Routing trees obtained through sequential optimization.

In contrast, Figure 8 shows the results of one-off optimization. Through the rational distribution of nodes—for example, by using the out-degree of nodes 14 and 38 in ALM routing trees 3 and 4—the delay and instability of ALM routing trees 3 and 4 were reduced. Although this optimized allocation slightly increased the delay and instability of the first routing tree, it reduced the delay and instability of the multiple ALM routing trees as a whole. This result further clarifies the limitations of independently optimizing the ALM routing tree for each session. In contrast, the one-off optimization method used in this paper can more effectively optimize the overall performance.

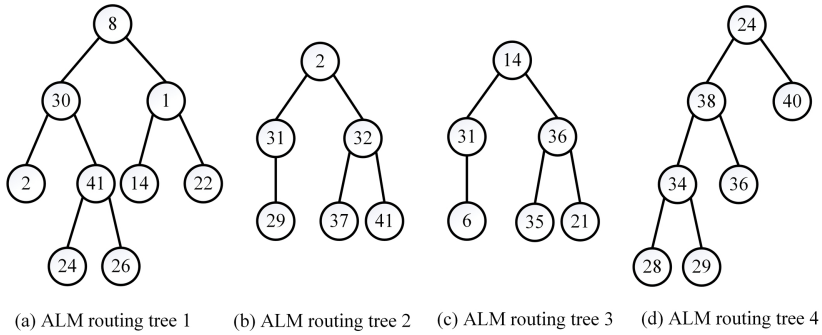


Figure 8. Routing trees obtained through one-off optimization.

5.2. Validation of the Penalty Function Mechanism

For the optimization of multiple co-existing ALM multicast routing trees, violating the out-degree constraint of nodes may cause failure of data transmission. Therefore, determining how to guide individual artificial fish to search in the feasible solution domain is highly important. In this study, a penalty mechanism was introduced to eliminate solutions that do not satisfy the constraints. In Figure 9, we compare the effect of the algorithm with and without the use of the penalty mechanism regarding the out-degree violation of nodes.

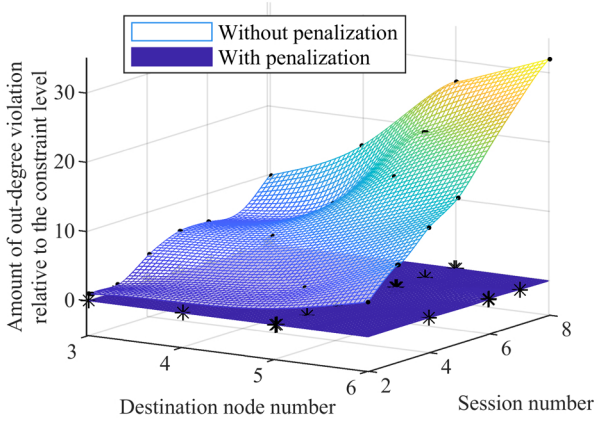


Figure 9. Fitted surfaces with and without penalty function for out-degree violations. The color represents the value that is out of bounds, the lighter the color the more out of bounds it is.

As the scale of the multicast sessions increased, among the results obtained with the algorithm without a penalty mechanism, a greater node out-degree violation indicated that a very large number of destination nodes needed to copy and forward the data in large quantities. When the performance limit of a node is exceeded, the end-host will be down, which will cause the session to fail. On the other hand, when the penalty mechanism was used, the results obtained with the algorithm did not include nodes exceeding the degree constraint. This demonstrates that the penalty mechanism can effectively solve the out-degree constraint problem. In particular, when the scale of multicast sessions increases, the algorithm with the penalty mechanism performed better in terms of reducing the out-degree violations of nodes.

5.3. Algorithm Convergence Analysis

The execution of various behaviors enables the artificial fish swarm to perform more flexible and diverse searches in the solution space. However, under some circumstances—especially when the problem is complex and the solution space is large—these behavior modes may cause the algorithm to converge slowly, and the algorithm may be prone to becoming trapped in local optimal solutions. To verify the convergence and accuracy of the algorithm for the optimization problem in this paper, we conducted an analysis of scenarios using networks containing 25, 50, and 75 randomly distributed user nodes. In these networks, four multicast sessions were input, where each multicast session contained one source node and eight destination nodes.

