IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 9 June 2023, accepted 14 July 2023, date of publication 24 July 2023, date of current version 17 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3298438

RESEARCH ARTICLE

Study on QoS Routing Optimization Algorithms for Smart Ocean Networks

TIAN JI[1], XIAOLEI CHANG[2], ZHENGJIAN CHEN[3], ZHENZHOU WU[1], RUIYU CHEN[4], CHENXI LI[1], AND CHEN XU[1]

[1]Research Institute of Tsinghua University in Shenzhen, Shenzhen 518000, China
[2]Tsinghua University, Beijing 100000, China
[3]Shenzhen Energy Group Company Ltd., Shenzhen 518000, China
[4]Shenzhen University, Shenzhen 518000, China

Corresponding author: Zhenzhou Wu (wuzhenzhou@qzcloud.com)

This work was supported in part by the Shenzhen Science and Technology Program under Grant SGDX20190917160803729.

ABSTRACT Nowadays, ocean networks gradually become increasingly important for communication among network entities such as maritime and sea-crossing users. However, ocean networks are highly dynamic because the communication links are composed of satellite and microwave links, which could be easily influenced by the environment such as local climate. Thus, network transmission in ocean networks faces great challenges, including low reliability and low efficiency. In this paper, we propose a smart ocean network architecture, where we use Software Defined Network (SDN) to perform unified management of the network, and Segment Routing (SR) to control data forwarding paths. In this way, we can control network flows and optimize network routing among diverse network entities in an ocean network. However, many Quality of Service (QoS) guaranteed applications in ocean networks, such as remote control, require lower delay. To guarantee the performance for such applications, we further propose QoS routing algorithms based on Fuzzy-Lagrange for the smart ocean networks architecture, where the optimization objective is to ensure service quality provided to users. According to experimental results, it is proved that, in comparison with the benchmark algorithms, the Fuzzy-Lagrange (FuzLag) algorithm proposed based on link fuzzification and Lagrangian Relaxation can improve the performance by 23% at most.

INDEX TERMS Ocean networks, control overhead, QoS routing, software defined network, segment routing.

I. INTRODUCTION

Both infrastructure and transmission technology of the ocean networks are developing fast nowadays, and ocean networks users raise an increasingly strict requirement for network transmission, especially for those delay-sensitive applications, such as remote control of industrial equipment, Virtual reality/Augmented Reality (VR/AR), etc. Therefore, ocean networks transmission should guarantee that the QoS satisfies user demands. However, the current network architecture model follows the best-effort principle. However, due to its dynamics, ocean networks are less likely to meet the user QoS requirements, such as low delay, high bandwidth, and high reliability [1]. For example, when cross-sea communications using microwaves are concerned, network transmission quality can be dramatically unstable because of a long transmission distance and climate changes can also greatly impact on the link bandwidth of microwaves.

The associate editor coordinating the review of this manuscript and approving it for publication was Tariq Umer.

Many previous studies on QoS routing focus on improving network performance experienced by users and guaranteeing that the relevant network demands be satisfied. To quantitatively evaluate QoS, some network service parameters are generally selected and used as evaluation metrics, such as packet loss probability, throughput, time delay, transmission feasibility, bandwidth, and jitter [2]. However, QoS routing in ocean networks is much more complex than traditional routing, because 1) ocean networks are much more heterogeneous, it is composed of different kinds of links including satellite, microwave, optical fiber, and WIFI, thus it is difficult to collect the global network information; 2) ocean networks are highly dynamic, as satellite/microwave links could fluctuate or fail frequently, and nodes could join or leave at any time causing topology changes.

As SDN emerges, brand new network architecture and model are constructed to decouple data and control planes, where the control plane collects the global topology information and make routing decisions, data plane forward packets according to routing information. In recent years, SR, as a data plane technology that carries forwarding instruction in packets, is prevalent to be combined with SDN, because SR can flexibly divert forwarding path and support incremental deployment in a scalable way.

With SDN and SR, all kinds of link information related to QoS can be collected from ocean networks, such as link delay, packet loss probability, link capacity, bandwidth, etc. After collecting this information, the SDN controller can make routing decisions with QoS routing algorithms, which many previous works have studied. However, SR carries forwarding instructions, which include an ordered list of segments along the path. The packet size will increase if the forwarding path is much different from the default path when considering the QoS requirements, leading to more significant control overheads. In ocean networks, the link capacity is usually limited due to the harsh ocean network environment. Thus, the network performance will degrade greatly if control overheads increase.

Taking control overheads into account make the QoS routing problem much more complex, and we need to achieve the tradeoff between the QoS performance and control overheads. In this study, we formulate the problem as an optimization problem, where the QoS performance is maximized and link delay, bandwidth, packet loss probability, and control overheads should satisfy the given constraints. We prove that the problem is Non-deterministic Polynomial Complete (NPC) by reducing it to a partition problem which is a famous existing NPC problem. Given its complexity, we devise a heuristic algorithm based on FuzLag, which can efficiently combine different metrics.

Finally, to evaluate the proposed algorithm, we carry out a comprehensive simulation using generated and real topologies, and compare it with the traditional Open Shortest Path First (OSPF) algorithm and another recently proposed QoS-aware Routing Scheme (QRS) algorithm. The simulation results show that our algorithms can significantly improve ocean networks performance.

The rest of this paper is organized as follows. We first discuss the related work in Section II. In section III, formulate the QoS routing model for smart ocean networks and prove its complexity. In Section IV, we propose FuzLag algorithm based on fuzzy logic and Lagrange relaxation, and in Section V, we evaluate our method through simulations. Finally, we conclude our paper in section VI.

II. RELATED WORKS

Many previous works studied QoS routing optimization problems. Masip et al. review current research situations of QoS based routing and introduce some persuasive and scientific programs that can solve challenges facing such routing [3]. Striegel et al. raise a multicast life cycle model for network multicast oriented QoS optimization to describe various conditions in a multicast process, including grouping dynamics, network dynamics, and traffic dynamics [4]. Regarding the relationship between Quality of Experience (QoE) and QoS in the field of video transmission, Chen et al. establish the correlation between QoS and QoE by summarizing the current QoE evaluation methods, feature analysis, advantages and other aspects of video transmission [5]. Scholars in the domain of wireless networks also profoundly investigate QoS optimization issues at multiple aspects. For example, Ehsan et al. review QoS routing optimization in wireless sensor networks, comprehensively introduce some of the existing energy-conservative routing technology, and also present some difficulties and challenges in improving routing protocols depending on QoS optimization in wireless multimedia sensor networks [6].

With more requirements put on the underlying network infrastructure, SDN has become a new hot research direction in recent years, especially QoS optimization in SDN attracts extensive attention from scholars. Guck et al. establish a mathematical model to solve QoS related issues of unicast routing in SDN. In addition to reviewing QoS routing algorithms for the past few years, they also present a novel 4-dimensional evaluation framework for QoS routing algorithms. This may contribute to a more scientific and reasonable evaluation of the performance of QoS optimization algorithms for unicast routing [7]. Karakus et al. summarize QoS optimization schemes in OpenFlow protocol based SDNs, involving varieties of technical solutions aimed at QoS optimization (e.g., a multimedia stream transmission mechanism, inter-domain routing, a resource protection mechanism, queue management, and scheduling algorithms) [8]. Considering that a default routing protocol is susceptible to network dynamics and external aggression in a scenario where SDN is combined with IoT, Guo et al. raise a QoS aware security routing protocol based on deep reinforcement learning to dynamically adjust routing strategies depending on the network status and further improving network transmission performance [9].

However, previous QoS routing algorithms can not directly fit into our work, because 1) ocean networks are more dynamic and heterogeneous than other networks; 2) the control overheads of segment routing in our architecture make the problem much more complex. Thus, we need to investigate the QoS routing problem in ocean networks.

III. PROBLEM FORMULATION

Communication of the smart ocean networks nodes may have their particular QoS requirements for transmission. For example, cross-sea communication users need to download some files or transfer videos; and their QoS demand may be a rather great transmission bandwidth accordingly. As for users performing offshore activities, the QoS demands may be low delay to ensure timeliness of data transmission at the time of performing urgent tasks. Certainly, QoS of many applications is involved with bandwidth, delay, jitter and packet loss probability, etc.

By the light of SDN and SR, the overall network status can be controlled and administered on the whole by a SDN controller. In terms of SR, it further lowers forwarding overhead of the forwarding devices such as a router, so that a router is only responsible for data package forward, but not storage of relevant forwarding tables or index search of forwarding tables. For diversified QoS requirements of data transmission in various nodes of the ocean networks, they may be satisfied to the greatest extent thanks to the control and coordination functions of SDN and efficient forwarding effects of SR. In this way, both network performance and users' QoE can be further boosted.

In a network at a certain scale, however, there are many data transmission paths from one node to another; different paths may correspond to different QoS indexes. For this reason, it is much more likely for data transmission effects and the result of meeting users' QoS requirements or not to differ dramatically. Therefore, a problem is raised in this research, that is, whether a rather preferable data transmission path meeting users' QoS demands can be identified by virtue of network management control of SDN and the forwarding technology of SR on one hand, and control overhead of SR data packages be also restricted as much as possible.

A. MODEL BUILDING

In the smart ocean network, its QoS routing scheme was abstracted to a simple undirected network topology $G = (V, E)$, where V refers to a set of nodes in the network, and E to a frontier set in this network. As for $\forall e_i \in E, e_i = \{n_{i1}, n_{i2}\}$ is selected to express nodes n_{i1} and n_{i2} to which the frontier e_i is connected.

The proposed QoS routing model used bandwidth, delay and packet loss probability as QoS indexes to analyze how users' QoS demands for these indexes can be respectively satisfied. For $\forall n_S \in N$ and $\forall n_D \in N$, a default transmission path $P_{default}$ exists between the source node n_S and the destination node n_D; and this path is expressed as follows: $P_{default} = \{e_S, e_{d1}, e_{d2}, \ldots, e_{dk}, e_D\}$. In a SR domain, there is no need to insert segment tables into data package headers if transmission between n_s and n_D follows $P_{default}$. In this case, no extra control overhead is incurred.

Therefore, six attributes described below can be achieved for $\forall e_i \in E$:

1) Cost $f(e_i)$. It represents the cost incurred by data flowing through the frontier e_i. According to the proposed model, the lower the cost of a frontier, the better the overall QoS performance of this frontier.
2) Bandwidth capacity C_i. It represents the maximum bandwidth allowed by the frontier e_i. Therefore, bandwidth passing through the frontier e_i should be no more than C_i according to this model. To facilitate modeling, bandwidth capacity for all frontiers C_i in the network is assumed to be consistent.
3) Current bandwidth overhead U_i. It represents the current bandwidth overhead of the frontier e_i, which is expressed in (1) below:

$$U_i = \frac{B_i}{C_i} \tag{1}$$

where, B_i stands for the bandwidth that has been occupied by the frontier e_i, and C_i for capacity of this frontier e_i. Under the circumstance that a user raises QoS requirements for bandwidth, bandwidth overhead U_i of this frontier should be no more than that required by the user.

1) Delay D_i. It represents delay of the frontier e_i. Regarding users with a requirement for low delay, the lower the delay of frontier e_i is, the higher the possibility for this frontier to meet their demands will be.
2) Packet loss probability pkl_i. It represents the packet loss probability of the frontier e_i. In terms of users having a requirement of low packet loss probability, a lower packet loss probability means that it is more likely for the frontier e_i to satisfy their demands.
3) Control overhead SL_i. It is incurred by forwarding data to the frontier e_i by virtue of SR. Specific to network flows of known source nodes and destination nodes, the following Equation (2) is written to express the control overhead [10].

$$\forall e_{mn} \in P^k_{div}, SL(f_k, m, n) = \begin{cases} 0, & \text{if } e_{mn} \in P^k_{default} \\ 1, & \text{otherwise} \end{cases} \tag{2}$$

Five attributes of the frontier in network $G = (V, E)$ are defined above. for known source node n_S and destination node n_D, a quintuple equation is written below to express $\forall e_i \in E$, that is:

$$tuple_i = \{f(e_i), \ U_i, \ D_i, pkl_i, \ SL_i \} \tag{3}$$

Specific to the source node n_S, the destination node n_D, and an identified path $P_{SD} = \{e_S, e_1, e_2, \ldots, e_D\}$ between them, the cost of this path can be evaluated by $f(e), e \in P_{SD}$, that is the cost of respective frontiers in the first place. As assumed, $f(e)$ is an additive parameter; and the cost of the path P_{SD} can be figured by (4) below:

$$f(P_{SD}) = \sum_{e \in P_{SD}} f(e) \tag{4}$$

For path P_{SD}, its cost is the sum of cost $f(e)$ of all frontiers along this path. When the cost of frontiers in the network is 1, the corresponding cost function is degenerated into the hop count of the path.

