Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1231979, 11 pages
https://doi.org/10.1155/2022/1231979

Research Article

Feudal Multiagent Reinforcement Learning for Interdomain Collaborative Routing Optimization

Zhuo Li, [1,2,3] **Xu Zhou**, [1,2] **Filip De Turck**, [3] **Taixin Li**, [1] **Yongmao Ren**, [1] **and Yifang Qin** [1]

[1] *Computer Network Information Center, Chinese Academy of Sciences, Beijing, China*
[2] *University of Chinese Academy of Sciences, Beijing, China*
[3] *Ghent University - imec, IDLab, Department of Information Technology, Gent, Belgium*

Correspondence should be addressed to Xu Zhou; zhouxu@cnic.cn

Received 22 December 2021; Revised 17 February 2022; Accepted 2 March 2022; Published 27 March 2022

Academic Editor: Narasimhan Venkateswaran

In view of the inability of traditional interdomain routing schemes to meet the sudden network changes and adapt the routing policy accordingly, many optimization schemes such as modifying Border Gateway Protocol (BGP) parameters and using software-defined network (SDN) to optimize interdomain routing decisions have been proposed. However, with the change and increase of the demand for network data transmission, the high latency and flexibility of these mechanisms have become increasingly prominent. Recent researches have addressed these challenges through multiagent reinforcement learning (MARL), which can be capable of dynamically meeting interdomain requirements, and the multiagent Markov Decision Process (MDP) is introduced to construct this routing optimization problem. Thus, in this paper, an interdomain collaborative routing scheme is proposed in interdomain collaborative architecture. The proposed Feudal Multiagent Actor-Critic (FMAAC) algorithm is designed based on multiagent actor-critic and feudal reinforcement learning to solve this competition-cooperative problem. Our multiagent learns about the optimal interdomain routing decisions, focused on different optimization objectives such as end-to-end delay, throughput, and average delivery rate. Experiments were carried out in the interdomain testbed to verify the convergence and effectiveness of the FMAAC algorithm. Experimental results show that our approach can significantly improve various Quality of Service (QoS) indicators, containing reduced end-to-end delay, increased throughput, and guaranteed over 90% average delivery rate.

1. Introduction

With the explosive growth of Internet traffic, network resources have been stressed due to the differentiated network requirements and sudden requirements, which has also led to an increase in the demand for interdomain transmission, network operation, and maintenance and management [1]. There are further requirements for the role of the network, requiring its QoS to be stable and controllable. To support more new applications on limited distributed resources, an efficient collaboration mechanism between Autonomous Systems (ASs) has become the key to solving the traffic surge in interdomain. BGP is an interdomain routing protocol for AS, used to exchange routing information between different AS during interdomain transmission [2]. To ensure good scalability and flexibility, BGP hides the internal information of the AS, including routing strategy, internal topology, and link bandwidth. However, BGP's opaque characters hinder the collaboration of the AS, which makes it challenging to guarantee end-to-end communication QoS [3]. With the emergence of centralized SDN technology, it is no longer necessary to exchange information through BGP but to exchange routing information through the SDN controller, which is deployed in each AS, and make interdomain routing decisions based on the collected topology information of the entire network [4]. There are still issues that contain interconnection, high latency, flexibility, and flexibility issues, whether the improvement mechanism is based on standard BGP or an interdomain routing approach employing SDN technology.

Through coming up with some heuristics to solve the simplified decision-making problem, like genetic algorithm

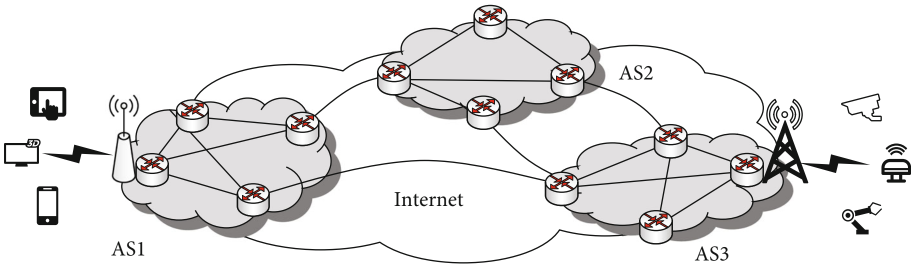

Figure 1: Interdomain routing on the interconnected AS.

(GA), Simulated Annealing Algorithm (SAA), or other various heuristics, we usually need to make a lot of testing and parameter adjustments to make things work as expected [5]. Reinforcement learning does not require prior knowledge other than environmental reward information and designs algorithms that learn to make better decisions by interacting with an environment. With the continuous progress of related theories and technologies, general reinforcement learning algorithms such as Deep Q-learning (DQN), Advantage Actor-Critic (A2C), and Deep Deterministic Policy Gradient (DDPG) have been introduced into routing optimization [6]. When the above-mentioned single-agent reinforcement learning algorithms are directly applied to a multiagent environment such as interdomain routing optimization, it is easy to cause nonstationary problems and make it difficult for training to converge. Multiagent reinforcement learning (MARL) combines multiagent learning and reinforcement learning, focusing on the sequential decision-making of multiple agents, and is aimed at using reinforcement learning to build a system in which multiple agents interact in the same environment to solve distributed decision-making control problems such as interdomain routing optimization problems [7]. However, the huge combination of state spaces, huge action spaces, and sparse rewards in MARL makes it difficult to train a good online decision model and feudal reinforcement learning can solve problems such as generalization and learning speed very well [8]. The main contributions of this paper are summarized as follows:

(1) This paper studied the problem of interdomain routing and proposed an interdomain collaborative architecture, and the MARL framework for collaborative routing is installed in the interdomain routing environment

(2) After proposing a collaborative routing model based on multiagent MDP, we introduce the multiagent actor-critic architecture and feudal reinforcement learning and construct a corresponding hybrid algorithm to solve the mapping collaborative routing optimization problem

(3) We evaluate the performance of the proposed approach compared to other baselines, and experimental results show that the proposed approach can decrease the complexity compared to the conventional MARL methods and achieve better performance than previous routing schemes

The remainder of this paper is organized as follows. In Section 2, we analyse the related works on interdomain routing and MARL. In Section 3, we detail the proposed interdomain collaborative architecture and the collaborative routing optimization model. In Section 4, we describe the FMAAC algorithm for collaborative routing based on multiagent actor-critic and the feudal reinforcement learning. Experiment results validate the performance of the FMAAC algorithm in Section 5. Finally, we conclude the paper and discuss the future works in Section 6.