The randomness of the artificial fish swarm algorithm may make the algorithm unstable. To reduce the impact of randomness on the algorithm results, multicast optimization was performed for each network 50 times, and a box plot was generated to show the locations of the distribution centers of these results and the distribution range. As shown in Figure 10, for the networks with 25 and 50 user nodes, the box plot appears as a straight line; as such, the maximum and minimum values are the same, and there are no outlier values. When the network size increased to 75, the convergence stability of the algorithm decreased slightly, with a maximum value of 0.74014 and a minimum value of 0.73346,

comprising a difference of only 0.9%. This indicates that the algorithm had relatively good stability under different network sizes.

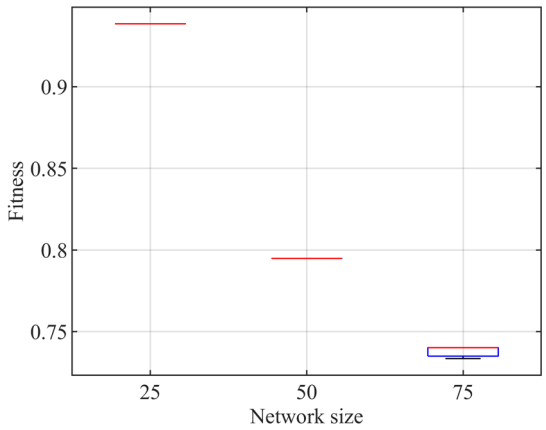


Figure 10. Fitness box plots under different network sizes. Black indicates the edge, red the upper quartile, and blue the lower quartile.

To further verify the convergence ability of the algorithm, the fitness values of the 50 results were summed and averaged, and the obtained iteration diagram is shown in Figure 11a. With network sizes of 25, 50, and 75 user nodes, the fitness value decreased rapidly at the beginning of the iteration, as the algorithm eliminated infeasible solutions. When the number of iterations reached approximately 10, the increase in the fitness value slowed down, as the solutions listed in the bulletin board already satisfied the constraints and its fitness function was low. Subsequently, as shown in Figure 11a–c, the algorithm approached the optimal solution as it iterated and converged at 81, 98, and 191 iterations, respectively.

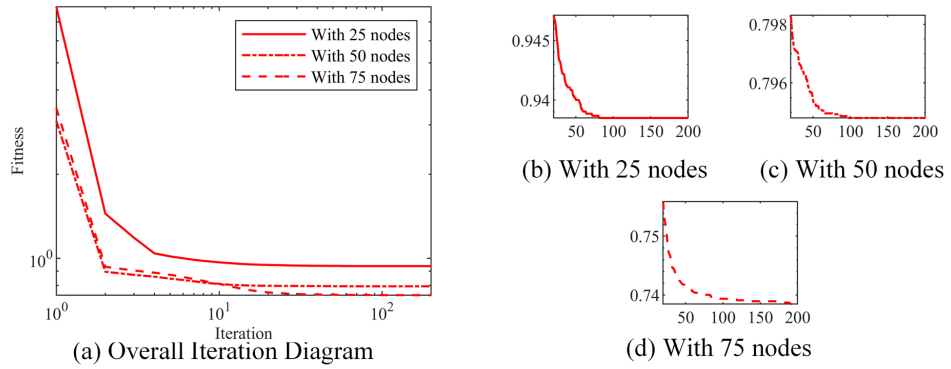


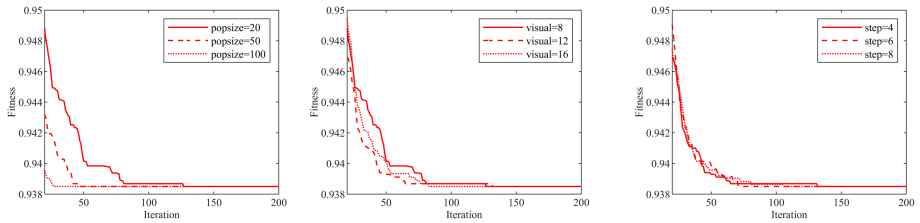
Figure 11. Algorithm evolution diagrams under different network sizes.

