Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 8890772, 17 pages
https://doi.org/10.1155/2021/8890772

Research Article

A Fuzzy-Logic-Based Double *Q*-Learning Routing in Delay-Tolerant Networks

Jiagao Wu, [1,2] **Fan Yuan**, [1,2] **Yahang Guo**, [1,2] **Hongyu Zhou**, [1,2] **and Linfeng Liu** [1,2]

[1] *School of Computer, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu 210023, China*
[2] *Jiangsu Key Laboratory of Big Data Security and Intelligent Processing, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu 210023, China*

Correspondence should be addressed to Jiagao Wu; jgwu@njupt.edu.cn

Received 6 April 2020; Revised 19 December 2020; Accepted 11 January 2021; Published 27 January 2021

Academic Editor: Michael McGuire

Copyright © 2021 Jiagao Wu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Delay-tolerant networks (DTNs) are wireless mobile networks, which suffer from frequent disruption, high latency, and lack of a complete path from source to destination. The intermittent connectivity in DTNs makes it difficult to efficiently deliver messages. Research results have shown that the routing protocol based on reinforcement learning can achieve a reasonable balance between routing performance and cost. However, due to the complexity, dynamics, and uncertainty of the characteristics of nodes in DTNs, providing a reliable multihop routing in DTNs is still a particular challenge. In this paper, we propose a Fuzzy-logic-based Double *Q*-Learning Routing (FDQLR) protocol that can learn the optimal route by combining fuzzy logic with the Double *Q*-Learning algorithm. In this protocol, a fuzzy dynamic reward mechanism is proposed, and it uses fuzzy logic to comprehensively evaluate the characteristics of nodes including node activity, contact interval, and movement speed. Furthermore, a hot zone drop mechanism and a drop mechanism are proposed, which can improve the efficiency of message forwarding and buffer management of the node. The simulation results show that the fuzzy logic can improve the performance of the FDQLR protocol in terms of delivery ratio, delivery delay, and overhead. In particular, compared with other related routing protocols of DTNs, the FDQLR protocol can achieve the highest delivery ratio and the lowest overhead.

1. Introduction

Nowadays, with the improvement of network technologies, the application scenarios are also expanding. Network environments with low delivery ratio, high latency, and limited connectivity are becoming increasingly common, such as interplanetary networks [1], sensor networks [2], and vehicular networks [3]. In these networks, for example, the frequent topology changes caused by the high mobility of nodes make the messages lost or delivered with large delay, and thus, the traditional routing protocols are unreliable or even invalid. In order to solve these problems, delay-tolerant networks (DTNs) [4] are proposed, where messages are forwarded by communication opportunities when nodes meet each other. However, the multihop routing in DTNs is still a considerable challenge, and an efficient routing algorithm adapting to the DTN environment is very necessary.

The typical routing in DTNs is to select the optimal path from the source node to the destination node through a set of intermediate nodes. Store-carry-forward-based routing is one of the most important mechanisms to deal with routing problems in DTNs. This mechanism ensures a message can be forwarded and successively stored on the intermediate nodes until the message reaches the destination node. Based on the store-carry-forward routing, researchers have proposed many routing protocols for DTNs, which can be classified into two types: replication-based routings and forwarding-based routings. However, the replication-based routing protocols (e.g., Epidemic [5] and Spray and Wait [6]) usually cause flooding and exhaust the limited network resources, because they generate excessive message copies in the network. Although the forwarding-based routing protocols (e.g., First Contact [7] and Direct Delivery [8]) can conserve the network resources, they may worsen the delivery ratio and latency of messages. Therefore, routing protocols that can deliver messages with high delivery ratio, short latency, and low overhead are still needed for DTNs.

The PRoPHET protocol [9] is a classic routing protocol, which relies on the history of encounters and transitivity to deliver copies of messages. However, the PRoPHET protocol cannot quickly adapt to node mobility and network topology changes. To solve these problems, routings based on reinforcement learning (RL) algorithm have been proposed. Delay-Tolerant Reinforcement-Based (DTRB) routing protocol [10] adopts the Q-Learning algorithm (Q-Leaning is one of reinforcement learning algorithms) to learn about routes in the network and replicate messages to the node with the best reward. But, the DTBR protocol takes the maximum action value as the optimal action, which may be obscured by overestimation. Therefore, in our previous work, the Double Q-Learning Routing (DQLR) protocol was proposed [11], which adopts the Double Q-Learning algorithm to obtain an unbiased estimation and improve the performance of message delivery. However, in real scenarios of DTNs, the characteristics of nodes (e.g., node activity, contact interval, and movement speed) are complex, dynamic, and uncertain, which will affect the performance of routing protocols. Therefore, Fuzzy Logic (FL) should be introduced into the routing of DTNs to handle the imprecise and uncertain characteristics and select the proper next hop nodes. It can be seen that FL improves the describing ability for these fuzzy characteristics and captures the essential meaning of fuzzy knowledge in DTNs.