2. Related Works

2.1. Interdomain Routing. On the Internet divided by AS, the routing problem can be divided into routing within a single AS and routing between ASs, that is, intradomain routing and interdomain routing. As shown in Figure 1, each AS can run any routing protocol on the Internet containing a collection of interconnected AS. Interdomain routing is used to solve the reachability of routing information, which works and needs to know about other routers within and between their AS [8]. BGP is a distance vector routing protocol that provides reachable routes and no loops between AS, which uses the path length between ASs as the main factor in most optimal path calculations. Due to the emergence of lightweight independent networks, the length of AS path in BGP has increased improperly, resulting in a decrease in real-time network traffic routing efficiency [9].

Interdomain routing optimization has been studied for a long time, and plenty of optimization schemes have been proposed. Aiming at the BGP's convergence delay, Alabdulkreem et al. [10] calculate the optimal value of the advertisement interval to minimize the convergence time without increasing the number of advertising messages. There can be competing factors to consider, such as completion time and resource utilization, which can conflict with each other, and it is hard to draw a balance. In fact, this problem is general NP-hard. Xiang et al. [11] systematically formulate the software-defined internetworking model and develop a Bayesian optimization algorithm to solve the NP-hard routing problem, which can find a near-optimal policy-compliant end-to-end route by sampling. To achieve better Internet real-time experience, Arins [12] proposed to store

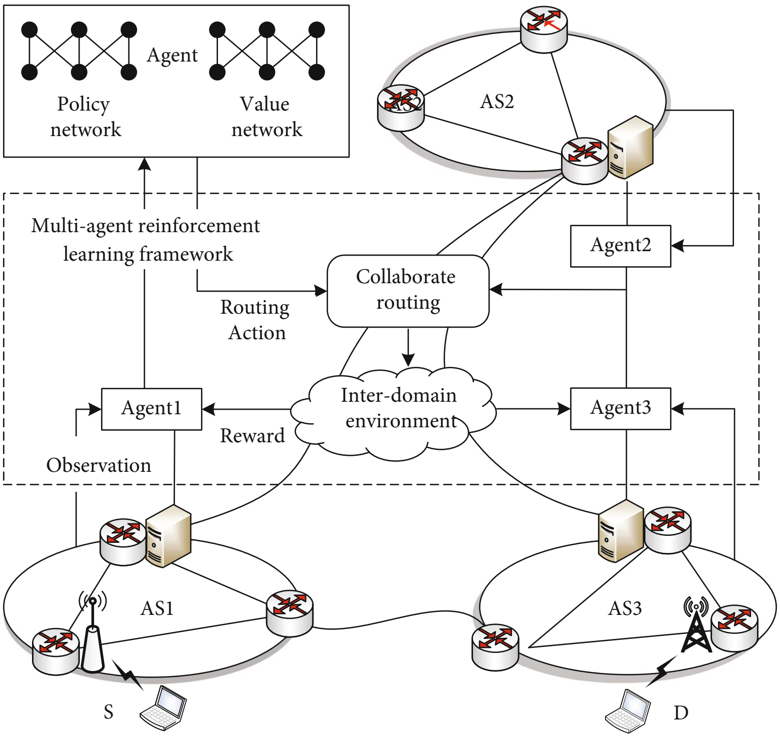

Figure 2: Interdomain collaborative architecture with the MARL framework.

the bilateral latency measurements in a decentralized blockchain network, and the SDN controller can route according to the shared data in the block. Zhong et al. [13] propose a novel multidomain routing paradigm that transforms the routing problem from heuristic-algorithm-based computation to artificial-intelligence-based data analytics.

2.2. Multiagent Reinforcement Learning. Reinforcement learning is very suitable for solving decision-making problems and has apparent advantages in routing optimization. Sun et al. [14] propose an intelligent network control architecture based on deep reinforcement learning that can dynamically optimize centralized routing strategies. Distributed optimization, game theory, and MARL have been proposed to solve the problems of inaccurate information prediction, high complexity, high cost, and poor scalability faced by centralized methods. Independent Q-Learning (IQL) is a MARL algorithm extended by the DQN, which executes a Q-learning algorithm on each agent, while the environment of each agent is dynamic and unstable, and this algorithm cannot converge [15]. The Centralized Training with Decentralized Execution (CTDE) framework has become the industry standard to reduce unnecessary costs imposed by agent communication [16]. Value Decomposition Networks (VDN) adopt the integration of the value function of each agent to obtain a value function of the joint action [17]. Although the CTDE framework is implemented, its joint action-value function cannot express some complex environments well.

Multiagent Deep Deterministic Policy Gradient (MADDPG) is an actor-critic-based CTDE algorithm, using DDPG as the underlying reinforcement learning algorithm for each agent [16]. Each agent is configured with a separate actor responsible for the generation of actions, and an independent critic is responsible for judging the actions made by the actor to assist in the training of the model. Multiagent-Attention-Critic (MAAC) uses a centralized critic and attention mechanism to integrate information from all agents for more efficient training [18]. Counterfactual multiagent policy gradients (COMA) distinguish the actual value of actions made by each agent through the counterfactual baseline, and credit assignment is realized by determining the contribution of each agent in collaboration [19]. The abovementioned MARL algorithms provide a good idea for solving interdomain routing optimization problem, while directly applying the MARL algorithms to the interdomain routing optimization problem is likely to generate issues such as high training difficulty, low training efficiency, and robustness, and further optimization is required to make it adapt to the interdomain routing environment.

3. System Model and Problem Formulation

In this section, we introduce the system model and formulation of the collaborative routing optimization problem with networked agents.

3.1. Interdomain Collaborative Architecture. We propose an interdomain collaborative architecture, as shown in Figure 2; the MARL framework for collaborative routing is installed in the interdomain routing environment. In the interdomain network scenario, an agent is installed in each AS, in which the policy network and the value network make and assess routing actions based on their observations of the

Table 1: Notations.