For a reason that the model should consider and meet users' QoS requirements, we need to determine whether the path P_{SD} can meet the specific QoS index notified by the source node. In this chapter, QoS requirements for bandwidth, delay and packet loss probability are primarily taken into account. Below, analyses are made on bandwidth, delay and packet loss probability of the path P_{SD}.

For the path $P_{SD} = \{e_S, e_1, e_2, \ldots, e_D\}$, its bandwidth overhead U_P is expressed in the following (5):

$$U_P = \max\left\{U_{e_s}, U_{e_1}, U_{e_2} \ldots, U_{e_D}\right\} \tag{5}$$

In line with the Wooden Bucket Theory and the law of network flow transmission, bandwidth overhead of a path depends on a frontier with the maximum bandwidth overhead among all frontiers along this path. If the bandwidth overhead of a path is excessively high, transmission of network flows may be restricted even though bandwidth overhead of other frontiers is rather low.

For the path $P_{SD} = \{e_S, e_1, e_2, \ldots, e_D\}$, its total delay D_P is written into the following (6):

$$D_P = \sum_{e \in P_{SD}} D_e \tag{6}$$

It is the delay caused by transmission in path P_{SD}, equal to accumulative delays of all frontiers along this path.

For the path $P_{SD} = \{e_S, e_1, e_2, \ldots, e_D\}$, the corresponding packet loss probability is figured out by the following (7):

$$pkl_P = 1 - \prod_{e \in P_{SD}} (1 - pkl_e) \tag{7}$$

To facilitate modeling and subsequent path computation, multiplication selected for packet loss probability calculation is substituted by addition. Regarding the packet loss probability of frontier e, the following formula (8) is obtained through adjustment:

$$pkl_e^* = -\ln(1 - pkl_e) \tag{8}$$

Therefore, equation (7) expressing the pkl_P of the path can be rewritten into the following (9):

$$pkl_P^* = \sum_{e \in P_{SD}} pkl_e^* \tag{9}$$

The above equation (9) can be utilized to calculate bandwidth overhead U_P, delay D_P and packet loss probability pkl_e^* of any path P_{SD}. Provided that the forwarding technology SR is adopted, control overhead incurred by SR data forwarding is inevitable in the event of $P_{SD} \neq P_{default}$. Therefore, control overhead SL_P of the path P_{SD} needs to be figured out as well by the following equation (10) and (11) [10]:

$$\forall e_{ij} \in E,\ a_{ij} = \begin{cases} 1, & \text{if } e_{ij} \in P \\ 0 & \text{, otherwise} \end{cases} \tag{10}$$

$$SL_k = \sum_{e_{ij} \in E} a_{ij} \times SL\left(f_k, i, j\right) \tag{11}$$

It should be noted that control overhead of each frontier in the network may change with variations in n_S and n_D, because as n_S and n_D change, the default path $P_{default}$ of new n_S or n_D differs from the previous one. In this scenario, the above equation should be used again to estimate SL_i of each frontier, and recalculate SL_P for the corresponding path P_{SD}.

Thus, the model becomes capable of figuring out specific values of bandwidth overhead, the total delay, the packet loss probability and the control overhead of any end-to-end path in network $G = (V, E)$. If $\forall n_S \in N$ at the source node, a user requests a path P_{SD} which is provided by the controller and arrives at the destination node in conditions of $\forall n_D \in N$ and $n_S \neq n_D$. In addition, the user raises the following QoS requirements:

$$D_P \leq \Delta_{delay}; \quad U_P \leq \Delta_{band}; \quad pkl_P^* \leq \Delta_{pkl} \tag{12}$$

This means that in network $G = (V, E)$, there may exist multiple paths satisfying users' QoS requirements of Δ_{delay}, Δ_{band} and Δ_{pkl} as well as the constraint λ over the control overhead of a path. If the number of these paths is k, a set of feasible paths can be denoted as $\boldsymbol{P} = \{P_1, P_2, \ldots, P_k\}$. Without a doubt, it is also possible that no paths meeting users' QoS requirements can be found, in which case, $\boldsymbol{P} = \emptyset$.Under the circumstance that the constructed model is aimed at finding one and only one path P_{div}, parameter z_i is designed to signify whether the path $P_i \in \boldsymbol{P}$ is selected or not. Moreover, z_i is expressed in the following (13):

$$z_i = \begin{cases} 1, & \textit{if } P_i \text{ selected as the path} \\ 0, & \textit{otherwise} \end{cases} \tag{13}$$

As the proposed model always selects only one path P_{div} from P, a set of feasible paths, z_i also satisfies the following equation (14):

$$\sum_{P_i \in \boldsymbol{P}} z_i = 1 \tag{14}$$

Specifically, optimization objectives are modelled as follows to provide users with a path P_{opt} with the least cost, satisfy their QoS requirements of Δ_{delay}, Δ_{band} and Δ_{pkl}, and optimize the control overhead incurred by data package transmission of SR.

Objective 1: Regarding users at the source node $\forall n_S \in N$ in network $G = (V, E)$, they need to establish connections with the destination node $\forall n_D \in N$ with $n_S \neq n_D$. Besides, QoS requirements of $D_P \leq \Delta_{delay}$, $U_P \leq \Delta_{band}$ and $pkl_P^* \leq \Delta_{pkl}$ are raised by them. In addition to identifying a path $P_{SD} \in \boldsymbol{P}$ with rather low control overhead for users, the model should be able to realize the following goal as well:

$$\min f\left(P_{SD}\right) \tag{15}$$

Corresponding constraints are written below:

$$U_P \leq \Delta_{band} \tag{16}$$
$$D_P \leq \Delta_{delay} \tag{17}$$
$$pkl_P^* \leq \Delta_{pkl} \tag{18}$$
$$SL_P \leq \lambda \tag{19}$$
$$U_P \leq U_{threshold} \tag{20}$$
$$P_{SD} \in \boldsymbol{P} \tag{21}$$

$$z_{SD} = 1 \text{ and } \sum_{P_i \in \boldsymbol{P}} z_i = 1 \tag{22}$$

In equation (15), $\min f(P_{SD}) = \min \sum_{e \in P_{SD}} f(e)$ expresses the minimum cost function $f(P_{SD})$ figured out for the path P_{SD}. As for Δ_{band}, Δ_{delay} and Δ_{pkl} requested by users for the obtained path P_{SD}, they are expressed in (16), (17) and (18). According to equation (19), the path P_{SD} figured out needs to meet a certain constraint λ over the control overhead. Here, λ is a parameter generated by a controller in line with relevant network status information. In equation (20), $U_{threshold}$ is the bandwidth overhead threshold in this network, signifying that the bandwidth overhead U_P of the selected path P_{SD} should not exceed $U_{threshold}$ for the sake of network performance. Generally, it is impossible that bandwidth overhead of a network approaches or reaches 100%, which may lead to serious network congestion and lower the transmission quality.. As expressed in equation (21), the selected path P_{SD} must belong to $\boldsymbol{P}$, a set of feasible paths from the source node n_S to the destination node n_D. Finally, one and only one path is allowed to be selected from $\boldsymbol{P}$, as expressed in (22).

Variable parameters involved in the proposed model here are listed in Table 1:

TABLE 1. Symbols of QoS assurance optimization and their meanings.

Parameters	Meanings
$G = (V, E)$	Network topology formed by nodes and
$P_{default}$	Default transmission path
$f(e_i)$	Cost function of the frontier e_i
C_i	Capacity of the frontier e_i
U_i	Bandwidth overhead of the frontier e_i
D_i	Delay of the frontier e_i
pkl_e	Packet loss probability of the frontier e_i
$f(P_{SD})$	Cost of the path P_{SD}
U_P	Bandwidth overhead of the path P_{SD}
D_P	Delay of the path P_{SD}
pkl_P	Packet loss probability of the path P_{SD}
SL_P	Control overhead incurred by path P_{SD}
Δ_{delay}	Users' QoS requirements for delay
Δ_{band}	Users' QoS requirements for bandwidth
Δ_{pkl}	Users' QoS requirements for packet loss probability
$\boldsymbol{P}$	All paths satisfying users' QoS requirements
z_i	Parameters enabling the number of paths selected to be 1
λ	Constraint over the control overhead of a path
$U_{threshold}$	Bandwidth overhead threshold for frontiers

B. COMPLEXITY ANALYSES

As described above, a QoS assurance model has been established with an aim to find a path that realizes SR based control overhead optimization on the premise of satisfying users' different QoS requirements. However, the objective function used to solve this model is a NP-Hard problem. Subsequently, corresponding proof will be presented in detail.

A recognized NP-Hard problem is firstly introduced as follows:

Lemma 2: (2)-partition problem) A multiset S is divided into two subsets S_1 and S_2, in which case, $S_1 \cup S_2 = S$, and $S_1 \cap S_2 = \emptyset$; and a sum of all elements in S_1 is equal to that of all elements in S_2. The 2-partition problem is NP-Hard.

By attributing the original problem to 2-partition, the objective function used to solve the model is proved to be a NP-Hard problem. As for bandwidth constraints of the path, this is a process of figuring out the maximum value of all frontiers along this path. Considering this, network topology pruning should be performed before calculating the path. In this way, it is expected to remove all frontiers exceeding the bandwidth overhead from this network and thus satisfy QoS requirements of users. In a proving procedure, constraints of bandwidth overhead are neglected and the number of constraints is lowered to 3, that is time delay, packet loss probability and control overhead of the path.

Theorem 3: The objective function is a NP-Hard problem.

Proof: For a reason that the objective function relates to multiple concrete computing methods, abstraction of it together with the network model is carried out, which is also proved based on problems with dual constraints. As assumed, there exist two different additive weights $w_1(e)$ and $w_2(e)$ for each frontier e in network $G = (V, E)$, where $w_1(e) > 0$ and $w_2(e) > 0$. Besides, n_S and n_D are designed to be source and destination nodes; and there are two positive numbers W_1 and W_2 at the meantime. Hence, the problem turns to determining whether a path $P_{SD} = \{e_S, e_1, e_2, \ldots, e_D\}$ resulting in $w_1(p) < W_1$ and $w_2(p) < W_2$ exists.

In the first place, an instance of the 2-partition problem is introduced; and the corresponding set can be written as $S = \{s_1, s_2, \ldots, s_n\}$. A network possessing $(n+1)$ nodes and $2n$ frontiers is then built. For respective nodes i and $i+1$ in this network, they are formed by connecting two frontiers, as shown in the following Fig. 1:

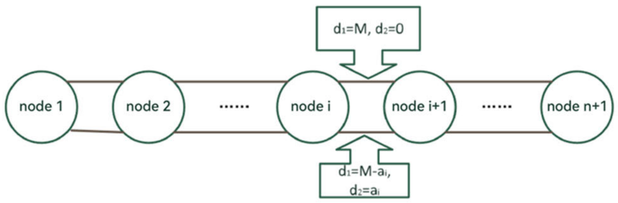


FIGURE 1. An example of the 2-partition problem.

Let $A = \sum_{i=1}^{n} s_i$ and $B = 2nA$. As two frontiers exist between two arbitrary nodes, the first weight of the first frontier between nodes i and $i+1$ is denoted as B, while its second weight as 0. In terms of their second frontier, first and second weights are respectively written into $B - s_i$ and s_i.

The following problem of path planning under two constraints is taken into consideration:

$$w_1(p) \leq nB - \frac{A}{2} \tag{23}$$

$$w_2(p) \leq \frac{A}{2} \tag{24}$$

where, p stands for a path from Nodes 1 to n. In this figure, it is clear that the sum of all weights of a frontier at either upper half or lower half part between nodes i and $i+1$ is B, that is,

$$w_1(i, i+1) + w_2(i, i+1) = B \tag{25}$$

Therefore, any path p between nodes 1 and $n+1$, a sum of its two weights can be expressed in the following (26):

$$w_1(p) + w_2(p) = nB \tag{26}$$

Because of $w_1(p) \leq nB - \frac{A}{2}$, we have

$$w_2(p) \geq \frac{A}{2} \tag{27}$$

In the event of $w_2(p) \leq \frac{A}{2}$, there exists

$$w_2(p) = \frac{A}{2} \tag{28}$$

Considering a frontier between nodes i and $i+1$, the weight $w_2(i, i+1)$ is 0 or s_i. Therefore, a subset S_2 of the original set S can be found; and the total sum of all elements in this subset is $\frac{A}{2}$, thus solving this problem. This is a living example of settling the 2-partition problem.