5.4. Parameter Sensitivity Analysis

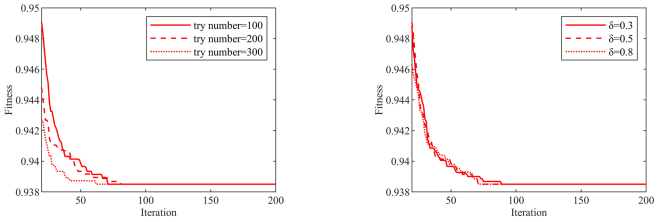
Swarm intelligence algorithms usually exhibit good adaptability. However, setting reasonable parameters is still a key task when using optimization algorithms. The appropriate selection of parameters can significantly improve the performance of the algorithm. The main parameters of the artificial fish swarm algorithm include the population size *popsize*, the field of view *visual*, the step size *step*, the number of attempts *try_number*, and the degree of congestion δ. Figure 12 shows the results of the algorithm from 20 to 200 iterations under different parameter settings.

Regarding the effect of the population size on the algorithm, as shown in Figure 12a, when the population size increased, the number of iterations needed for the algorithm to converge decreased. However, in each iteration, the number of AFs participating in the optimization search increased. Therefore, this parameter had no significant impact on the

overall convergence time. As can be seen from Figure 12b–e, setting different values for the other parameters affected only the iterative process of the algorithm and had a relatively insignificant impact on the final convergence result, which indicates that the algorithm is insensitive to parameter changes and has good robustness.



(a) Mean curves with different values of *popsize* (b) Mean curves with different values of *visual* (c) Mean curves with different values of *step*



(d) Mean curves with different values of *try_number* (e) Mean curves with different values of δ

Figure 12. Comparison of the effect of each parameter on the performance of the algorithm.

5.5. Selection of Weights

In this study, the optimization of the ALM routing tree involves two objectives, namely, the delay and the instability, with corresponding weights w_1 and w_2, respectively. The selection of these weights directly affects the performance of the algorithm and search results. In the experiments, the magnitude of the observed delay was much greater than that of the instability. This may have caused the delay to be too significant in the overall optimization process, leading to the contribution of instability being ignored. By adjusting the weights, the influence of the different objectives during the optimization process can be controlled.

The weights w_1 and w_2 can be determined in a number of ways. For example, the subjective judgment method [29], statistical method [30], and sensitivity analysis [31] can be used. However, neither of the first two methods is applicable; the subjective judgment method requires an expert's deep understanding of the problem and an accurate estimation of the contribution of each objective. Statistical methods require a large amount of supporting data; however, the resulting data of this problem are related to the number of source nodes, the destination nodes, the number of multicast sessions, and the network distribution and size, making this method costly. In contrast, sensitivity analysis, which directly assesses the impact of input parameters on the model output, is a simple and intuitive approach that requires less data and is easy to understand and implement.

Therefore, we used sensitivity analysis, and different weight combinations were used to cover the possible weight value ranges. The influences of these weights on the final optimization result were investigated, as shown in Figure 13. In general, there was an increase in the weight ratio (w_2/w_2) as the instability of the ALM routing tree gradually increased, while the total delay continuously decreased. This is due to the increase in the value of the weight w_1; that is, the contribution of the delay increased.

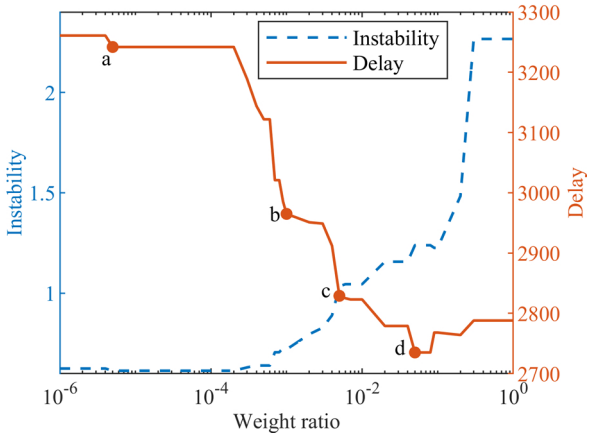


Figure 13. Comparison of results obtained under different weights.