Notation	Meaning
G	Graph
i, j	Agent
t	Time period
μ	Agent collection
ν	Link collection
N	Number of agents
S	Interdomain global state
A	Action space
O	Observation function
R	Reward function
T	Transition function
π	Policy
γ	Discount factor

AS's environment, respectively. By sending routing actions to the relevant network controller for forwarding, calculating the overall reward and the individual reward of each agent according to the transmission result, and adjusting the next round of routing actions by maximizing the overall reward, the collaborative routing obtains the global optimal interdomain routing. In each time-step, agents make decisions jointly with other agents through information interaction and learn how to generate routing information based on their own local observations during the training process or to determine whether the communication is needed and which agents to communicate with [20]. At the same time, it must learn and find the maximum reward given by the environment to obtain the best strategy. In the process of running after training, it is necessary to explicitly make routing decisions based on the information transmitted by other agents.

Take a simple service as an example in Figure 2, when the source user located in the AS1 needs to transmit data to the destination user in AS3, its network transmission requirements are recognized by the agent in AS1, and the agent in AS1 generates a service identifier according to the characteristics of the requested service, which triggers pathway AS to coordinate the software and hardware resources in their respective network domains to meet users' service requirements. Specifically, the agent in AS1 updates the forwarding table by issuing forwarding messages. Then, it is necessary to coordinate the network resources of the AS that need to pass through to meet interdomain transmission requirements, update network status information, and feed resource usage. Finally, the agent in AS3 obtains users' service requirements and coordinates the network resources to meet them. In general, the above combinatorial optimization problem can be reduced to a classic discrete optimization problem, and it can be solved in polynomial time by linear programming, genetic algorithm, reinforcement learning, and other methods.

3.2. Multiagent MDP. Considering a system of many agents operating in the interdomain routing environment, we assume that the interdomain state and routing actions can be observed by all agents, and only the rewards are unique to each agent [21]. Since single-agent reinforcement learning can be formalised in terms of MDP, we then model the optimization problems of interdomain routing as multiagent MDP. The main notations used in this work are summarized in Table 1. The interdomain collaborative architecture with the MARL framework can be represented by an undirected graph $G(\mu, \nu)$, where $i \in \mu$ represents each agent and $(i, j) \in \nu$ represents each interdomain link.

Multiagent MDP can be represented by a tuple $(G, N, S, A, O, R, T, \pi)$, where N represents the number of agents, S represents the global state space shared by all agents deployed in the interdomain collaborative architecture, A represents the specific action space of all agents at each step, O represents the collection of observations of all agents, R represents the corresponding rewards, T represents the state transition probability, and π represents the routing policy. Specifically, we address the action, state, and reward definitions in the multiagent MDP model.

(1) State definition: each agent can observe the flow state of its own AS, and we define the observations of each agents as O_i. The more comprehensive the information observed, the better the decisions made. To extend the observable range, more state information can be obtained from neighbor ASs or even all, but this also adds additional communication. Here, we only consider that the state is affected by the one-hop neighborhood, and the state is composed of its own observed state and the routing actions with its neighboring agents that are one hop away from it

(2) Action definition: for flows with different destinations, the set of candidate next hops advertised by their routes may be different, so we define the set of next hops of all forwarded flows in the next round as the action space. Let $\{A_i\} = \prod A_i$ represent the action space of all agents and $A_i = (a_1, a_2, a_3, \cdots a_N)$ represent the specific actions of all agents at each step. The policy π_i of each agent i is determined by the policy network and value network, and the state transition probability $T : S \times A \longrightarrow [0, 1]$ is designed to transfer all agents to a new state after performing actions in the current state. Each agent i follows a decentralized policy $\pi_i : S_i \times A_i \longrightarrow [0, 1]$ to choose its own action $A_{i,t} \sim \pi_i(\cdot \mid S_{i,t})$ at time t

(3) Reward definition: according to the current state S and the actions A_i made by each agent i, the agent can receive corresponding rewards $R_i : S_i \times A_i \longrightarrow \mathbb{R}$ from the environment. The objective of multiagent MDP is to find an optimal policy for each agent, and we can maximize the sum of its future expected rewards:

$$\mathbb{E}[R_0^{\pi}] = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t \mathrm{R}_i^t\right], \tag{1}$$

where T is the cumulative number of expected rewards in the future and γ is the discount factor, which is usually set to a number slightly less than 1

3.3. Collaborative Routing Optimization. Whether it is a collaborative environment, a competitive environment, or a mixed environment, the core of the research in the field of multiagents is how to solve the instability problem in the actual environment. The interdomain routing environment is obviously a mixed environment, and the agents in each AS have a game relationship, which means that the change of the routing policy made by one agent will lead to the adjustment of the routing polices of all other agents. Let $\theta = [\theta_1, \theta_2, \theta_3, \cdots, \theta_N] \in P$ indicate the parameters of the collection of the agent's polices and $\pi(A \mid S)$ represent the parameterized policy. Then, the average reward $\mathbb{R}(\theta)$ for all agents under policy θ is

$$\mathbb{R}(\theta) = \lim_{T} \frac{1}{T} \mathbb{E}\left(\sum_{t=0}^{T-1} \frac{1}{N} \sum_{i \in N} R_{t+1}^{i} \right) = \sum_{i=1}^{N} \pi_{\theta}^{i}(s, a) R_{\theta}^{i}(s, a). \quad (2)$$

Correspondingly, we define the expected reward obtained after performing a routing action under the state as the global differential action-value function, and this function is shared by all agents:

$$Q_{\theta}(S, A) = \mathbb{E}[R_t^{\pi} \mid S_t = S, A_t = A]. \quad (3)$$

We further define the state-value function to reflect the current state; it is the action-value function's expectations about actions:

$$V^{\pi}(S) = \mathbb{E}[R_t^{\pi} \mid S_t = S]. \quad (4)$$

According to the definition of the multiagent MDP in the above chapters, the objective of multiagent MDP is to find an optimal policy for each agent; then, we can further define the collaborative routing optimization problem of finding a policy collection θ such that reward is maximized.

$$\max_{\theta \in \mathbb{P}} \mathbb{R}(\theta) = \max_{\theta \in \mathbb{P}} \sum_{i=1}^{N} \pi_{\theta}^{i}(s, a) R_{\theta}^{i}(s, a). \quad (5)$$

However, the above global function requires the rewards of all agents to be unbiasedly estimated, so we need to design a MARL algorithm based on consistency constraints. The algorithm can spread local information between agents, thereby promoting the establishment of cooperative relations between agents.