On the contrary, if there is a subset S_2 in the original set S and the total sum of all elements in this subset is $\frac{A}{2}$, and in the event of $s_i \in S_2$ for nodes i and $i+1$, the second frontier should be selected; otherwise, we select the first one. Regarding the path p obtained in this way, we have

$$w_2(p) = \frac{A}{2} \tag{29}$$

As

$$w_1(p) + w_2(p) = nB \tag{30}$$

there exists

$$w_1(p) = nB - \frac{A}{2} \tag{31}$$

In this case, a solution to the objective function can be obtained as long as a solution to the 2-partition problem is figured out. Therefore, the objective function is attributed to a 2-partition problem, that is a NP-Hard problem. With regards to the problem proposed in this study, it has three constraints of delay, packet loss probability and control overhead. Such a problem with three constraints can be also attributed into that with two constraints, that is a dual-constraints problem. Furthermore, such a problem of dual constraints has been proved to be NP-Hard. Hence, the objective function solved in this paper is also a NP-Hard problem. End of proof.

IV. FUZLAG ALGORITHM BASED ON FUZZY LOGIC AND LAGRANGIAN RELAXATION

In the above context, the network model was described and the corresponding objective function was defined. Besides, the proposed problem is proved to be NP-Hard. On this basis, a FuzLag algorithm is put forward in this paper based on fuzzy logic and Lagrangian Relaxation with an expectation to figure out a feasible solution approaching the optimal solution of the model's objective function. In this way, we endeavor to provide users with a feasible path that meets their QoS requirements, realizes control overhead optimization, and is close to the corresponding optimal path. The general algorithm is presented as shown in Table 2 below:

TABLE 2. Pseudocodes of the FuzLag algorithm.

```
Algorithm 4.1 Framework of Lagrangian Relaxation algorithm based on fuzzy logic
    Input: The number of network nodes ($N$), bandwidth overhead $U$ of all frontiers, delay $D$, packet loss probability $pkl$, source node $n_S$ and destination node $n_D$ and their default path $P_{default}$, constraint $\lambda$ over control overhead, bandwidth overhead constraint $\Delta_{band}$, time delay constraint $\Delta_{delay}$, packet loss probability constraint $\Delta_{pkl}$, bandwidth overhead threshold $U_{threshold}$, iteration number $iter$, and $\varepsilon \geq 0$.
1.  Construct an adjacency matrix $\boldsymbol{U}$ for bandwidth overhead $U$, an adjacency matrix $\boldsymbol{D}$ for delay $D$, an adjacency matrix $\boldsymbol{PKL}$ for packet loss probability $pkl$, and an adjacency matrix $\boldsymbol{SL}$ for control overhead $SL$.
2.  Initialize the frontier cost adjacency matrix f=0
3.  for i = 1, …, $N$ do
4.      for j=1, …, N do
5.          if $\boldsymbol{U}(i,j) \geq \Delta_{band}$ || $\boldsymbol{U}(i,j) \geq U_{threshold}$
6.              $\boldsymbol{U}(i,j) = +\infty$ //Perform pruning for frontiers with bandwidth beyond that required by users or the corresponding threshold
7.              $\boldsymbol{f}(i,j) = +\infty$
8.  $path = function(\boldsymbol{f}, \boldsymbol{PKL}, \boldsymbol{D}, \boldsymbol{SL}, n_S, n_D, \Delta_{pkl}, \Delta_{delay}, iter)$ //Utilize input parameters to figure out a feasible path meeting relevant constraints for the source node
9.  return $path$ //Return to a concrete path
```

Algorithm 4.1 gives an overall framework of the Lagrangian Relaxation algorithm based on fuzzy logic. Firstly, input of this algorithm consists of bandwidth overhead U of nodes and frontiers in the network $G = (V, E)$ and all frontiers acquired by a controller, and their delay D, packet loss probability pkl, source node n_S and destination node n_D and the default path $P_{default}$ between n_S and n_D, constraint λ over control overhead of the generated path, bandwidth overhead constraint Δ_{band}, time delay constraint Δ_{delay}, packet loss probability constraint Δ_{pkl}, bandwidth overhead threshold $U_{threshold}$, and iteration number $iter$. Bandwidth overhead U, time delay D, packet loss probability pkl and the default path $P_{default}$ of all frontiers are adopted to construct 4 adjacency matrixes of network, that is an adjacency matrix $\boldsymbol{U}$ for bandwidth overhead, an adjacency matrix $\boldsymbol{D}$ for delay, an adjacency matrix $\boldsymbol{PKL}$ for packet loss probability, and an adjacency matrix $\boldsymbol{SL}$ for control overhead. Subsequently, the frontier cost function $\boldsymbol{f}$ is initialized.

Regarding frontiers in nonconformity with relevant conditions in the network, pruning is carried out. According to computational formulas of path bandwidth overhead and delay during modeling, computing of bandwidth overhead is a process of taking its maximum value. For this reason, if bandwidth overhead of a frontier along a path exceeds its constraint, bandwidth overhead of this path must also go beyond its constraint. Under the circumstance that delay of frontier along a path is greater than its constraint during additive computing of delay, delay of this path also exceeds its constraint inevitably because all delays of frontiers along this path are positive. Therefore, specific to frontiers with bandwidth overhead exceeding users' QoS requirements for bandwidth or $U_{threshold}$, other frontiers of the adjacency matrix $\boldsymbol{U}$ and the cost function $\boldsymbol{f}$ configured for bandwidth overheads corresponding to the former frontiers are set at rather large values. Similarly, large values are assigned to frontiers of the adjacency matrix $\boldsymbol{D}$ and the cost function $\boldsymbol{f}$ established for delay corresponding to other frontiers with delay exceeding that required by users. By means of pruning, both computing efficiency and performance of subsequent frontier cost computing and path finding functions can be improved. When unqualified frontiers are undergoing pruning, the frontier cost function is computed at the same time. Here, the *fuzzy* function used to calculate the frontier cost uses packet loss probability $\boldsymbol{PKL}(i, j)$ and delay $\boldsymbol{D}(i, j)$ that correspond to this frontier as its input, so that a specific frontier cost $\boldsymbol{f}(i, j)$ is obtained. Moreover, the *fuzzy* function computing is based on the fuzzy logic, which will be specifically described in Section IV-A.

When the cost of all frontiers in the entire network has been obtained by the *fuzzy* function, pruning of unqualified frontiers is also completed. After that, the Lagrangian Relaxation algorithm can be utilized to input the adjacency matrix f of cost, the adjacency matrix $\boldsymbol{D}$ of delay, the adjacency matrix $\boldsymbol{PKL}$ of packet loss probability, the adjacency matrix $\boldsymbol{SL}$ of control overhead, the source node n_S, the destination node n_D, users' QoS requirement Δ_{pkl} for packet loss probability, users' QoS requirement Δ_{delay} for delay, and the maximum iteration number $iter$. At last, a feasible path from n_S to n_D can be figured out; and in this path, not only are users' QoS requirements satisfied, but control overhead incurred by data package forwarding is optimized. As for specific path calculation procedures, they will be elaborated in subsequent sections.

A. A FUZZY INFERENCE SYSTEM BASED ON QOS DEMANDS

For any frontier e in a network, the bandwidth overhead constraint as a maximization parameter has been satisfied during pruning, because this frontier has two additive parameters of delay and packet loss probability. The constructed model aims to find the best possible path conforming to users' QoS demands. Therefore, the frontier cost as an objective function should be defined to take both delay and packet loss probability into comprehensive consideration. As described above, estimations on delay and packet loss probability relating to frontiers cannot be readily defined in accurate mathematical ways. Considering this, the Mamdani fuzzy inference system was introduced and utilized to figure out specific costs depending on delay and packet loss probability of each frontier in combination with membership functions and fuzzy rules. Specific values of the frontier cost in the network are further incorporated in subsequent path computing schemes on one hand; and on the other hand, they also serve as criteria for QoS evaluation of network frontiers.

1) FUZZY VARIABLES

In this fuzzy inference system, controllers select, according to the network topology collected and packet loss probability and delay of each frontier, the packet loss probability pkl_e and the delay D_e of any frontier e as the input fuzzy variables. Moreover, pkl_e is set within a range of $\left[0, pkl_{max}\right]$. The corresponding fuzzy set can be divided into 5 categories of "e with quite low packet loss probability", "e with very low packet loss probability", "e with moderate packet loss probability", "e with very high packet loss probability", and "e with quite high packet loss probability". In terms of the frontier D_e, it has a range of $[0, D_{max}]$, where D_{max} refers to the upper bound of frontier delay in the network, that is the delay corresponding to a frontier with the highest delay in general cases. Accordingly, the fuzzy set consists of 5 categories, that is "e with quite large delay", "e with very large delay", "e with moderate delay", "e with very low delay", and "e with quite low delay". The inputted value is cost function $f(e)$ in a range of [0, 1]. In this case, the corresponding fuzzy set has 5 categories as well, that is "e with quite poor QoS performance", "e with very poor QoS performance", "e with moderate QoS performance", "e with quite very good QoS performance", and "e with quite good QoS performance".

2) MEMBERSHIP FUNCTION

In this fuzzy inference system, 5 fuzzy sets are obtained from perspectives of the packet loss probability, the delay and the QoS performance. This signifies that there are 5 membership functions accordingly. Regarding membership function selection, Gaussian curves are comparatively smooth and curves of membership functions are in normal distribution; besides, not only is the goodness of fitting also preferable as far as the packet loss probability and the delay are concerned in many cases of the network, the corresponding anti-interference performance is strong. Therefore, membership functions of packet loss probability fuzzy sets are defined by 5 Gaussian curves in the event of $\sigma = 0.002$ and $\mathrm{c} = 0, \frac{\mathrm{pkl}_{\mathrm{max}}}{4}, \frac{\mathrm{pkl}_{\mathrm{max}}}{2}, \frac{3\mathrm{pkl}_{\mathrm{max}}}{4}$ and $\mathrm{pkl}_{\mathrm{max}}$, which are expressed as (32), shown at the bottom of the next page.

If $pkl_{max} = 0.02$, the corresponding diagram can be depicted as Fig. 2:

Likewise, membership functions for fuzzy sets of frontier delay are also defined by 5 Gaussian curves in a condition of $\sigma = 0.1$ and $\mathrm{c} = 0, \frac{\mathrm{D}_{\mathrm{max}}}{4}, \frac{\mathrm{D}_{\mathrm{max}}}{2}, \frac{\mathrm{D}_{\mathrm{max}}}{4}$ and $\mathrm{D}_{\mathrm{max}}$; and expressions of these functions can be written as (33), shown at the bottom of the next page

If $D_{max} = 1ms$, the corresponding diagram is as Fig.3:

Since membership functions of input variables and their fuzzy sets have been defined, membership functions

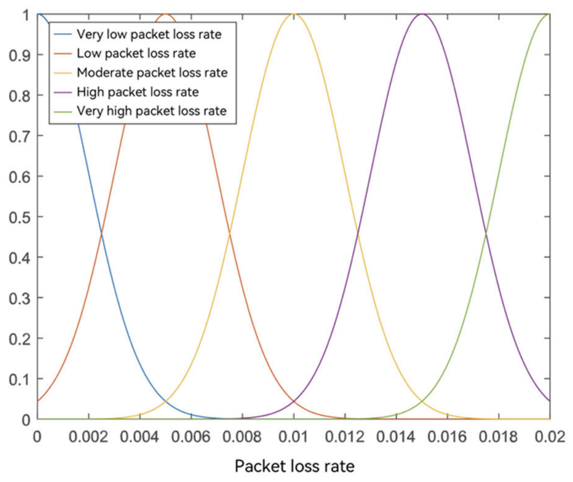


FIGURE 2. A diagram of membership functions based on the packet loss probability.

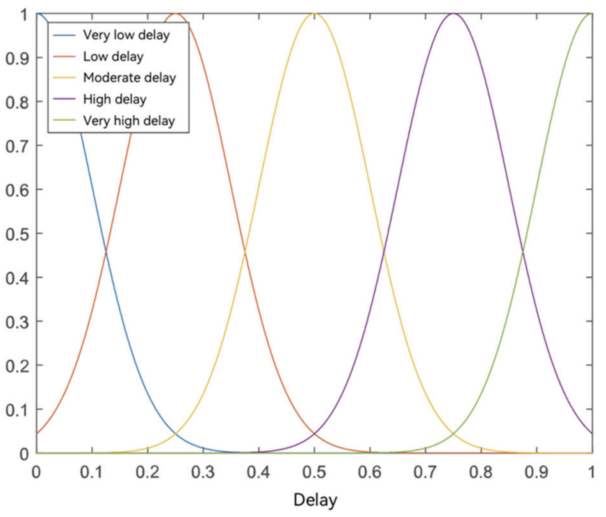


FIGURE 3. A diagram of membership functions based on the delay.

corresponding to fuzzy sets of QoS performance as an input variable need to be defined as well. Considering that the trigonometric function is most widely applied in a fuzzy control system and it performs rather well in stability and robustness, trigonometric membership functions are adopted here, which can be expressed as (34), shown at the bottom of the next page.