In the process of gradually increasing the weight ratio, several inflection points appeared, indicated by the red points (a, b, c, and d) in the figure. These points are the turning points where the rate of decrease in the delay became slower, the rate of increase in the instability became greater, or both. Table 4 lists the results under the weight ratios corresponding to these points. By analyzing these points, we could obtain a locally optimal weight ratio; that is, a significant reduction in the instability or delay can be obtained without a significant increase in the delay or instability, respectively.

Table 4. Comparison of optimization results with different weights.

	Weight	Total Delay (ms)	Instability
	0:1	3633	0.567
a	0.00005:1	3242	0.614
b	0.001:1	2965	0.720
c	0.005:1	2823	1.019
d	0.05:1	2735	1.239
	1:0	2788	2.266

For example, consider the process from point a to point b: the delay was sharply reduced, while the instability increased slightly. Therefore, choosing point a will be less unstable than choosing any point between a and b, and the delay will not increase much as the delay does not change sharply. Meanwhile, the delay is lower when point b is selected, and the increase in instability is not significant.

When only the instability was optimized, the instability value reached 0.567. Meanwhile, when only the total delay was optimized, the total delay reached 2788 ms. These results provide a reference for weight selection under different optimization objectives, such that the algorithm can be flexibly adapted to the specific needs of a given application. Decision makers can consider the importance of each objective to the overall goal and determine the optimal combination of weights by considering the practicality, expertise, and relevant interests.

5.6. Analysis of Solution to the Routing Tree Problem for a Single Multicast Session

Although this study is optimizing the multiple co-existence application layer multicast routing tree structure problem, the method is equivalent to optimizing a single multicast session when we set the input to only one multicast session. To evaluate the performance of the proposed algorithm regarding the optimization of a single multicast session, a multicast session in which the number of source nodes was 8 and the numbers of destination nodes were 2, 3, 14, 26, 24, 37, 22, 35, 28, 29, and 31 was set up. The algorithm in this paper was compared with three single multicast session multi-objective optimization algorithms, namely, Cao's algorithm [7], the CL-S [8], and the VDM [9]; all three algorithms are for

single multicast sessions. Because of the differences between their optimization models and the one we formulated, we set the end-to-end latency constraint and the degree constraint in Cao's algorithm to 300 and 5 and made the CL-S take the form of transmission delay in our formulated optimization model and constructed the virtual distance for the VDM based on our objective. Such a modification ensures comparability but will not alter the performance. Table 5 shows that the proposed algorithm was superior to the three algorithms used for comparison, in terms of its total delay and instability.

Table 5. Comparison of ALM routing tree optimization results.

Algorithm	Total Delay (ms)	Instability
CL-S	1346	0.070
VDM	1359	0.073
CAO	1411	0.042
DAFSA	1075	0.016

6. Discussion

We investigate a key limitation of existing application layer multicast (ALM) routing optimization algorithms, namely, that these algorithms mainly focus on the optimization of individual multicast routing trees, whereas sequential one-by-one optimization is usually required when dealing with multiple co-existing multicast sessions. However, as the experimental results show, this sequential optimization approach can very easily lead to an excessive out-degree of user nodes suitable for forwarding data, triggering node congestion. And this phenomenon will be more serious with increases in the multicast session size, which will lead to a failure of data transmission in the session. Moreover, it is also difficult to make reasonable use of the node out-degree if we want to take into account the node out-degree constraints in the sequential optimization. The node out-degree reservation strategy mentioned in the previous section is a simple method of out-degree allocation, but it only ensures that all nodes can satisfy the constraints, which can easily lead to falling into a local optimum. Specifically, the routing tree optimized first performs well, while the performance of the routing tree optimized later gets worse. For multiple co-existing multicast sessions, such an allocation appears to be extremely unfair, and the performance of all multicast sessions cannot be optimized.