4. Feudal Multiagent Reinforcement Learning

This section builds the corresponding algorithm based on the feudal reinforcement learning framework and multiagent actor-critic to solve the formulated optimization problem.

4.1. Feudal Reinforcement Learning. Due to the enormous state space combination, huge action space combination, and sparse reward in multiagent MDP, a good model cannot be obtained directly through training. Concerning the experience of solving other complex problems, the optimization problem can be decomposed into several easy-to-solve sub-optimization problems, which is hierarchical reinforcement learning, which can be divided into two types according to hierarchical methods [22]. One is based on goals, and the primary method is to select a specific goal so that the agent trains toward these goals. It can be foreseen that the difficulty of this method is how to select a suitable goal. The other is multilevel control, whose practice abstracts different control layers and controls the lower layer via the upper layer [23]. Feudal reinforcement learning is a typical multilevel control, and its control level is divided into three levels. The current level is manager, the upper level is supermanager, and the next level is submanager [24]. When the feudal mechanism is adopted, each element in the defined initially multiagent MDP needs to be redefined from both managers and workers in feudal reinforcement learning [25].

The state space of the managers directly uses the state space of the original multiagent MDP, and the state of the multiagent MDP and the goal generated by the managers are used as the state space of the workers. Regarding the action space of the managers, the option-critic framework solution in the MAAC framework is that the action space is used to select different workers to perform operations. For example, the managers first select the first worker to execute and then execute the second worker. To fit the interdomain collaborative routing environment, the managers set a global collaborative optimization goal and then allow the workers to execute it while the workers' action space remains unchanged. The managers' reward can directly use the original multiagent MDP's reward. The long-term credit assignment makes the managers' goals cruder on the time scale so that the original sparse reward is not so sparse in the managers' view. However, the workers' reward cannot directly use the reward of the original problem but can use the more intensive reward evolved from the goal generated by the managers. In addition, the workers' transition model and discount rate remain unchanged, while the transition function of the managers will change. The managers pay more attention to long-term rewards to have a more significant discount rate than the workers by transforming into a transition policy gradient.

4.2. Multiagent Actor-Critic. Temporal-difference learning is one of the most widely used learning methods in reinforcement learning, which combines the Monte Carlo methods and the dynamic programming and can be divided into value-based reinforcement learning and policy-based reinforcement learning. Policy gradient uses a policy neural network to generate the agent's policies, which increases the probability of the agent taking actions that can get higher rewards and reduce the probability of the agent taking actions that get lower rewards by constantly updating the policy neural network. Actor-critic is a reinforcement learning method that combines temporal-differential learning and policy gradient. The actor refers to the policy function, and the critic refers to the value function. With the help of the value function, the actor-critic can update the parameters in a single step, without waiting for the end of the round to update [26]. MARL is focusing on how to generate the correct and optimal policy update gradient. However, the policy gradient usually performs poorly in a multiagent environment, as there are usually significant differences in multiagent collaboration. In a cooperative and competitive environment taking interdomain collaborative routing as an example, the reward of each agent usually depends on the actions of many other agents. To maximize the cumulative reward, we can write the policy gradient of the expected cumulative reward for singer agent as

$$\nabla_\theta \mathbb{R}(\theta) = \mathbb{E}_{S,A}[\nabla_\theta \log \pi(A \mid S) Q^\pi(S, A)], \tag{6}$$

where the parameterized function $Q^\pi(S, A)$ is called critic and $\pi(A \mid S)$ is called actor. Only the reward of the agent's own actions shows more variability, thereby increasing the variance of its gradient.

CTDE is a MARL framework can alleviate instability, whose central controller only conducts training, which is turned off during the execution phase, and each agent makes its own decisions [27]. MAAC is an extension of singer-agent actor-critic by adopting the CTDE framework, where the critic of each agent can obtain the action information of all other agents [28]. In the model training phase, the critic completes the centralized training, which equals the number of agents. By centralizing the data of all agents, the critic gives each agent a relatively stable global future reward expectation. At the same time, the actor can also use the information provided by centralized critic for training. In the model operation stage, each agent has its actor responsible for generating the basic policy, and the critic no longer needs to participate in the decision-making. Let $\pi = [\pi_1, \pi_2, \pi_3, \cdots, \pi_N]$ indicate the set of all agents' policies; we can rewrite the policy gradient of the expected cumulative reward for each agent as

$$\nabla_\theta \mathbb{R}(\theta) = \mathbb{E}_{S,A}[\nabla_\theta \log \pi(A \mid O) Q^\pi(o_1, \cdots, o_N, a_1, \cdots, a_N)], \tag{7}$$

where $Q^\pi(o_1, \cdots, o_N, a_1, \cdots, a_N)$ represents the state-action function, which takes centralized data as input, and $O_i = (o_1, o_2, o_3, \cdots o_N)$ contains the observation information of the AS where the agent is located and other observable state information. Let $y = (R + \gamma Q'(o_1', \cdots, o_N', a_1', \cdots, a_N'))$ indicate the target value; all critics are updated together to minimize a joint regression loss function by sharing parameters:

$$L(\theta) = \mathbb{E}_{S,A}\left[(Q^\pi(o_1, \cdots, o_N, a_1, \cdots, a_N) - y)^2\right], \tag{8}$$

where $A_i' = (a_1', \cdots, a_N')$ is the output value of the target policy network in the next state. Since each agent learns its state-action function independently, each agent can have a different reward function, which can complete cooperation or competition tasks in interdomain collaborative routing scenarios [29].

4.3. Feudal Multiagent Actor-Critic. Combine the feudal reinforcement learning into the MAAC framework, which can integrate the centralized data of critic more hierarchically and reasonably and has more advantages when the number of agents rises. We proposed Feudal Multiagent Actor-Critic (FMAAC), which is an extension of MAAC with feudal hierarchy for collaborative routing optimization, and the procedures of the proposed approach are shown in Algorithm 1. At each time-step, the manager selects an action from its current policy and exploration, and the worker selects and executes an action. After executing a round of routing actions, the interdomain routing environment gives the manager and the worker corresponding reward R^M and R^W, respectively. The initial state, managers' and workers' actions, rewards, and new state are stored in replay buffer B^M and B^W, respectively. Then, the policy gradient is used to update managers' and worker's actor-critic networks. Finally, the update method of the target network parameters θ_i is the Soft update method, λ is the target update coefficient, and each parameter is updated to the current policy network in a small amount [30].