Accordingly, the membership function image is as Fig.4:

3) DESIGN OF FUZZY RULES

Fuzzy inference needs to be conducted depending on fuzzy rules in a rule base after packet loss probability and delay of the network frontier are inputted in the fuzzy inference system. For a reason that fuzzy rules have a direct influence on subsequent processes of fuzzy set generation and ambiguity resolution, etc., the rule base is critical to fuzzy inference. From the perspective of QoS assurance, QoS performance of a frontier may be very good according to an arbitrary rule in the context where its packet loss probability and delay are very low. On the basis of this thought, the following rule base is established in this study:

IF packet loss probability is very low and delay is quite low, THEN QoS performance is very good;

$$\mu(pkl)=\begin{cases} e^{-\frac{pkl^2}{0.002^2}}, & \textit{corresponding to a fuzzy set of ``e with quite low packet loss probability''} \\ e^{-\frac{\left(pkl-\frac{pkl_{max}}{4}\right)^2}{0.002^2}}, & \textit{corresponding to a fuzzy set of ``e with very low packet loss probability''} \\ e^{-\frac{\left(pkl-\frac{pkl_{max}}{2}\right)^2}{0.002^2}}, & \textit{corresponding to a fuzzy set of ``e with moderate packet loss probability''} \\ e^{-\frac{\left(pkl-\frac{3pkl_{max}}{4}\right)^2}{0.002^2}}, & \textit{corresponding to a fuzzy set of ``e with very high packet loss probability''} \\ e^{-\frac{(pkl-pkl_{max})^2}{0.002^2}}, & \textit{corresponding to a fuzzy set of ``e with quite high packet loss probability''} \end{cases} \tag{32}$$

$$\mu(D)=\begin{cases} e^{-\frac{D^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with quite low delay''} \\ e^{-\frac{\left(D-\frac{D_{max}}{4}\right)^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with very low delay''} \\ e^{-\frac{\left(D-\frac{D_{max}}{2}\right)^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with moderate delay''} \\ e^{-\frac{\left(D-\frac{3D_{max}}{4}\right)^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with very high delay''} \\ e^{-\frac{(D-D_{max})^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with quite high delay''} \end{cases} \tag{33}$$

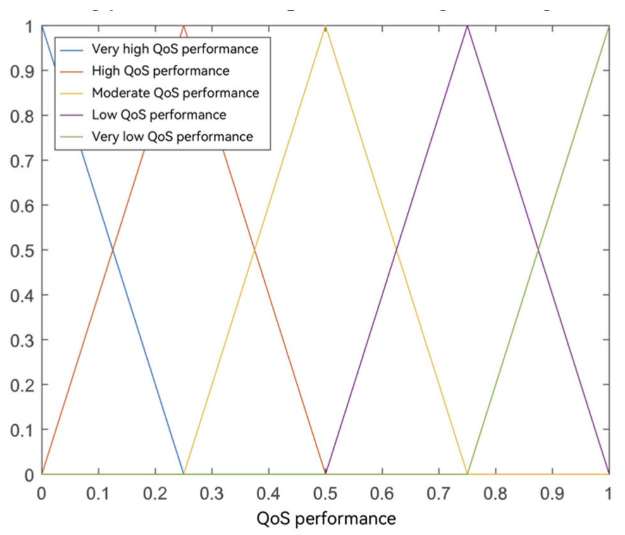


FIGURE 4. A diagram of membership functions based on the QoS performance.

IF packet loss probability is very low and delay is very low, THEN QoS performance is very good;

IF packet loss probability is very low and delay is moderate, THEN QoS performance is moderate;

IF packet loss probability is very low and delay is very high, THEN QoS performance is very poor;

IF packet loss probability is very low and delay is quite high, THEN QoS performance is quite poor;

IF packet loss probability is very low and delay is quite low, THEN QoS performance is very good;

IF packet loss probability is very low and delay is very low, THEN QoS performance is very good;

IF packet loss probability is very low and delay is moderate, THEN QoS performance is moderate;

IF packet loss probability is very low and delay is very high, THEN QoS performance is very poor;

IF packet loss probability is very low and delay is quite high, THEN QoS performance is quite poor;

IF packet loss probability is moderate and delay is quite low, THEN QoS performance is moderate;

IF packet loss probability is moderate and delay is very low, THEN QoS performance is moderate;

IF packet loss probability is moderate and delay is moderate, THEN QoS performance is moderate;

IF packet loss probability is moderate and delay is very high, THEN QoS performance is very poor;

IF packet loss probability is moderate and delay is quite high, THEN QoS performance is quite poor;

IF packet loss probability is very high and delay is quite low, THEN QoS performance is very poor;

IF packet loss probability is very high and delay is very low, THEN QoS performance is very poor;

IF packet loss probability is very high and delay is moderate, THEN QoS performance is very poor;

IF packet loss probability is very high and delay is very high, THEN QoS performance is quite poor;

IF packet loss probability is very high and delay is quite low, THEN QoS performance is quite poor;

IF packet loss probability is quite high and delay is quite low, THEN QoS performance is quite poor;

IF packet loss probability is quite high and delay is very low, THEN QoS performance is quite poor;

IF packet loss probability is quite high and delay is moderate, THEN QoS performance is quite poor;

IF packet loss probability is quite high and delay is very high, THEN QoS performance is quite poor;

IF packet loss probability is quite high and delay is quite high, THEN QoS performance is quite poor.

In this base, all possible fuzzy inference scenarios are described for packet loss probability and delay of network frontier. Without a doubt, this rule base is both scientific and general. The relationship of two preconditions is denoted by "and"; and the rule of operation follows intersection operation of fuzzy sets. For example, with respect to a frontier with very high packet loss probability and very low delay, its QoS performance remains below the moderate level, which is caused by the high packet loss probability despite the fact that its delay satisfies users' QoS requirements. In this case, the result of this operation is defined to be "THEN QoS performance is very poor".

4) AGGREGATION OF FUZZY SETS AND AMBIGUITY RESOLUTION

Based on each fuzzy rule in this base, specific values of inputted variables can be obtained. Then, an appropriate

$$\mu(D)=\begin{cases} e^{-\frac{D^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with quite low delay''}\\ e^{-\frac{\left(D-\frac{D_{max}}{4}\right)^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with very low delay''}\\ e^{-\frac{\left(D-\frac{D_{max}}{2}\right)^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with moderate delay''}\\ e^{-\frac{\left(D-\frac{3D_{max}}{4}\right)^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with very high delay''}\\ e^{-\frac{(D-D_{max})^2}{0.1^2}}, & \textit{corresponding to a fuzzy set of ``e with quite high delay''} \end{cases} \tag{34}$$

implication method is utilized to map these values on a fuzzy set of corresponding output. Here, "truncation" is selected as the implication method to capture membership functions of fuzzy sets below specific values. Afterwards, aggregation is performed for fuzzy sets that are obtained by each rule by means of "max". At last, the aggregated fuzzy set undergoes ambiguity resolution. Commonly used ambiguity resolution approaches include maximum membership, weighted mean and center of gravity (CoG). Due to comparatively low accuracy and poor effects of maximum membership and weighted mean methods, the ambiguity resolution method of COG is adopted here. The corresponding principle is that: CoGs of membership functions for the aggregated fuzzy sets that can enclose an area with the horizontal axis can be used as the output, which is expressed in equation (35) below:

$$v = \frac{\int v\mu_v(v)\,dv}{\int \mu_v(v)\,dv} \tag{35}$$

By means of ambiguity resolution, a specific value f is acquired eventually. It means that QoS performance can be evaluated as f for a frontier with packet loss probability pkl and delay D. In other words, f represents the cost of this frontier. According to costs of respective frontiers, users' QoS demands and the corresponding control overhead constraint, a feasible path with the least cost and satisfying the constraint can be found through computing.

B. A CONTROL OVERHEAD OPTIMIZATION ALGORITHM BASED ON LAGRANGIAN RELAXATION

In the above fuzzy inference system, values assigned to QoS performance of each frontier can be worked out after network topology pruning based on relevant network status information (e.g., network topology, network bandwidth and delay) collected by a controller. Moreover, the cost of respective frontiers is also obtained in line with these values. To find a path with the least possible cost for users means that a path of which QoS performance is as good as possible is acquired, which remains consistent with our original intention of model building.

However, it is much likely that users' QoS demands cannot be satisfied by figuring out the frontier cost based on fuzzy inference and finding a shortest path based on the frontier cost. Relevant reasons can be described as follows. The cost of each frontier is obtained by comprehensively considering both packet loss probability and delay of this frontier; and users also pose specific constraints over packet loss probability and delay. In this context, a constraint among them may not be satisfied. As the objective function has been proved to be NP-Hard, its optimal solution cannot be obtained in polynomial time. Therefore, the principle of Lagrangian Relaxation was selected here to design a control overhead optimization algorithm based on Lagrangian Relaxation. While users' QoS demands can be met, a feasible path of the least possible cost is also found to realize control overhead optimization.

Based on the frontier cost obtained through network topology pruning and fuzzy inference, the objective function of this model can be rewritten as follows:

Objective 4: To find a path $P_{SD} \in \boldsymbol{P}$ satisfying a condition (36) below:

$$min \sum_{e\in P_{SD}} f(e) \tag{36}$$

Constraints are (37) to (39):

$$\sum_{e\in P_{SD}} D_e \le \Delta_{delay} \tag{37}$$

$$\sum_{e\in P_{SD}} pkl_e^* \le \Delta_{pkl} \tag{38}$$

$$\sum_{e\in P_{SD}} SL_e \le \lambda \tag{39}$$

The reason why this objective function is proved to be a NP-Hard problem is that this function contains three constraints Δ_{delay}, Δ_{pkl} and λ with inflexible rigidity. According to Lagrangian Relaxation, constraints with inflexible rigidity are incorporated as Lagrangian multipliers into the objective function by virtue of relaxation. Through iterations, upper and lower bounds and Lagrangian multipliers of the problem to be solved are continuously updated. Finally, a feasible solution that is close to or even equal to the optimal solution is obtained. Additionally, Lagrangian multipliers γ_1, γ_2 and γ_3 are introduced in the objective function, so that constraints $\sum_{e\in P_{SD}} D_e \le \Delta_{delay}$, $\sum_{e\in P_{SD}} pkl_e^* \le \Delta_{pkl}$and $\sum_{e\in P_{SD}} SL_e \le \lambda$ can be relaxed into the objective function. Therefore, the objective function is updated.

Objective 5: To find a path $P_{SD} \in \boldsymbol{P}$ satisfying the following equation (40):

$$\begin{aligned} L(\gamma_1, \gamma_2, \gamma_3) &= min \sum_{e\in P_{SD}} f(e) + \gamma_1 \left(\sum_{e\in P_{SD}} D_e - \Delta_{delay} \right) \\ &\quad + \gamma_2 \left(\sum_{e\in P_{SD}} pkl_e^* - \Delta_{pkl} \right) + \gamma_3 \left(\sum_{e\in P_{SD}} SL_e - \lambda \right) \\ &= min \sum_{e\in P_{SD}} f(e) + \gamma_1 \sum_{e\in P_{SD}} D_e + \gamma_2 \sum_{e\in P_{SD}} pkl_e^* \\ &\quad + \gamma_3 \sum_{e\in P_{SD}} SL_e - \gamma_1\Delta_{delay} - \gamma_2\Delta_{pkl} - \gamma_3\lambda \end{aligned} \tag{40}$$

where, $\gamma_1 \ge 0$, $\gamma_2 \ge 0$ and $\gamma_3 \ge 0$.

The new objective function expressed in equation (40) is referred to as a relaxation problem of the preceding objective function. Regarding Lagrangian multipliers γ_1, γ_2 and γ_3 introduced, they are new independent variables in the new objective function which is a dual function of the previous problem. From another point of view, as long as values of γ_1, γ_2 and γ_3 are determined, the original problem turns into another problem of finding an unconstrained shortest single-weight path. Then, a proper single-source shortest path algorithm such as Dijkstra can be used to solve this path.

How to solve the dual function $L(\gamma_1, \gamma_2, \gamma_3)$ is a problem to consider followingly. This function relates to three variables of γ_1, γ_2 and γ_3. Firstly, we need to prove that $L(\gamma_1, \gamma_2, \gamma_3)$ is a lower bound of the original problem.

Theorem 5: $L(\gamma_1, \gamma_2, \gamma_3)$ is a lower bound of the original problem.