The discrete artificial fish swarming algorithm we designed takes multiple co-existing multicast sessions as a whole and achieves the optimization of the objective function values of multiple co-existing application layer multicast routing trees by continuously evolving artificial fish with higher fitness functions. Due to the introduction of a penalty function mechanism, this approach helps the algorithm filter the solutions that do not satisfy the node out-degree constraints and avoids session instability caused by node congestion. Experimental results show that the algorithm achieves satisfactory results, with trade-offs in node allocation across multicast sessions and a reduced overall delay and reduced instability. In addition, we note that our proposed algorithm is also effective in optimizing individual multicast sessions. Steiner nodes, selected by the DAFSA, have been proven to be very suitable as intermediate nodes for forwarding data, thus guaranteeing the performance of individual multicast sessions.

However, the present algorithm also has limitations. First, in setting the weights, when the network structure changes significantly, such as when the number of multiple co-existing application layer multicast sessions increases or the nodes of each session become complex, we are unable to find the optimal weighting parameter through multiple experiments because of the huge cost involved. This is mainly due to the fact that single-objective weighting methods are very sensitive to the choice of weights and are usually difficult to adapt to new contexts or changes in objectives. To overcome these problems, we propose to consider using a multi-objective decision-making approach [32] to optimize the relationship between multiple objectives more comprehensively. Further, this study deals

with node out-degree constraints using a penalty function, but the effect of the penalty function is highly dependent on the chosen penalty parameter, which adds to the complexity of the problem [33]. In solving these problems, an improved spanning tree algorithm ensures that the nodes in the current multicast session do not exceed the constraints. But how to consider multiple co-existing complete graphs and generate application layer multicast routing trees that satisfy the constraints to avoid the use of the penalty function remains a problem that requires in-depth research.

Moreover, the application layer multicast routing tree construction problem is usually dynamic in nature. That is, the network topology, multicast session members, etc., may change over time. The swarm intelligence algorithm has difficulty in dealing with dynamically changing problems at the time of application, and as the problems become more complex, its search space becomes larger, which can easily lead to a decreased search efficiency. In contrast, trained neural networks have the ability to generalize to unseen situations and can adapt to the complexity of the problem by learning patterns and features of the data without the need for explicit rules [34]. Therefore, we will include neural network methods in our future research.

7. Conclusions

In this paper, a one-off optimization method based on the discrete artificial fish swarm algorithm was proposed, which optimizes multiple co-existing application layer multicast sessions simultaneously, rather than optimizing them sequentially and independently. The contributions of this paper can be summarized as follows:

(1) The use of the DAFSA was proposed for the determination of multiple co-existing ALM routing trees. Compared with the sequential optimization algorithm, the multiple ALM routing trees obtained presented a better performance in terms of delays and stability. For the optimization of a single ALM routing tree, the proposed algorithm also outperformed other existing algorithms.
(2) In terms of degree constraint processing, a penalty function was used to ensure that the out-degree of nodes in the entire ALM routing tree did not exceed the constraint limit, effectively preventing the algorithm from obtaining infeasible solutions.
(3) In the evaluation section, we improved Prim's algorithm such that an ALM routing tree with a low delay and a low stability could be obtained from the subgraph. This process can be understood as the decoding process of the DAFSA such that this algorithm can be applied to the considered problem; in this way, a better ALM routing tree scheme can be iteratively generated.
(4) When processing application layer networks of various scales, the system quickly and reliably reached the optimal solution or a state close to the optimal solution. This stable convergence helps to improve the applicability and effectiveness of the algorithm in various network scenarios.
(5) Even under different parameter settings, the proposed method still reached a stable convergence state after a relatively short number of iterations, which helps to improve the reliability and applicability of the algorithm in practical problems.
(6) We have provided a variety of applicable weight determinations that can help provide more practically applicable decision support.

However, there are still some directions worthy of further exploration and improvement:

(1) When converting a multi-objective problem into a single-objective problem, we faced problems such as weight selection and information loss. To better retain the trade-off relationships between the objectives, a multi-objective optimization algorithm, such as the multi-objective genetic algorithm, can be introduced. Directly addressing multi-objective problems is a direction worth exploring.
(2) When dealing with constrained problems, a penalty function-based algorithm is often used. However, the introduction of a penalty function may complicate the calculation of the evaluation process, and determining the penalty factor is difficult.