5. Simulation Experiments

5.1. Experimental Setup. The experimental environment of interdomain collaborative routing was developed based on ns3-gym, which is an open-source project for RL research written in Python [31]. As shown in Figure 3, ns3-gym is a framework that integrates both OpenAI gym and ns-3 network simulator [32, 33]. The interdomain testbed is a multidomain network simulator based on ns-3, and the Interprocess Communication (IPC) between the different machines of the interdomain testbed and the ns-3 network simulator is Socket [34], and we used a network size of $1600 \times 1600\,\mathrm{m}^2$ in the ns-3 network simulator [35]. In terms of the experimental hardware platform, the CPU is Intel Xeon E5 2630, the GPU is NVIDIA GeForce RTX 2080 Ti, the memory is 64 GB DDR4, the installed operating system is Ubuntu 18.04.5 LTS, and the reinforcement learning framework is PyTorch. We implement the proposed FMAAC algorithm using the Multiagent Particle Environments (MAPE), which is an open-source framework for building a multiagent testbed based on OpenAI gym [16]. On condition that the number of agents and training parameters is designed according to the interdomain collaborative routing environment and the representation of state information, rewards, and ending conditions is designed according to actual needs in MAPE, the framework will automatically generate a function interface for the MARL algorithm. Based on the above experimental parameter settings, we compare the proposed FMAAC algorithm with the following three latest benchmark algorithms: the MARL algorithms with CTEDE framework represented by MADDPG and MAAC [16, 18].

MADDPG is a multiagent policy gradient algorithm in which the agent learns a centralized critic based on the observations and actions of all agents [16]. MAAC algorithm introduces the attention mechanism so that the centralized data used by the critic can be more rationally integrated,

```
Input: Initialize inter-domain environments with $N$ agents contained managers and workers
Output: All managers' and workers' routing policies
for each episode $\alpha = 1$ to $\mu$ do
    Initialize a random process $N$ for routing actions exploration, get the initial state $S$
    for each time-step $t = 1$ to $\upsilon$ do
        Managers select an action $A_i^M$ under the current policy $\pi_t$ and exploration
        Workers select and execute an action $A_i^M$
        Receive the reward $R^M$, $R^W$ and observe the next newly state $S'$
        Store replay buffer $B^M \longleftarrow \{S, A^M, R^M, S'\}$ and $B^W \longleftarrow \{S, A^W, R^W, S'\}$
        for each agent $j = 1$ to $N$ do
            Sample a random minibatch from $B^M$ and $B^W$
              Update actor by using policy gradient:
                  $\nabla_\theta J(\theta) = \mathbb{E}_{S,A}[\nabla_\theta \log \pi(A \mid O) Q^\pi(o_1,\cdots,o_N, a_1,\cdots,a_N)]$
              Update critic by minimizing the loss:
                  $L(\theta) = \mathbb{E}_{S,A}[(Q^\pi(o_1,\cdots,o_N, a_1,\cdots,a_N) - (R + \gamma Q'(o'_1,\cdots,o'_N, a'_1,\cdots,a'_N)))^2]$
              end for
          Update target network parameters:
              $\theta'_i = \lambda\theta_i + (1 - \lambda)\theta'_i$
    end for
end for
```

Algorithm 1: Feudal Multiagent Actor-Critic for collaborative routing.

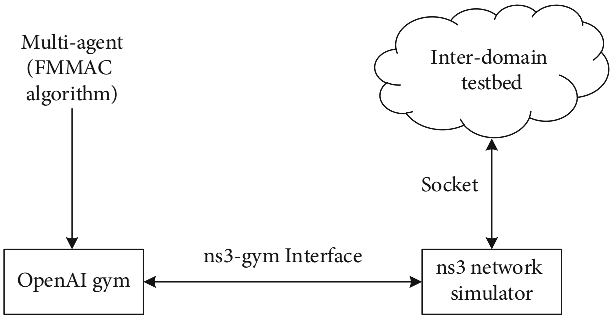

Figure 3: Interdomain routing environment based on ns3-gym.

Table 2: Training parameters.

Parameters	Value
Optimizer	Adam
Actor learning rate	0.01
Critic learning rate	0.01
Minibatch size	512
Target update coefficient	0.01
Discount factor	0.95
Network hidden units	64
Activation function	ReLU
Memory pool size	10^6

which also uses MAPE as the test environment, so this algorithm is selected as the baseline for performance comparison [18]. In the process of model training, FMAAC, MADDPG, and MAAC use the same parameter settings as shown in Table 2 [36]. All use the most commonly used Adam optimizer, which has the advantages of fast convergence speed and easy parameter adjustment, and the learning rate of actor and critic is set to 0.01.

The minibatch size extracted each time during training is 512, the target update coefficient in the Soft update method is set to 0.01, and the discount factor of the expected reward is set to 0.95. Critic and policy networks are represented by the three layers of Multilayer Perceptron (MLP), each layer has 64 hidden units, ReLU is used as the activation function, and the memory pool size is 10^6.

5.2. Experimental Results

5.2.1. Training Results. The experiment first verifies the convergence performance of the FMMAC algorithm in the offline training, and we take the average rewards to highlight the convergence performance. As shown in Figure 4, we plot the learning curves with error bars over 50000 training episodes for the proposed FMAAC algorithm and the MADDPG and MAAC algorithms as the baselines, which illustrate the average rewards per training episode. It can be clearly seen that when all algorithms enter the convergence stage, the value of the average episode rewards of the FMAAC algorithm is higher than that of other baseline algorithms. The MADDPG algorithm cannot find a better policy due to its relatively large observation space for all agents, and its performance is at the bottom. MAAC has made a good trade-off in the exploration-exploitation, while each agent's local observation cannot provide enough information to make an optimal prediction of its expected rewards. Compared with the baseline algorithms, the FMAAC algorithm can infer the decisions of other agents more accurately due to the existence of feudal control, which can achieve more efficient cooperation and achieve better results in an interdomain collaborative routing environment. For the training of the interdomain routing optimization model, the proposed