Proof: Targeted at the original problem, it is assumed that it has an optimal solution P^{opt}, so that the dual function turns into (41):

$$\begin{aligned}
&L(\gamma_1, \gamma_2, \gamma_3)\\
&= min \sum_{e\in P^{opt}} f(e) + \gamma_1\left(\sum_{e\in P^{opt}} D_e - \Delta_{delay}\right)\\
&\quad + \gamma_2\left(\sum_{e\in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3(\sum_{e\in P^{opt}} SL_e - \lambda)\\
&= min \sum_{e\in P^{opt}} f(e) + \gamma_1\left(\sum_{e\in P^{opt}} D_e - \Delta_{delay}\right)\\
&\quad + \gamma_2\left(\sum_{e\in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3(\sum_{e\in P^{opt}} SL_e - \lambda) \qquad (41)
\end{aligned}$$

As $P^{opt} \in \boldsymbol{P}$, there exist $\sum_{e\in P^{opt}} D_e \le \Delta_{delay}$, $\sum_{e\in P_{SD}} pkl_e^* - \Delta_{pkl}$and $\sum_{e\in P^{opt}} SL_e \le \lambda$.

As $\gamma_1 \ge 0$, $\gamma_2 \ge 0$ and $\gamma_3 \ge 0$, we have $\gamma_1(\sum_{e\in P^{opt}} D_e - \Delta_{delay}) \le 0$, $\gamma_2\left(\sum_{e\in P_{SD}} pkl_e^* - \Delta_{pkl}\right) \le 0and\gamma_3(\sum_{e\in P^{opt}} SL_e - \lambda) \le 0$.

Therefore,

$$\begin{aligned}
&L(\gamma_1, \gamma_2, \gamma_3)\\
&\quad = min \sum_{e\in P^{opt}} f(e) + \gamma_1(\sum_{e\in P^{opt}} D_e - \Delta_{delay})\\
&\qquad + \gamma_2\left(\sum_{e\in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3(\sum_{e\in P^{opt}} SL_e - \lambda)\\
&\quad \le min \sum_{e\in P^{opt}} f(e)\\
&\quad = f\left(P^{opt}\right) \qquad (42)
\end{aligned}$$

$L(\gamma_1, \gamma_2, \gamma_3)$ is a lower bound of the original problem. End of proof.

The above proof indicates that dual function $L(\gamma_1, \gamma_2, \gamma_3)$ is a lower bound of the original problem, which points out the direction of dual function optimization. Because $L(\gamma_1, \gamma_2, \gamma_3)$ is a lower bound of the original problem, $L(\gamma_1, \gamma_2, \gamma_3)$ should approximate the optimal solution of the original problem to the greatest extent in order to maximize it. In this case, a dual problem is generated as shown in (43) below:

$$\boldsymbol{L} = \max L(\gamma_1, \gamma_2, \gamma_3) \qquad (43)$$

Dual problem of the original problem is to maximize the dual function $L(\gamma_1, \gamma_2, \gamma_3)$, enabling the dual function to approximate the optimal solution of the original problem as close as possible. As a result, the original problem can be solved with the help of γ_1, γ_2 and γ_3. In the course of solving this dual problem, an upper bound of the original problem needs to be determined as well. To approach the optimal solution of the original problem as close as possible, a difference between upper and lower bounds of the original problem should be constantly reduced. Subsequently, a method is raised to confirm its upper bound.

Theorem 6: For given γ_1, γ_2 and γ_3, cost $f(P)$ obtained by solving the relaxation problem for path P is the upper bound of the original problem.

Proof: As γ_1, γ_2 and γ_3 are known, path P obtained by solving the relaxation problem has two situations:

① P is the optimal solution P^{opt} of the original problem, that is $P = P^{opt}$, so there exists $f(P) = f\left(P^{opt}\right)$;

② P is a feasible solution of the original problem, but not its optimal solution P^{opt}. Considering that P is not an optimal solution of the original problem, we have $f(P) \ge f\left(P^{opt}\right)$.

To sum up, $f(P) \ge f\left(P^{opt}\right)$ is satisfied. Therefore, cost $f(P)$ of path P obtained by solving the relaxation problem serves as its upper bound for given γ_1, γ_2 and γ_3. End of proof.

According to Theorem 1, Theorem 2 and relevant proof, it can be known that the difference between upper and lower bounds becomes small enough by constantly solving the dual problem, increasing lower bound of the original problem but decreasing its upper bound. In this way, a feasible solution close to the optimal solution is acquired. There are many methods to solve the dual problem. Among them, the subgradient method performs better. By continuously updating Lagrangian multipliers and increasing the solution to a dual problem, this method makes the dual problem unceasingly approximate the optimal solution of the original problem by virtue of directions and step sizes of subgradient. In specific, this process can be expressed in equation (44):

$$\gamma^{n+1} = \max\{0, \gamma^n + s_n g_n\} \qquad (44)$$

where, elements γ^k and $k \in \{1, 2, \ldots, n\}$ in sequence $\left\{\gamma^1, \gamma^2, \ldots, \gamma^n\right\}$ are Lagrangian multipliers obtained after the *kth* iteration. Therefore, to apply the subgradient method, the initial Lagrangian multiplier γ^0 should be set in the first place. s_n and g_n refer to subgradient and step size of n^{th} iteration, respectively. Moreover, subgradient and step size can be respectively interpreted as direction and distance of optimization. On the premise of a correct subgradient and step size are selected, the difference between upper and lower bounds can be increasingly lowered for the original problem by means of constant iterations, so that convergency of results is realized. At last, a feasible solution that produces a preferable optimization effect is obtained.

Firstly, subgradient is defined as follows:

Definition 6: For a real-valued function $f(x)$, if a vector $g \in R^n$ is the subgradient of $f(x)$ at the point $x_0 \in R^n$, the vector g satisfies the following equation (45) in a condition of $\forall x \in R^n$:

$$f(x) - f(x_0) \leq g(x - x_0) \tag{45}$$

According to Definition 6, a subgradient is established for the dual problem. Here, γ_1^k, γ_2^k and γ_3^k respectively represent Lagrangian multipliers γ_1, γ_2 and γ_3 obtained after k^{th} iteration. It is assumed that path P_k is the optimal solution to the dual function $L\left(\gamma_1^k, \gamma_2^k, \gamma_3^k\right)$ after the k^{th} iteration, a subgradient of γ_1^k is written into the following (46):

$$g_k^1 = \sum_{e \in P_k} D_e - \Delta_{delay} \tag{46}$$

The subgradient of γ_2 is shown in (47):

$$g_k^2 = \sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl} \tag{47}$$

The subgradient of γ_3 is shown in (48):

$$g_k^3 = \sum_{e \in P_k} SL_e - \lambda \tag{48}$$

Subsequently, we need to prove that g_k^1, g_k^2 and g_k^3 are subgradients of γ_1, γ_2 and γ_3 after the k^{th} iteration.

Theorem 7: g_k^1 is the subgradient of γ_1 after the kth iteration; g_k^2 is the subgradient of γ_2 after the kth iteration; and g_k^3 is the subgradient of γ_3 after the kth iteration.

Proof As γ_1, γ_2 and γ_3 are unrelated dependent variables of the dual function $L(\gamma_1, \gamma_2, \gamma_3)$, we only need to prove that g_k^1 is the subgradient of γ_1 after the kth iteration; similarly, g_k^2 and g_k^3 are subgradients of γ_2 and γ_3 after the k^{th} iteration. This is proved by the following (49) to (52):

Because

$$g_k^1 = \sum_{e \in P_k} D_e - \Delta_{delay} \tag{49}$$

A Lagrangian multiplier with respect to the delay constraint after the k^{th} iteration is assumed to be γ_1^k; then, for $\forall x \in R$, there exists

$$\begin{aligned}
&g_k^1\left(x - \gamma_1^k\right) \\
&= \left(\sum_{e \in P_k} D_e - \Delta_{delay}\right)\left(x - \gamma_1^k\right) \\
&= \left(\sum_{e \in P_k} D_e - \Delta_{delay}\right) x - \left(\sum_{e \in P_k} D_e - \Delta_{delay}\right) \gamma_1^k \\
&= \sum_{e \in P_k} f(e) + x\left(\sum_{e \in P_k} D_e - \Delta_{delay}\right) \\
&\quad + \gamma_2\left(\sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3\left(\sum_{e \in P_k} SL_e - \lambda\right) \\
&\quad - \left[\sum_{e \in P_k} f(e) + \gamma_1^k\left(\sum_{e \in P_k} D_e - \Delta_{delay}\right)\right. \\
&\quad \left. + \gamma_2\left(\sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3\left(\sum_{e \in P_k} SL_e - \lambda\right)\right]
\end{aligned} \tag{50}$$

To solve a Lagrangian dual problem L(x, γ_2, γ_3), the obtained path is assumed as P_x; then, the Lagrangian function corresponding to P_x is the minimal. That is, for $\forall P \in \boldsymbol{P}$, there exists

$$\begin{aligned}
&\sum_{e \in P_x} f(e) + x\left(\sum_{e \in P_x} D_e - \Delta_{delay}\right) \\
&\quad + \gamma_2\left(\sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3\left(\sum_{e \in P_x} SL_e - \lambda\right) \\
&\leq \sum_{e \in P} f(e) + x\left(\sum_{e \in P} D_e - \Delta_{delay}\right) \\
&\quad + \gamma_2\left(\sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3\left(\sum_{e \in P} SL_e - \lambda\right)
\end{aligned}$$

That is,

$$\begin{aligned}
&L(x, \gamma_2, \gamma_3) \\
&\leq \sum_{e \in P} f(e) + x\left(\sum_{e \in P} D_e - \Delta_{delay}\right) \\
&\quad + \gamma_2\left(\sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3\left(\sum_{e \in P} SL_e - \lambda\right)
\end{aligned} \tag{51}$$

Thus,

$$\begin{aligned}
&g_k^1\left(x - \gamma_1^k\right) \\
&= \sum_{e \in P_k} f(e) + x\left(\sum_{e \in P_k} D_e - \Delta_{delay}\right) \\
&\quad + \gamma_2\left(\sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3\left(\sum_{e \in P_k} SL_e - \lambda\right) \\
&\quad - [\sum_{e \in P_k} f(e) + \gamma_1^k\left(\sum_{e \in P_k} D_e - \Delta_{delay}\right) \\
&\quad + \gamma_2\left(\sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}\right) + \gamma_3\left(\sum_{e \in P_k} SL_e - \lambda\right)] \\
&\geq L(x, \gamma_2) - L\left(\gamma_1^k, \gamma_2\right)
\end{aligned} \tag{52}$$

In this way, g_k^1 is proved to be the subgradient of γ_1 after the k^{th} iteration. Similarly, g_k^2 and g_k^3 are subgradients of γ_2 and γ_3 after the k^{th} iteration. End of proof.

In line with the above proof, it is clear that $g_k^1 = \sum_{e \in P_k} D_e - \Delta_{delay}$, $g_k^2 = \sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}$ and $g_k^3 = \sum_{e \in P_k} SL_e - \lambda$ are all subgradients of Lagrangian multipliers γ_1, γ_2 and γ_3 after the kth iteration in the process of solving the dual problem L. After confirmation of subgradients, optimization direction needs to be determined when the dual problem is solved. Subsequently, step size s_k corresponding to the kth iteration should be also identified. A classical subgradient step size computing formula is selected as (53), shown at the bottom of the page, where, UB_k and LB_k stand for upper and lower bounds obtained after the kth iteration, respectively. Their values are figured out based on Theorem 1 and Theorem 2 in each round of iteratively updating Lagrangian multipliers. By virtue of (53) above, specific values of step size s_k are continuously updated based on values of upper bound UB_k and lower bound LB_k. In this way, the process of approximating the optimal solution becomes more scientific, more reasonable and more efficient.

As subgradients and step sizes are determined, we need to clarify conditions in which iterations of the algorithm are ended. Generally, the following iteration ending situations occur in the course of Lagrangian relaxation:

1) Reaching the preset maximum iteration number *iter*. As infinite iterations are impossible, the maximum iteration number *iter* is designed. When the iteration number reaches *iter*, the algorithm ends. Although this method is simple and direct, it is difficult to ensure quality of the acquired feasible solution.
2) An upper bound is equal to a lower bound, that is $UB_k = LB_k$. In this case, upper bound of the original problem coincides with its lower bound, signifying that the feasible solution obtained is the optimal solution of the original problem. Then, the iteration stops. This is an ideal result of optimization. Considering that the original problem is NP-Hard, such a case can be extremely rare as the problem size keeps enlarging.
3) A sufficiently small difference of upper and lower bounds, that is $UB_k - LB_k \le \varepsilon$, where $\varepsilon \ge 0$ and ε is small enough. In this scenario, a difference between upper and lower bounds is below the preset ε. Therefore, it is deemed that the difference of upper and lower bounds is low enough, so that the iteration stops.

Concerning this algorithm, the above three situations are used as conditions of iteration termination. Firstly, the maximum iteration number *iter* and a small enough positive number ε are inputted before the algorithm starts, ensuring that the algorithm stops when the maximum iteration number is satisfied or the difference of upper and lower bounds is small enough. In addition, since upper and lower bounds have been figured out in each iteration process, we need to estimate their difference. If they are equal, or their difference is no more than ε, the algorithm stops.