For this problem, the use of an adaptive penalty function or other advanced constraint-processing techniques can be considered, in order to more effectively address the constraint conditions and reduce the complexity of the search space.

Author Contributions: Conceptualization, Y.L., N.W. and Q.L.; data curation, Y.L., N.W. and W.Z.; methodology. Y.L., N.W. and F.L.; validation, F.L., W.Z. and N.W.; investigation, N.W., Y.L. and W.Z.; writing—original draft preparation, N.W.; writing—review and editing, Y.L., N.W., Q.L. and F.L.; visualization, W.Z., N.W. and Q.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The code for the DAFSA can be provided upon request from the corresponding author, Feng Liu.

Conflicts of Interest: Author Wei Zhang was employed by the company China Mobile System Integration Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Deering, S.E. Multicast routing in internetworks and extended LANs. In Proceedings of the Symposium Proceedings on Communications Architectures and Protocols, Stanford, CA, USA, 16–18 August 1988; pp. 55–64.
2. Deering, S.E. *Host Extensions for IP Multicasting*; Stanford University: Stanford, CA, USA, 1988.
3. Chu, Y.h.; Rao, S.G.; Seshan, S.; Zhang, H. A case for end system multicast. *IEEE J. Sel. Areas Commun.* **2002**, *20*, 1456–1471. [CrossRef]
4. Su, J.; Cao, J.; Zhang, B. A survey of the research on ALM stability enhancement. *Chin. J Comput.* **2009**, *32*, 576–590.
5. Liu, Q.; Tang, R.; Ren, H.; Pei, Y. Optimizing multicast routing tree on application layer via an encoding-free non-dominated sorting genetic algorithm. *Appl. Intell.* **2020**, *50*, 759–777. [CrossRef]
6. Cao, J.; Su, J.; Wu, C. Modeling and analyzing the instantaneous stability for application layer multicast. In Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference, IEEE, Yilan, Taiwan, 9–12 December 2008; pp. 217–224.
7. Cao, J.; Su, J. Delay-bounded and high stability spanning tree algorithm for application layer multicast. *J. Softw.* **2010**, *21*, 3151–3164. [CrossRef]
8. Lin Hou, D.L.a.T. Algorithms of Spanning Tree Based on the Stability Probability and Contribution Link of Nodes for Application Layer Multicast. *J. Comput. Res. Dev.* **2012**, *49*, 2559–2567.
9. Mercan, S.; Yuksel, M. Virtual direction multicast: An efficient overlay tree construction algorithm. *J. Commun. Netw.* **2016**, *18*, 446–459. [CrossRef]
10. Lin, H.C.; Lin, T.M.; Wu, C.F. Constructing application-layer multicast trees for minimum-delay message distribution. *Inf. Sci.* **2014**, *279*, 433–445. [CrossRef]
11. Xiaofei, L.; Yiliang, X.; Xuance, S.; Demin, L. Application Layer Multicast Model with Low Delay and High Stability. *J. Donghua Univ.* **2023**, *40*, 74.
12. Li, D.; Wang, Z.; Wei, Y.; Yao, J.; Tan, Y.; Yang, Q.; Wang, Z.; Cao, X. Generation of Low-Delay and High-Stability Multicast Tree. *Comput. Mater. Contin.* **2023**, *76*, 561–572. [CrossRef]
13. Chen, H.; Wang, S.; Li, J.; Li, Y. A hybrid of artificial fish swarm algorithm and particle swarm optimization for feedforward neural network training. In Proceedings of the International Conference on Intelligent Systems and Knowledge Engineering 2007, Chengdu, China, 15–16 October 2007; Atlantis Press: Amstelkade, The Netherlands, 2007; pp. 1025–1028.
14. Sheverdin, A.; Monticone, F.; Valagiannopoulos, C. Photonic inverse design with neural networks: The case of invisibility in the visible. *Phys. Rev. Appl.* **2020**, *14*, 024054. [CrossRef]
15. Pan, Y.; Yu, Z.; Wang, L. Genetic Algorithm for Solving Application Level Multicast Routing Problems. *Mini-Micro Syst.* **2009**, *26*, 55–58.
16. Ma, X.; Tang, R.; Kang, J.; Liu, Q. Optimizing application layer multicast routing via artificial fish swarm algorithm. In Proceedings of the 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), IEEE, Changsha, China, 13–15 August 2016; pp. 115–120.
17. Vik, K.H.; Halvorsen, P.; Griwodz, C. Evaluating Steiner-tree heuristics and diameter variations for application layer multicast. *Comput. Netw.* **2008**, *52*, 2872–2893. [CrossRef]
18. Popescu, A.; Constantinescu, D.; Erman, D.; Ilie, D. *A Survey of Reliable Multicast Communication*; IEEE: Piscataway, NJ, USA, 2007.
19. Ljubić, I. Solving Steiner trees: Recent advances, challenges, and perspectives. *Networks* **2021**, *77*, 177–204. [CrossRef]
20. Dreyfus, S.E.; Wagner, R.A. The Steiner problem in graphs. *Networks* **1971**, *1*, 195–207. [CrossRef]
21. Garey, M.R. Computers and intractability: A guide to the theory of np-completeness, freeman. *Fundamental* **1997**, 498–500.
22. Prim, R.C. Shortest connection networks and some generalizations. *Bell Syst. Tech. J.* **1957**, *36*, 1389–1401. [CrossRef]
23. Jnr, J.K. On the shortest spanning subtree and the traveling salesman problem. *Proc. Am. Math. Soc.* **1956**, *7*, 48–50.