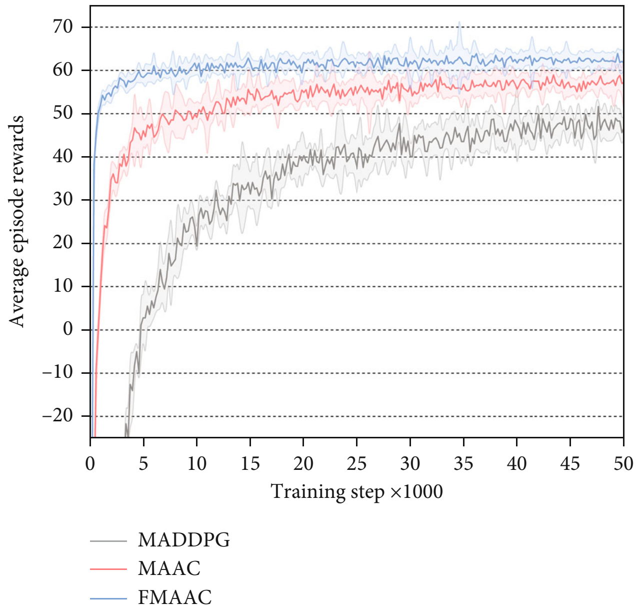

Figure 4: Average episode rewards on the interdomain testbed.

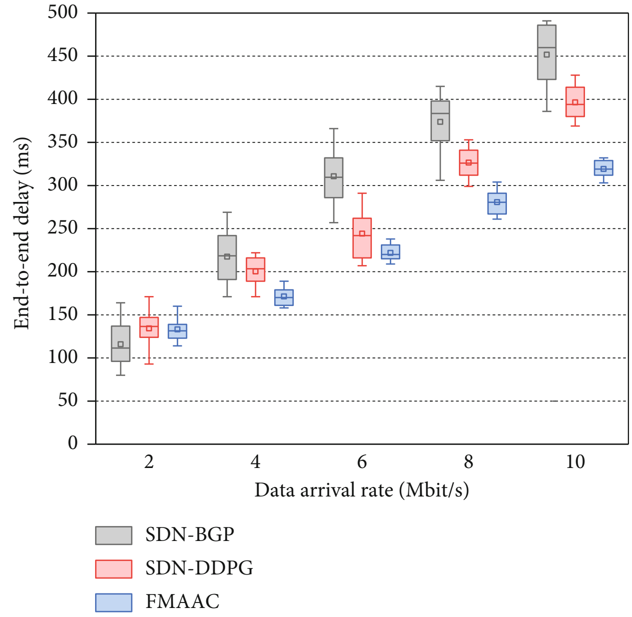

Figure 5: End-to-end delay at different data arrival rate.

method combines the properties of feudal reinforcement learning to partition tasks spatially rather than temporally, allowing it to be successfully applied to large discrete action spaces in reinforcement learning tasks, which can better meet the requirements of interdomain collaborative routing optimization than the other two baseline algorithms.

5.2.2. Evaluation Results. In verifying the advantages of the multiagent algorithm, the FMAAC is compared with SDN-DDPG and SDN-BGP in the routing optimization performance [37, 38]. SDN-DDPG uses single-agent DDPG to optimize SDN to reduce network operating delay and increase throughput, while the SDN-BGP improves BGP performance through centralized control in SDN. Experiments first pay attention to the performance of FMMAC on end-to-end delay, which is a significant parameter for evaluating the performance of routing schemes. To get closer to the actual network scenario, we use five different data arrival rates in the experiment. For each data arrival rate, a corresponding flow matrix is generated for the training and testing of the FMAAC and SDN-DDPG, and the model after training 50000 steps is used as the comparison object.

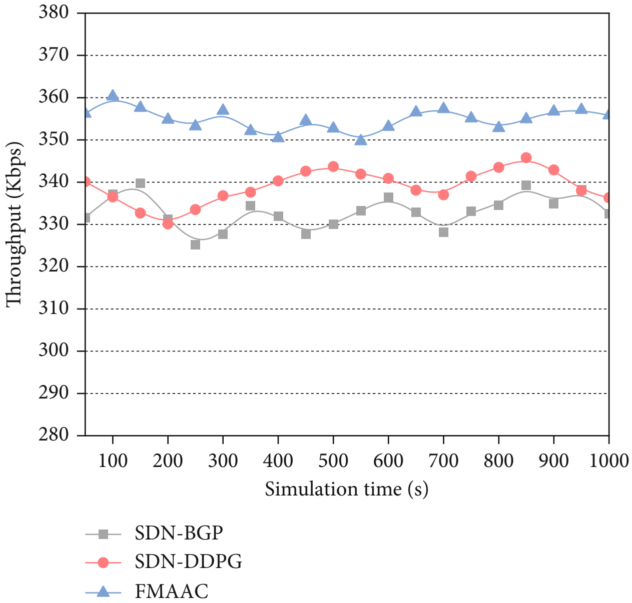

Figure 6: Average throughput at different simulation time periods.

In order to make the observed end-to-end delay more convincing, the three schemes in our experiment sampled the value of the end-to-end delay 10 times. The comparison results of the three schemes are shown in Figure 5 in the form of box plots. The upper and bottom parts of the rectangle in the figure represent the upper quartile and lower quartile of the observed end-to-end delay obtained from the experiment, the horizontal line in the middle of the rectangle represents the median of the experimental observation data, and the upper and lower ends of the straight line extending from the rectangle represent the maximum and minimum values of the observation data in the inner limit range. In addition, invalid data outside the inner limit range is not displayed for brevity. Experimental results show that the end-to-end delay of the route configuration optimized by FMMAC is less than the delay of the SDN-BGP generating route and the delay of SDN-DDPG optimized route generation, which verifies the effectiveness of the FMMAC optimized interdomain routing.