To sum up, the Lagrangian relaxation algorithm proposed for satisfying users' QoS demands and implementing control overhead optimization can be represented by the following TABLE 3:

TABLE 3. Pseudocodes of the Lagrangian relaxation algorithm.

Algorithm 4.1 An algorithm framework of Lagrangian Relaxation based on the fuzzy logic inference cost

Input: The number of network nodes (N), cost matrix $\boldsymbol{f}$, bandwidth overhead $\boldsymbol{U}$, delay D, packet loss probability $\boldsymbol{PKL}$, control overhead $\boldsymbol{SL}$, source node n_S and destination node n_D, time delay constraint Δ_{delay}, packet loss probability constraint Δ_{pkl}, control overhead constraint λ, iteration number $iter$, and $\varepsilon \ge 0, \theta \ge 0$.
Output: A feasible solution path p

```
01  Initialize $\gamma_1 = 2, \gamma_2 = 2,\ \gamma_3 = 2, k = 1, UB_0 = +\infty,\ \varepsilon \ge 0$
02              $p_{opt1} = Dijkstra(n_S, n_D, f)$
03  if ($D_{p_{opt1}} \le \Delta_{delay}$ &&$pkl_{p_{opt1}} \le \Delta_{pkl}$&& $SL_{p_{opt1}} \le \lambda$)
04      return $p_{opt1}$
05  else
06      $p_D = Dijkstra(n_S, n_D, \boldsymbol{D})$
07      if ($D_{p_D} \ge \Delta_{delay}$ )
08          return fail
09      else
10          While($k \le iter$)
11          $p_k = Dijkstra(n_S, n_D, \boldsymbol{f} + \gamma_1 \boldsymbol{D} + \gamma_2 \boldsymbol{PKL} + \gamma_3 \boldsymbol{SL})$
12              if ($d_{p_{opt1}} \le \Delta_{delay}$ && $pkl_{p_{opt1}} \le \Delta_{pkl}$&&$SL_{p_{opt1}} \le \lambda$ && $f(p_k) \le UB_{k-1}$)
13                  $UB_k = f(p_k)$, count=0
14              else
15                  $UB_k = UB_{k-1}$
16              $LB_k = f(p_k) + \gamma_1(D_{p_k} - \Delta_{delay}) + \gamma_2\left(pkl_{p_{opt1}} - \Delta_{pkl}\right) + \gamma_3(SL_{p_k} - \lambda)$
17              if ($UB_k - LB_k \le \varepsilon$)
18                  return $p_k$
19              else
20  $s_k = \frac{UB_k - LB_k}{(D_{p_k} - \Delta_{delay})^2 + (\sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl})^2 + (SL_{p_k} - \lambda)^2}$
21              $\gamma_1 = \max\{0, \gamma_1 \times s_k \times (D_{p_k} - \Delta_{delay})$
22              $\gamma_2 = \max\{0, \gamma_2 \times s_k \times (pkl_{p_{opt1}} - \Delta_{pkl})$
23                  $\gamma_3 = \max\{0, \gamma_3 \times s_k \times (SL_{p_k} - \lambda)$
24              $k++$
```

As can be seen, input of this algorithm covers a cost matrix $\boldsymbol{f}$ of the network, and its bandwidth overhead matrix $\boldsymbol{U}$, delay $\boldsymbol{D}$, packet loss probability $\boldsymbol{PKL}$, control overhead matrix $\boldsymbol{SL}$, source node n_S, destination node n_D, time delay constraint Δ_{delay}, packet loss probability constraint Δ_{pkl}, control overhead constraint λ, the maximum iteration number *iter*, a small enough positive number ε, and a step size parameter β.

$$s_k = \frac{UB_k - LB_k}{\left(\sum_{e \in P_k} D_e - \Delta_{delay}\right)^2 + \left(\sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}\right)^2 + \left(\sum_{e \in P_k} SL_e - \lambda\right)^2} \tag{53}$$

In Line 01, Lagrangian multipliers γ_1, γ_2 and γ_3 are initialized into $\gamma_1 = 2$, $\gamma_2 = 2$ and $\gamma_3 = 2$, $k = 1$ initialized into a recorder of the iteration number, and the upper bound UB_0 prior to iteration initialized into a positive number high enough. Moreover, a small enough positive number ε is also initialized to estimate whether the difference of upper and lower bounds is as low as that expected. In Line 02, Dijkstra algorithm is utilized to figure out a path with the least cost from n_S to n_D. From Lines 03 to 04, the path is evaluated, determining whether it meets users' demands for Δ_{delay}, Δ_{pkl} and λ. If this path has already satisfied all constraints, it is the optimal solution and thus directly outputted. Otherwise, as described in Line 06, the Dijkstra algorithm is used again to figure out a path with the least delay from n_S to n_D. If the path still fails in satisfying users' demands for Δ_{delay}, it is clear that no paths can satisfy users' delay requirements in this network. Consequently, the path computing is terminated. If the path with the least delay is found to remain no more than that Δ_{delay} requested by users, it is likely that a feasible solution exists and it satisfies users' QoS demands. Subsequently, we proceed to an iteration procedure of the Lagrangian Relaxation algorithm. In each round of iteration, starting with Line 11, delay and control overhead of each frontier are firstly aggregated to its cost based on current Lagrangian multipliers γ_1, γ_2 and γ_3. Specifically, it is denoted as $f^k(e) = (e) + \gamma_1 D_e + \gamma_2 pkl_e^* + \gamma_3 SL_e$, where k refers to the iteration number. For a reason that each frontier merely has one weight after relaxation, that is $f^k(e)$, we can adopt the Dijkstra algorithm to work out a path with the lowest $f^k(e)$ from n_S to n_D. In Lines 12-13, the path worked out is evaluated to make sure whether it satisfies users' demands for delay and control overhead constraints, and whether the cost corresponding to it is below the upper bound UB_{k-1} of the previous one iteration. If the above conditions are all satisfied, the path is considered as a feasible solution; and its cost may serve as an upper bound obtained during the current iteration. Therefore, a value assigned to the cost of this path is updated and used as the upper bound UB_k during the present iteration. In Line 17, a lower bound LB_k obtained in the current iteration is calculated. Through theoretical demonstration, it is known that the aggregated cost $f(p_k) + \gamma_1 \left(D_{p_k} - \Delta_{delay}\right) + \gamma_2 \left(pkl_{p_{opt1}} - \Delta_{pkl}\right) + \gamma_3 (SL_{p_k} - \lambda)$ corresponding to this path serves as the lower bound LB_k. Thus, both UB_k and LB_k are figured out for the current iteration. In Lines 18-19, the difference of UB_k and LB_k is calculated. If their difference is smaller than ε, it is believed that the feasible solution is extremely close to the optimal solution. In this case, the corresponding path is outputted as a feasible path that satisfies users' QoS demands and realizes control overhead optimization. In Line 23, the step size is calculated, which corresponds to (55). In Lines 24 and 25, Lagrangian multipliers γ_1, γ_2 and γ_3 are updated by virtue of the step size and subgradients (i.e., $g_k^1 = \sum_{e \in P_k} D_e - \Delta_{delay}$, $g_k^2 = \sum_{e \in P_{SD}} pkl_e^* - \Delta_{pkl}$ and $g_k^3 = \sum_{e \in P_k} SL_e - \lambda$). At this point, this round of iteration ends. As described above, this is the whole process of the proposed FuzLag algorithm based on fuzzy inference and Lagrangian Relaxation.

In a word, the proposed FuzLag algorithm is capable of eliminating frontiers that fail to meet relevant conditions by means of network topology pruning, thus boosting computing efficiency. Besides, it takes packet loss probability and delay of each frontier into comprehensive consideration with the help of a fuzzy inference system, thus acquiring specific values of the corresponding cost. Based on thoughts of Lagrangian Relaxation, a feasible path performing rather well in satisfying users' demands for delay, packet loss probability and control overhead optimization is also found by the proposed algorithm on the premise. As a result, the goal of building such a model is achieved.

V. EXPERIMETNAL RESULTS AND ANALYSES

A. EXPERIMENTAL CONFIGURATIONS

Simulation experiments are conducted in Windows 10 as those described below specific to the proposed FuzLag algorithm based on fuzzy logic and Lagrangian Relaxation. Windows 10 system is provided with a memory of 16 GB, and a CPU of Intel i7-9750H. MATLAB R2018b, programming software, was utilized to complete the simulated experiments. Moreover, the platform and software used in the experiments are detailed in the following Table 4:

TABLE 4. Platform and software.

Hardware/software	Description
System	Windows10
Memory	16GB
CPU	Intel i7-9750H
Programming software	MATLAB R2018b

Some real network topologies were selected to carry out the simulated experiments. Internet Topology Zoo consists of some real network topologies in different places of the world [11]. Here, AttMpls, SwitchL3 and GtsCe topologies were chosen from the Topology Zoo, covering cross-sea and inland topologies featuring high representativeness and generalizability. In terms of the corresponding network topology scale, it is listed in the following Table 5:

TABLE 5. Network topology scale.

Hardware/software		Description
AttMpls	25	57
SwitchL3	42	63
GtsCe	149	193

As indicated by node and frontier counts of the above three network topologies, the network topology scale involved in the simulation experiments shows a tendency of gradual increases from being small to being medium and further to being large. Different algorithms were operated in such three network topologies, which may contribute to synthesizing different performance properties and differences of diverse algorithms subjected to various network scales. In this way, experimental results can be much more credible, scientific and reasonable. Based on the law and specific situations of practical network flow transmission in each network topology, for a link with bandwidth overhead exceeding 50%, network transmission congestion takes place in this link. Considering this, bandwidth overhead conforming to (0,0.5) uniform distribution is randomly generated for each link in the present study. Similarly, the delay randomly generated for respective links complies with (0,1ms) uniform distribution; and for the packet loss probability randomly generated for all links separately, it is in consistency with (0,0.5%) uniform distribution. Thanks to such configurations, delay and bandwidth overhead of network frontiers are featured with certain randomness on one hand, and also preferably coincide with network transmission situations in practice. Hence, experimental results are both scientific and reliable.

Two algorithms were utilized during each experiment for comparison with the proposed FuzLag algorithm, clarifying their advantages and disadvantages in diverse measurement standards. Moreover, such two algorithms are described as follows:

1) OSPF [12]. OSPF and OSPF protocols are most extensively applied interior gateway protocols (IGPs). To ensure consistency with the actual network situations, a path obtained by OSPF is assumed to be the shortest path with the least hop counts between a given source node and a given destination node; and this path also acts as the default path $P_{default}$ of the source and the destination nodes.
2) QRS [13]. This is a QoS optimization algorithm raised in 2019. After QoS parameter aggregation for network frontiers, the Dijkstra algorithm is adopted to figure out the shortest path, which is known as the QoS optimization scheme of QRS. The corresponding formula is given (54) below:

$$\mathrm{C}(i,j) = w_1 \times pkl + w_2 \times U + w_3 \times SL \tag{54}$$

where, constants w_1, w_2 and $w3$ are designed to control proportions occupied by different items in the link computing formula, where $w_1, w_2$and$w_3 \in [0, 1]$. After network flow transmission related information and QoS demands have been collected, the SDN controller is able to work out weights of respective links based on the current network topology and network status information captured. Eventually, the shortest path is obtained by means of the Dijkstra algorithm; and the shortest path serves as a path of QoS optimization.

During experiments, some parameters should be set up and used to evaluate performance of different algorithms. As the present chapter focuses on a QoS optimization routing problem under constraints of QoS and control overhead, the following three different parameters are configured to assess algorithm performance and results.

1) Path computing success rate ξ. According to the above proof, a NP-Hard problem is investigated in this chapter. Not only are network status information variations rather complicated, but users' QoS demands may be rigorous or relaxed. Under such circumstances, a QoS optimization path that satisfies all constraints may not be always figured out when the algorithm is executed. Therefore, the success rate ξ is designed to measure computing capability of the QoS optimization algorithm, which can be expressed in equation (55) below:

$$\xi = \frac{S}{S+L} \tag{55}$$

In different network topologies, some network flow sets with particular QoS optimization demands may be randomly generated. In the above equation, S is the number of paths that are worked out by a given algorithm and conform to QoS transmission requirements. L refers to the number of paths that are worked out by a given algorithm and fail in conforming to corresponding QoS transmission requirements. If ξ is rather high, it is much likely for the path found by the given algorithm to meet relevant QoS optimization demands; but if ξ is low, probability for the obtained path to meet QoS optimization demands can be also low.