24. Singh, K.; Sundar, S. A hybrid genetic algorithm for the degree-constrained minimum spanning tree problem. *Soft Comput.* **2020**, *24*, 2169–2186. [CrossRef]
25. Majumder, S.; Barma, P.S.; Biswas, A.; Banerjee, P.; Mandal, B.K.; Kar, S.; Ziemba, P. On multi-objective minimum spanning tree problem under uncertain paradigm. *Symmetry* **2022**, *14*, 106. [CrossRef]
26. Yang, X.S. Nature-inspired optimization algorithms: Challenges and open problems. *J. Comput. Sci.* **2020**, *46*, 101104. [CrossRef]
27. Pourpanah, F.; Wang, R.; Lim, C.P.; Wang, X.Z.; Yazdani, D. A review of artificial fish swarm algorithms: Recent advances and applications. *Artif. Intell. Rev.* **2023**, *56*, 1867–1903. [CrossRef]
28. Norouzi, M.; Fleet, D.J.; Salakhutdinov, R.R. Hamming distance metric learning. *Adv. Neural Inf. Process. Syst.* **2012**, 1–9.
29. Paramanik, A.R.; Sarkar, S.; Sarkar, B. OSWMI: An objective-subjective weighted method for minimizing inconsistency in multi-criteria decision making. *Comput. Ind. Eng.* **2022**, *169*, 108138. [CrossRef]
30. Zheng, J.; Du, Z.; Zou, J.; Yang, S. A weight vector generation method based on normal distribution for preference-based multi-objective optimization. *Swarm Evol. Comput.* **2023**, *77*, 101250. [CrossRef]
31. Goodridge, W.S. Sensitivity analysis using simple additive weighting method. *Int. J. Intell. Syst. Appl.* **2016**, *8*, 27. [CrossRef]
32. Jiang, S.; Zou, J.; Yang, S.; Yao, X. Evolutionary dynamic multi-objective optimisation: A survey. *ACM Comput. Surv.* **2022**, *55*, 76. [CrossRef]
33. Liang, J.; Ban, X.; Yu, K.; Qu, B.; Qiao, K.; Yue, C.; Chen, K.; Tan, K.C. A survey on evolutionary constrained multiobjective optimization. *IEEE Trans. Evol. Comput.* **2022**, *27*, 201–221. [CrossRef]
34. Schuetz, M.J.; Brubaker, J.K.; Katzgraber, H.G. Combinatorial optimization with physics-inspired graph neural networks. *Nat. Mach. Intell.* **2022**, *4*, 367–377. [CrossRef]