Next, we carried out a simulation experiment on throughput. In this experiment, we use the same data arrival rate and the size of packets and take the real-time throughput in the interdomain routing environment as the optimization goal. SDN-DDPG and FMMAC have been trained 50000 times in the early training stage. The performance of the three mechanisms in throughput is shown in Figure 6. It can be seen obviously that the real-time throughput range

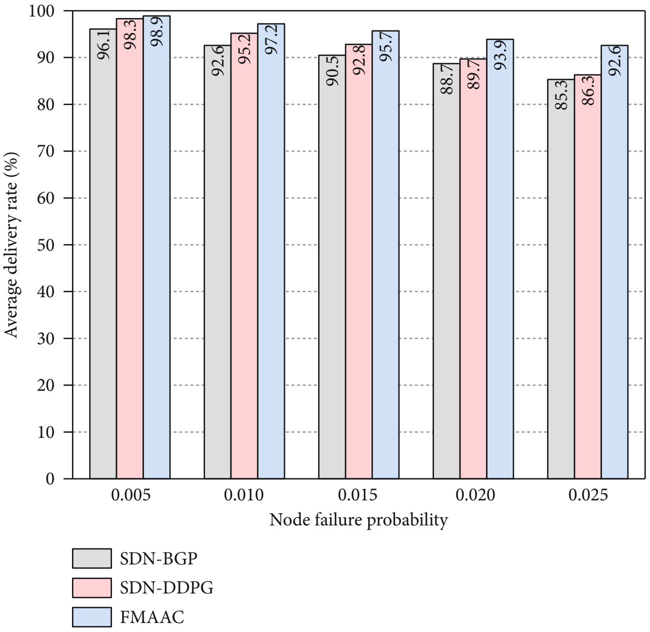

Figure 7: Average delivery rate vs. node failure probability.

of SDN-BGP and SDN-DDPG ranges from 325 Kbps to 345 Kbps while using FMMAC as a routing mechanism increases the throughput range from 350 Kbps to 360 Kbps. This is due to the fact that when extensive routing table information exchanges are considered, the bandwidth consumption cannot be further minimized due to the transmission of large amounts of data, which will contribute to an even lower throughput as it was observed. SDN-BGP and SDN-DDPG need to transmit a large amount of data required for routing optimization, which not only produces a higher end-to-end delay but also increases the transmission burden of the network link and reduces the actual throughput. However, FMAAC uses MARL to make routing decisions, and agents in each AS calculate the best routing path across the entire network through the policy network and value network in each federation agent.

In addition to the end-to-end delay and throughput in the above experiment, another significant metric for evaluating the performance of the routing mechanism is the average delivery rate. We measure the average delivery rate of three schemes using different node failure probabilities in this experiment, which is used to express sudden situations in the interdomain routing testbed, such as destination network unreachable or link failure. Under the condition of keeping the same data arrival rate, SDN-BGP, SDN-DDPG, and FMMAC are, respectively, run as the routing mechanism, and the SDN-DDPG and FMMAC running here have also been trained for 50000 times. In Figure 7, the average delivery rate for each scheme is shown. We observe that the average delivery rate of each scheme decreases when the node failure probability increases. When the value of node failure probability is relatively small, the average delivery rate of the three mechanisms is comparatively similar. When the value of node failure probability continues to increase, the average delivery rate of SDN-BGP and SDN-DDPG has dropped below 90%, while the average delivery rate of FMAAC can still be maintained above 90%. FMAAC has the support of interdomain collaborative architecture so that it can better face problems due to sudden situations. Setting such as end-to-end delay, throughput, and average delivery rate as the FMMAC's optimization goals can achieve guaranteed end-to-end QoS.

6. Conclusion

In this paper, we have studied the problem of interdomain routing in decentralized multidomain networks. We propose an interdomain collaborative architecture, and the MARL framework for collaborative routing is installed in the interdomain routing environment. We further defined the mapping optimization problem and designed a corresponding algorithm.

FMAAC is based on multiagent actor-critic and feudal reinforcement learning. Experimental evaluation results have demonstrated that the proposed approach can decrease the complexity compared to the conventional MARL methods and achieve better performance on end-to-end delay, throughput, and average delivery rate than previous routing schemes. As future work, our MARL method will be extended to consider different objectives, such as packet loss and resources utilization, and we will compare our MARL method with more interdomain routing optimization methods. We will further study the feasibility of the proposed model in other areas and improve it to make it scalable to similar optimization problems in other multiagent environments.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key R&D Program of China (Grant No. 2018YFB1800100); the National Natural Science Foundation of China (Grant No. U1909204); the Beijing Natural Science Foundation, China (Grant No. 4202082); and the Open Research Projects of Zhejiang Lab (Grant No. 2021LC0AB03).

References

[1] H. Yang, Y. Liang, J. Yuan, Q. Yao, A. Yu, and J. Zhang, "Distributed blockchain-based trusted multidomain collaboration for mobile edge computing in 5G and beyond," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 11, pp. 7094–7104, 2020.

[2] R. B. Silva and E. S. Mota, "A survey on approaches to reduce BGP interdomain routing convergence delay on the Internet," *IEEE Communication Surveys and Tutorials*, vol. 19, no. 4, pp. 2949–2984, 2017.

[3] J. Zhao, F. Li, D. Ren, J. Hu, Q. Yao, and W. Li, "An intelligent inter-domain routing scheme under the consideration of diffserv QoS and energy saving in multi-domain software-defined flexible optical networks," *Optics Communications*, vol. 366, pp. 229–240, 2016.

[4] L. You, L. Wei, L. Junzhou, J. Jian, and X. Nu, "An inter-domain multi-path flow transfer mechanism based on SDN and multi-domain collaboration," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 758–761, Ottawa, ON, Canada, 2015.

[5] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proceedings of the 15th ACM workshop on hot topics in networks*, pp. 50–56, 2016.

[6] T. P. Lillicrap, J. J. Hunt, A. Pritzel et al., "Continuous control with deep reinforcement learning," 2015, https://arxiv.org/abs/1509.02971.

[7] X. Zhao, C. Wu, and F. Le, "Improving inter-domain routing through multi-agent reinforcement learning," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 1129–1134, Toronto, ON, Canada, 2020.

[8] S. Agarwal, C. N. Chuah, and R. H. Katz, "OPCA: robust interdomain policy routing and traffic control," in *2003 IEEE Conference onOpen Architectures and Network Programming*, pp. 55–64, San Francisco, CA, USA, 2003.

[9] P. Sermpezis and X. Dimitropoulos, "Can SDN accelerate BGP convergence? — a performance analysis of inter-domain routing centralization," in *2017 IFIP Networking Conference (IFIP Networking) and Workshops*, pp. 1–9, Stockholm, Sweden, 2017.