1) Delay of the obtained paths satisfying QoS constraints. Likewise, a path with the best possible QoS performance is found in this chapter for users on the premise of satisfying QoS constraints of network flow. Therefore, delay is also an index measuring QoS performance of links and paths. Here, such delay is worked out by a given algorithm to evaluate the algorithm performance. Regarding given network topologies and algorithms, algorithm performance is assessed by calculating avg_D, that is the average delay of all paths that satisfy QoS constraints and are obtained provided that all algorithms produce results. It is expressed in the following (56):

$$avg_D = \frac{\sum_{D\in S} D}{S} \tag{56}$$

where, S stands for the number of paths that can be worked out by means of all algorithms and also satisfy QoS requirements for transmission, and $\sum_{D\in S} D$ for the total sum of delay values corresponding to a path worked out by a specific algorithm.

1) Packet loss probability of paths found to satisfy QoS constraints. The present chapter is intended to find a path with the best possible QoS performance on the premise of satisfying QoS constraints for network flows. Considering this, packet loss probability is also an index that can be utilized to measure QoS capability of links and paths here. Therefore, the packet loss probability worked out by a particular algorithm is selected to measure performance of this algorithm. With respect to given network topologies and algorithms, algorithm

performance is evaluated by calculating avg_{pkl}, that is the average packet loss probability of all paths that satisfy QoS constraints and are obtained provided that all algorithms produce results. It is expressed in the following (57):

$$avg_{pkl} = \frac{\sum_{pkl \in S} pkl}{S} \tag{57}$$

In the above equation (57), S represents the number of paths that can be worked out by all algorithms and satisfy QoS requirements for transmission; and $\sum_{pkl \in S} pkl$ is the total sum of packet loss probabilities corresponding to a path obtained by a particular algorithm.

During the experiments, 100-500 transmission requests from source to destination nodes are randomly generated in accordance with the network topology. Thanks to such a large sample size, algorithm comparison results can be quite scientific and credible.

B. PATH COMPUING SUCCESS RATE ξ

When delay and packet loss probability constraints are raised gradually in small-sized, medium-sized, and large-sized network topologies (i.e., AttMpls, SwitchL3, and GtsCe), path computing success rates of three different algorithms accordingly change in a condition of given QoS requirements for transmission, as depicted in Fig. 5 to Fig. 10. It can be observed from Figures 6, 8 and 10 that constraints over path delay are gradually relaxed along with gradual increases in delay constraints. This signifies that the path computing success rates of such three algorithms progressively go up in succession. Similarly, as shown in Fig. 5, Fig. 7, and Fig. 9 where the packet loss probability constraints are raised progressively, constraints over the packet loss probability of a path are gradually relaxed. Considering this, the success rate of finding a path by such three algorithms is progressively elevated in succession. Under the circumstance of quite strict delay or packet loss probability constraints, the success rates of such three algorithms are all very low in early phases for the following reasons. When lots of network frontiers fail to meet constraints, a vast majority of paths selected on the premise of successful pathfinding can coincide with each other, so that their success rates are the same. If the constraints become gradually relaxed, the success rate of the FuzLag algorithm can be substantially improved by 3∼20% in comparison with another two algorithms. In terms of QRS that takes QoS related indexes of network frontiers into comprehensive consideration, its success rates are raised by 2∼10% if compared with OSPF in a condition of rather relaxed constraints. Through such comparisons, it is much more likely for the proposed FuzLag algorithm to find a path for users on the premise of satisfying their particular QoS demands and optimizing the control overhead. In other words, the path computing success rates of the FuzLag algorithm are higher than those of another two algorithms.

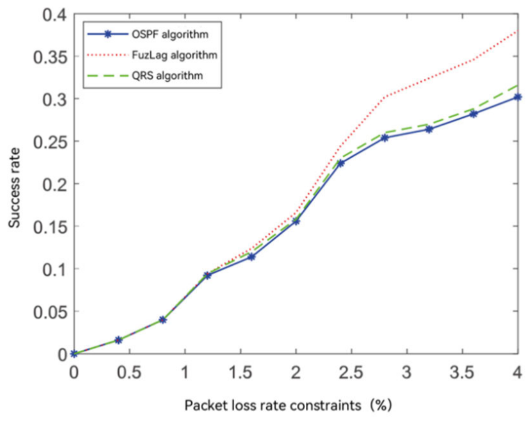


FIGURE 5. Success rates under packet loss probability variations in a small-sized network.

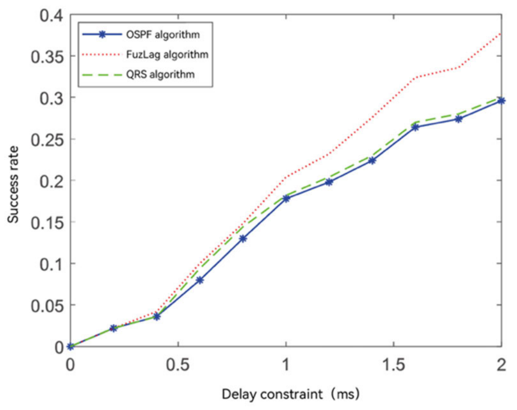


FIGURE 6. Success rates under delay variations in a small-sized network.

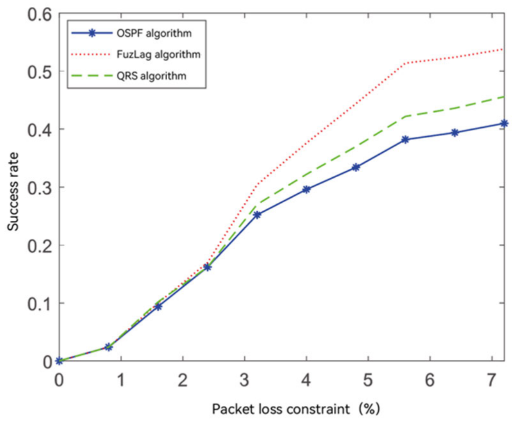


FIGURE 7. Success rates under packet loss probability variations in a medium-sized network.

C. DELAY

In AttMpls, SwitchL3, and GtsCe, values of the average delay are obtained for a path set that is worked out by three different algorithms when the corresponding packet loss probability

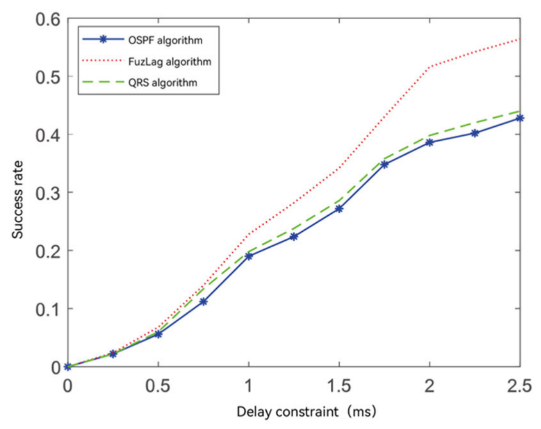


FIGURE 8. Success rates under delay variations in a medium-sized network.

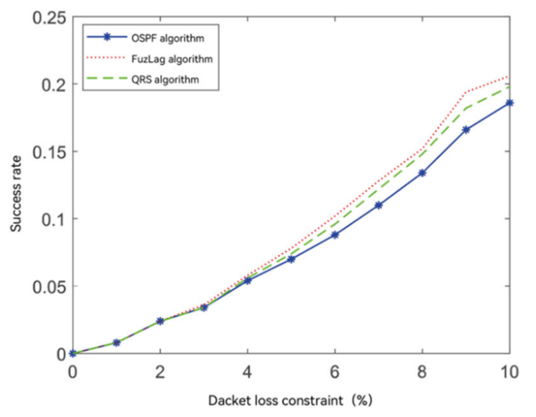


FIGURE 9. Success rates under packet loss probability variations in a large-sized network.

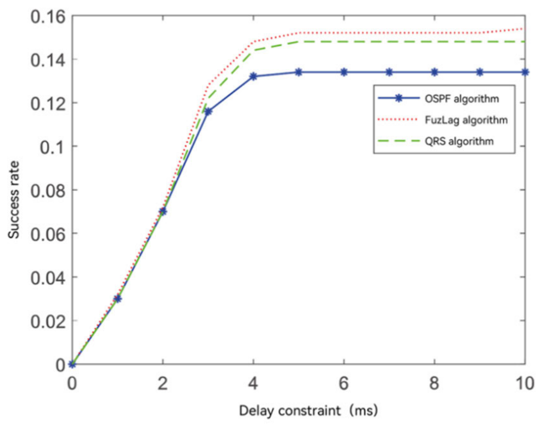


FIGURE 10. Success rates under delay variations in a large-sized network.

remains unchanged and the delay constraint changes from being rigorous to being relaxed gradually, as revealed in Fig. 11 to Fig. 13. Clearly, the average delay of paths obtained by three algorithms shows a rising tendency as the delay

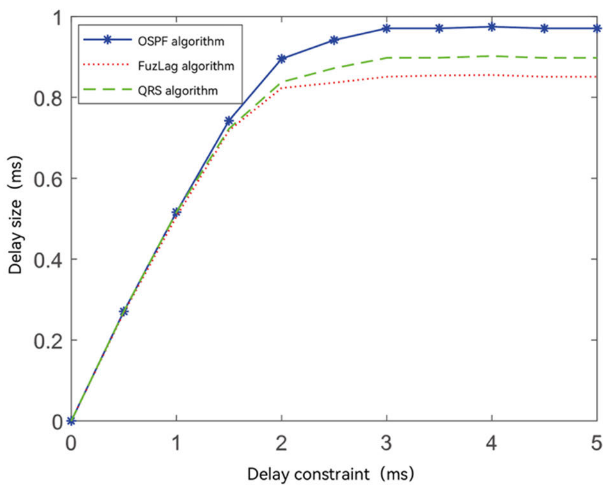


FIGURE 11. Delay of a small-sized network.

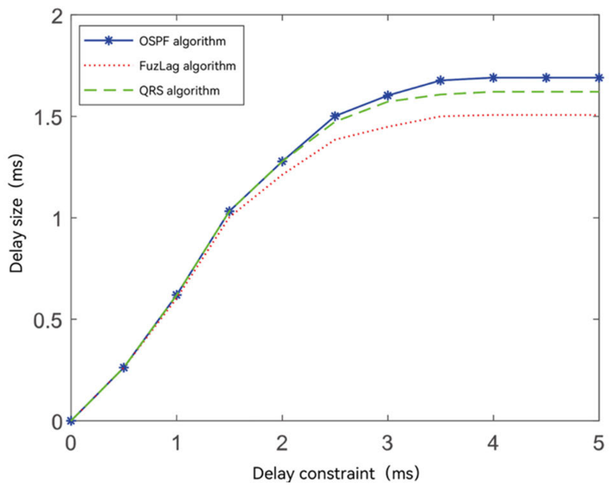


FIGURE 12. Delay of a medium-sized network.

constraint becomes relaxed. The corresponding reasons are that: 1) a gradual increase in delay constraints leads to an increasing number of paths satisfying delay requirements; 2) the average delay also goes up on the whole due to constraint relaxation. In early phases when the delay constraint is rather rigorous, paths obtained by three algorithms are consistent, making it difficult to present their differences in performance. As it becomes relaxed little by little, the proposed FuzLag algorithm gradually outperforms another two algorithms in their average delay. Due to its stronger adaptability, it can find the optimal path in a shorter period. To be specific, the delay arising from FuzLag is about 0.3∼11% lower than that of OSPF; and when comparing with the delay performance of QRS, FuzLag generates a delay 0.2∼8% smaller approximately. Therefore, FuzLag is superior to OSPF and QRS as far as their performance is concerned in a case where their delay constraints are comparatively relaxed.

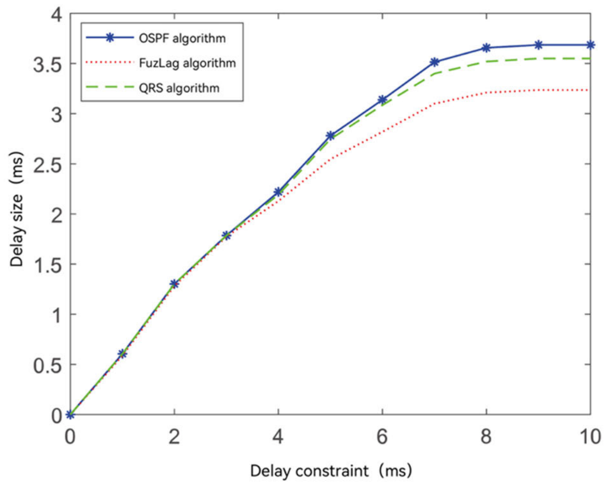


FIGURE 13. Delay of a large-sized network.

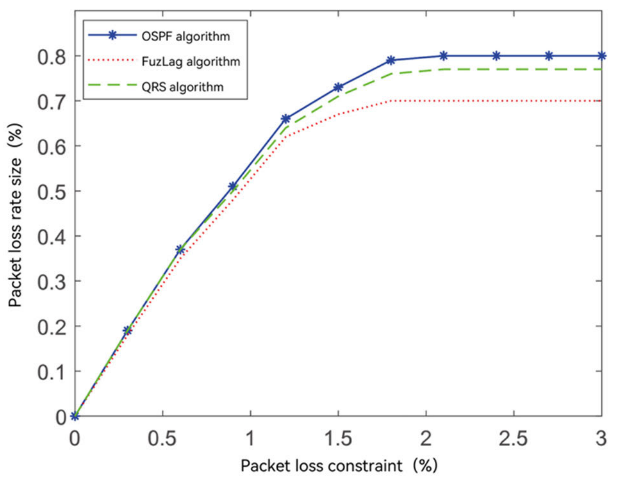


FIGURE 15. Packet loss probability of a medium-sized network.

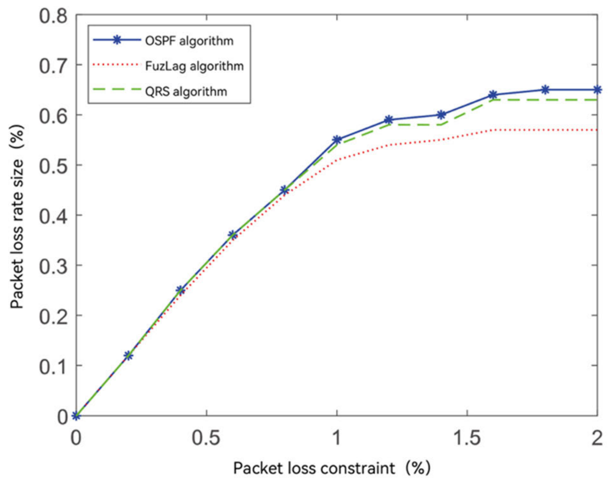


FIGURE 14. Packet loss probability of a small-sized network.

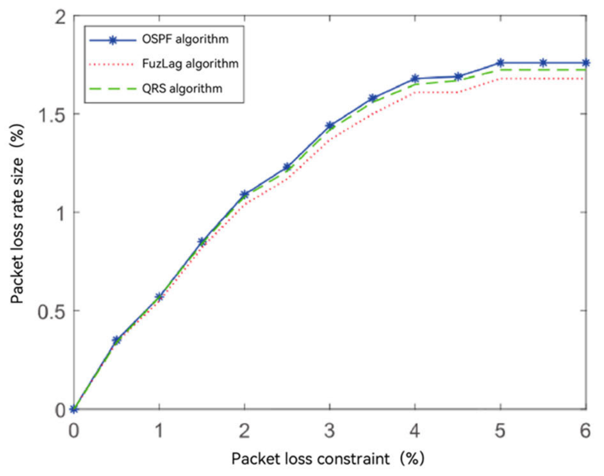


FIGURE 16. Packet loss probability of a large-sized network.

D. LOSS PACKET PROBABILITY

In AttMpls, SwitchL3, and GtsCe, likewise, when their delay constraints are fixed and packet loss probability constraints become relaxed, the average values of packet loss probability in a set of paths worked out by such three algorithms are plotted in Fig. 14 to Fig. 16. In the process where the packet loss probability constraints are relaxed progressively, the average packet loss probability of paths obtained by OSPF, QRS, and FuzLag is also raised for the following reasons. First, a gradual increase in the packet loss probability makes the number of paths satisfying users' QoS demands for delay increase; and second, constraint relaxation enables the average packet loss probability of paths to go up. In the early phases of strict constraints, average packet loss probabilities of paths worked out by such three algorithms are quite close to each other. With constraint relaxation, the average packet loss probability of paths obtained through FuzLag turns out to be smaller than those generated by OSPF and QRS. More particularly, the performance of FuzLag is improved by 0.2∼9% if compared with OSPF as far as their packet loss probabilities are concerned; and by contrast to QRS, the performance of the former is boosted by 0.1∼6%. What is worth mentioning is that GtsCe has a rather strict requirement for the computing capability of such three algorithms due to its large size. Thus, the average packet loss probability of OSPF is almost the same as that of QRS; and their performance difference remains no more than 0.1%. FuzLag shows no quite obvious performance advantages in this aspect. In comparison with OSPF and QRS, its performance is improved by about 1%.

VI. CONCLUSION AND PROSPECTS

To achieve the purpose of QoS assurance for transmission in a smart ocean network, the present study considers control overhead incurred by SR, and selects methods of link QoS index fuzzification and Lagrangian Relaxation. It is expected to work out a path that satisfies users' QoS requirements and provides the best possible QoS, and improves user experience in network transmission. By simulation experiments, we verify the performance of the proposed algorithm. The concrete contributions of this study are as follows:

To realize QoS assurance in a smart ocean network, a model is built based on users' QoS demands, which is proved to be a NP-Hard problem. Specific to this objective function, a FuzLag algorithm is put forward based on fuzzy inference and Lagrangian Relaxation. QoS related performance of links is fuzzified, so that weights of the corresponding path can be obtained. In addition, QoS constraints are relaxed into the objective function, figuring out a path that satisfies users' QoS demands and provides the best possible QoS at last.

Moreover, it turns out in this study that the proposed FuzLag algorithm based on fuzzy inference and Lagrangian Relaxation not only generates a very low path computing success rate in a case where constraints are comparatively rigorous, but also lowers the corresponding delay and packet loss probability to a small extent. Considering this, we will focus on how to further reduce delay and packet loss probability in future research. Besides, because the smart ocean network architecture is not implemented in a practical ocean network system, concrete implementation of its functions will be conducted in a real ocean network system.

REFERENCES

[1] X. Xiao and L. M. Ni, "Internet QoS: A big picture," *IEEE Netw.*, vol. 13, no. 2, pp. 8–18, Mar./Apr. 1999.

[2] C. Aurrecoechea, A. T. Campbell, and L. Hauw, "A survey of QoS architectures," *Multimedia Syst.*, vol. 6, no. 3, pp. 138–151, May 1998.

[3] X. Masip-Bruin, M. Yannuzzi, J. Domingo-Pascual, A. Fonte, M. Curado, E. Monteiro, F. Kuipers, P. Van Mieghem, S. Avallone, G. Ventre, P. Aranda-Gutiérrez, M. Hollick, R. Steinmetz, L. Iannone, and K. Salamatian, "Research challenges in QoS routing," *Comput. Commun.*, vol. 29, no. 5, pp. 563–581, 2006.

[4] A. Striegel and G. Manimaran, "A survey of QoS multicasting issues," *IEEE Commun. Mag.*, vol. 40, no. 6, pp. 82–87, Jun. 2002, doi: 10.1109/MCOM.2002.1007412.

[5] Y. Chen, K. Wu, and Q. Zhang, "From QoS to QoE: A tutorial on video quality assessment," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 2, pp. 1126–1165, 2nd Quart., 2015, doi: 10.1109/COMST.2014.2363139.

[6] S. Ehsan and B. Hamdaoui, "A survey on energy-efficient routing techniques with QoS assurances for wireless multimedia sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 2, pp. 265–278, 2nd Quart., 2012.

[7] J. W. Guck, A. Van Bemten, M. Reisslein, and W. Kellerer, "Unicast QoS routing algorithms for SDN: A comprehensive survey and performance evaluation," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 388–415, 1st Quart., 2018.

[8] M. Karakus and A. Durresi, "Quality of service (QoS) in software defined networking (SDN): A survey," *J. Netw. Comput. Appl.*, vol. 80, pp. 200–218, Feb. 2017.

[9] X. Guo, H. Lin, Z. Li, and M. Peng, "Deep-reinforcement-learning-based QoS-aware secure routing for SDN-IoT," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6242–6251, Jul. 2020.

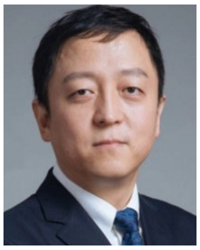
[10] R. Y. Chen and L. Cui, "Research on network routing optimization and service quality assurance technology for ocean network," Shenzhen Univ., Shenzhen, China, Tech. Rep., 2023.

[11] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet topology zoo," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765–1775, Oct. 2011.

[12] J. T. Moy, *OSPF: Anatomy of an Internet Routing Protocol*. Reading, MA, USA: Addison-Wesley Professional, 1998.

[13] R. H. Jhaveri, R. Tan, and S. V. Ramani, "Real-time QoS-aware routing scheme in SDN-based robotic cyber-physical systems," in *Proc. IEEE 5th Int. Conf. Mechatronics Syst. Robots (ICMSR)*, May 2019, pp. 18–23.

TIAN JI was born in Jingdezhen, China, in 1989. He received the M.S. degree in big data from Tsinghua University, in 2023. He is currently a Core Member of the Future Internet Research Center, Research Institute of Tsinghua University in Shenzhen. He has participated in digital transformation projects for multiple local governments and leading state-owned enterprises, covering various aspects, including but not limited to data management and analysis, system integration, network architecture, and application development. His research interests include big data, cloud computing, and edge computing technologies, such as smart cities and the industrial internet.

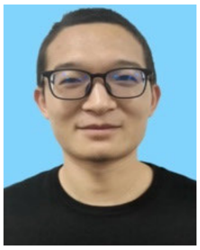


XIAOLEI CHANG was born in 1975. He received the bachelor's degree in computer science and technology, the master's degree in management science and engineering, and the Ph.D. (Engineering) degree in electronic information from Tsinghua University, in 1999 and 2001, respectively.

He is currently a Senior Engineer with the Information Technology Center/Network Research Institute, Tsinghua University. He is also the Deputy Director of the Next Generation Internet Research and Development Center, Research Institute of Tsinghua University in Shenzhen. He has been engaged in information technology application research and engineering practice, transformation of scientific, and technological achievements and technology enterprise incubation for a long time. He has completed more than 20 research projects, published more than 20 papers and two books. His research interests include green data centers, cloud computing, edge computing, and cyberspace security.

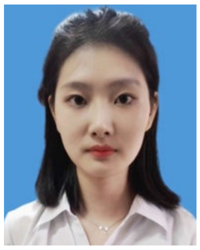
ZHENGJIAN CHEN was born in Huai'an, Jiangsu, China, in 1981. He received the bachelor's degree in engineering from Xi'an Jiaotong University, in 2004, the master's degree in engineering from the Harbin Institute of Technology, in 2012, and the master's degree in economics from Peking University, in 2019.

He has been a Technician with the Shenzhen Energy Group, since 2004, and obtained the title of a Senior Engineer, in 2012. In 2020, he was the Head of Smart Energy Technology and the President of the Shenzhen Energy Group, Smart Energy Research Institute. He is also the lead of the research and development of integrated energy dispatching systems, intelligent power plants, virtual power plants, and other system platforms. He has participated in two monographs, published 15 papers and 20 patents. His research interests include integrated energy, microgrids, energy digitization, the application of artificial intelligence, and 5G in the energy industry.



ZHENZHOU WU received the bachelor's degree in software engineering from Jilin University, in 2009. He is currently a senior architect, a lisp programmer, and the research and development director of several internet companies. From 2009 to 2011, he was a Search Engineer with Avepoint, responsible for the company's full-text search department. From 2011 to 2014, he was a Background Architect with Fenglin Volcano Technology Company Ltd., supporting the game platform with tens of millions of users and millions of users online at the same time. From 2014 to 2017, he was the Research and Development Director of Oudmon Technology Company Ltd., to develop the IoT platform, which has access to more than 50 million wearable devices. Since 2017, he has been the Research and Development Director of the Next Generation Internet Research and Development Center, Research Institute of Tsinghua University in Shenzhen, dedicated to the field of cloud native, distributed networks, and distributed computing.

RUIYU CHEN received the B.S. degree from South China Normal University, Guangzhou, China, in 2019. He is currently pursuing the M.S. degree with Shenzhen University. His research interests include software-defined networks, segment routing, and traffic engineering.



CHENXI LI was born in Chenzhou, Hunan, China, in 1999. She received the bachelor's degree in materials science and engineering from the China University of Mining and Technology, in 2020. Since 2021, she has been a Research Assistant with the Next Generation Internet Research and Development Center, Research Institute of Tsinghua University in Shenzhen. Her research interests include the development and application of 5G, edge computing, and other new generation information technologies.

CHEN XU was born in Changde, Hunan, China, in 1988. She received the bachelor's degree from the Hunan University of Science and Technology, in 2011. Currently, she is a Researcher with the Next Generation Internet Research and Development Center, Research Institute of Tsinghua University in Shenzhen. Her research interests include big data analysis, network slicing, and SD-WAN.