[10] E. A. Alabdulkreem, H. S. Al-Raweshidy, and M. F. Abbod, "MRAI optimization for BGP convergence time reduction without increasing the number of advertisement messages," *Procedia Computer Science*, vol. 62, pp. 419–426, 2015.

[11] Q. Xiang, J. Zhang, K. Gao et al., "Toward optimal software-defined interdomain routing," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pp. 1529–1538, Toronto, ON, Canada, 2020.

[12] A. Arins, "Blockchain based inter-domain latency aware routing proposal in software defined network," in *2018 IEEE 6th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE)*, pp. 1-2, Vilnius, Lithuania, 2018.

[13] Z. Zhong, N. Hua, Z. Yuan, Y. Li, and X. Zheng, *Routing without routing algorithms: an AI-based routing paradigm for multi-domain optical networks*, Optical Fiber Communication Conference (OFC) 2019, OSA Technical Digest (Optical Society of America), 2019, paper Th2A.24.

[14] P. Sun, Y. Hu, J. Lan, and M. Chen, "TIDE: Time-relevant deep reinforcement learning for routing optimization," *Future Generation Computer Systems*, vol. 99, pp. 401–409, 2019.

[15] J. Foerster, N. Nardelli, G. Farquhar et al., "Stabilising experience replay for deep multi-agent reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning*, pp. 1146–1155, 2017.

[16] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *the 31st International Conference on Neural Information Processing Systems*, pp. 6382–6393, 2017.

[17] P. Sunehag, G. Lever, A. Gruslys et al., "Value-decomposition networks for cooperative multi-agent learning," 2017, https://arxiv.org/abs/1706.05296.

[18] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *Proceedings of the 36th International Conference on Machine Learning, PMLR*, vol. 97, pp. 2961–2970, 2019, http://proceedings.mlr.press/v97/iqbal19a.html.

[19] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018, https://ojs.aaai.org/index.php/AAAI/article/view/11794.

[20] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Basar, "Fully decentralized multi-agent reinforcement learning with networked agents," in *Proceedings of the 35th International Conference on Machine Learning, PMLR*, vol. 80, pp. 5872–5881, 2018, http://proceedings.mlr.press/v80/zhang18n.html.

[21] S. Mukhopadhyay and B. Jain, "Multi-agent Markov decision processes with limited agent communication," in *Proceeding of the 2001 IEEE International Symposium on Intelligent Control (ISIC '01) (Cat. No.01CH37206)*, pp. 7–12, Mexico City, Mexico, 2001.

[22] T. G. Dietterich, "Hierarchical reinforcement learning with the MAXQ value function decomposition," *Journal of Artificial Intelligence Research*, vol. 13, pp. 227–303, 2000.

[23] A. S. Vezhnevets, S. Osindero, T. Schaul et al., "Feudal networks for hierarchical reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning, PMLR*, vol. 70, pp. 3540–3549, 2017, http://proceedings.mlr.press/v70/vezhnevets17a.html.

[24] I. Casanueva, P. Budzianowski, P. H. Su et al., "Feudal reinforcement learning for dialogue management in large domains," 2018, https://arxiv.org/abs/1803.03232.

[25] J. Ma and F. Wu, "Feudal multi-agent deep reinforcement learning for traffic signal control," in *the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 816–824, 2020.

[26] C. Xu, S. Liu, C. Zhang, Y. Huang, Z. Lu, and L. Yang, “Multi-agent reinforcement learning based distributed transmission in collaborative cloud-edge systems,” *IEEE Transactions on Vehicular Technology*, vol. 70, no. 2, pp. 1658–1672, 2021.

[27] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, “QTRAN: learning to factorize with transformation for cooperative multi-agent reinforcement learning,” in *Proceedings of the 36th International Conference on Machine Learning, PMLR*, vol. 97, pp. 5887–5896, 2019, https://proceedings.mlr.press/v97/son19a.html.

[28] H. Ryu, H. Shin, and J. Park, “Multi-agent actor-critic with hierarchical graph attention network,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 5, pp. 7236–7243, 2020.

[29] M. A. L. Silva, S. R. de Souza, M. J. F. Souza, and A. L. Bazzan, “A reinforcement learning-based multi-agent framework applied for solving routing and scheduling problems,” *Expert Systems with Applications*, vol. 131, pp. 148–171, 2019.

[30] T. Haarnoja, A. Zhou, K. Hartikainen et al., “Soft actor-critic algorithms and applications,” 2018, https://arxiv.org/abs/1812.05905.

[31] P. Gawłowicz and A. Zubow, “ns-3 meets OpenAI gym: the playground for machine learning in networking research,” in *the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 113–120, 2019.

[32] G. Brockman, V. Cheung, L. Pettersson et al., “OpenAI gym,” 2016, https://arxiv.org/abs/1606.01540.

[33] G. Riley and T. Henderson, *Modeling and Tools for Network Simulation*, Springer, 2010.

[34] V. Le, N. Ioini, H. Barzegar, and C. Pahl, “A multi-domain network simulator based on NS-3,” in *the 10th International Conference on Simulation and Modeling Methodologies, Technologies and Applications*, pp. 217–224, 2020, https://www.scitepress.org/Link.aspx?doi=10.5220/0009831602170224.

[35] A. Alanazi and E. Khaled, “Real-time QoS routing protocols in wireless multimedia sensor networks: study and analysis,” *Sensors*, vol. 15, no. 9, pp. 22209–22233, 2015.

[36] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor,” in *Proceedings of the 35th International Conference on Machine Learning, PMLR*, vol. 80, pp. 1861–1870, 2018, https://proceedings.mlr.press/v80/haarnoja18b.

[37] Z. Chen, J. Bi, Y. Fu, Y. Wang, and A. Xu, “MLV: a multi-dimension routing information exchange mechanism for inter-domain SDN,” in *2015 IEEE 23rd International Conference on Network Protocols (ICNP)*, pp. 438–445, San Francisco, CA, USA, 2015.

[38] V. Kotronis, A. Gamperli, and X. Dimitropoulos, “Routing centralization across domains via SDN: a model and emulation framework for BGP evolution,” *Computer Networks*, vol. 92, pp. 227–239, 2015.