In this paper, we propose a Fuzzy-logic-based Double Q -Learning Routing (FDQLR) protocol, as an improvement of DQLR, whose objectives are to increase the delivery ratio and decrease the overhead of the message transmission in DTNs by integrating fuzzy logic and Double Q-Learning jointly. In FDQLR, the characteristics of nodes, including node activity (the activity of a node meeting other nodes), contact interval (the contact interval time of a node encountering other nodes), and movement speed (the movement speed of node), are considered jointly by fuzzy logic. Then, the fuzzy logic results are used in the reward mechanism of the Double Q-Learning routing. To the best of our knowledge, there are few works using fuzzy logic to evaluate the characteristics of nodes in Double Q-Learning routing algorithm of DTNs.

The main contributions of this paper are as follows:

(1) We propose a Fuzzy-logic-based Double Q-Learning Routing (FDQLR) protocol that combines the fuzzy logic with a Double Q-Learning algorithm to improve the decision of the best next hop in the routing of DTNs

(2) A fuzzy dynamic reward mechanism is proposed, which adopts the fuzzy logic to comprehensively evaluate the characteristics such as node activity, contact interval, and movement speed, then these characteristics are converted into a fuzzy reward coefficient in the immediate reward function of the Double Q-Learning routing

(3) A hot zone mechanism is proposed to mark the nodes that meet the destination node within a certain time to improve the message forwarding efficiency. In addition, a drop mechanism is proposed, which drops the delivered messages and the older undelivered messages properly to alleviate the buffer overflows

The rest of this paper is organized as follows: In Section 2, we review the related work briefly. In Section 3 and Section 4, the proposed system model and routing algorithm are described in detail, respectively. In Section 5, the proposed work is evaluated comprehensively, along with extensive results and discussions. Finally, we conclude the paper with some future works in Section 6.

2. Related Work

Delay-tolerant networks (DTNs) are wireless mobile networks, which are characterized by frequent disconnectivity and high latency. DTNs are intended to deal with the scenarios involving intermittent connectivity between adjacent nodes, lack of contemporaneous end-to-end links, and exceptionally high delays and error rates. To solve these problems, several routing protocols have been proposed, such as Epidemic and PRoPHET. The Epidemic routing protocol [5] is one of the flooding routings that replicate messages to neighbor nodes randomly, without using any predictions of the path forwarding probabilities. The Probabilistic Routing Protocol (PRoPHET) [9] relies on the delivery predictability of the current node's neighbors to deliver messages to reliable relay nodes. Thus, the PRoPHET protocol exploits the historical information of the node encounters and message deliveries to determine whether to forward messages. The Predict and Relay (PER) routing protocol [12] adopts a landmark trajectory prediction method that is different from previous prediction-based DTN routing algorithms because it considers when two nodes will encounter each other. However, those protocols mentioned above cannot estimate the movement pattern of nodes very well. Therefore, some routing protocols based on the reinforcement learning (RL) algorithm have been proposed recently, which can indirectly estimate the movement pattern of nodes and constantly learn the knowledge from the network. Reinforcement learning [13] refers to a trial-and-error way of learning and can be expressed as a Markov decision-making process. In the network environment of routing selection based on reinforcement learning, nodes attempt to learn information collected from neighbor nodes and networks. Then, the corresponding rewards can be given, and the action with the highest reward can be regarded as the best policy for the next hop routing. Q-Learning [14] algorithm is well-known as a Temporal Difference (TD) algorithm for reinforcement learning and is wildly applied to dynamic networks. Q-Learning algorithm can obtain the best choice via continuous interactions with the environment rather than the prior knowledge. QLAODV [15] is an enhanced routing protocol for ad hoc networks, which relies on a Q-Learning algorithm to infer the network link states and dynamically change the routes based on the learned information. QLAODV also utilizes a route change request/reply mechanism to check the path availability in a real-time manner. Adaptive Reinforcement-Based Routing (ARBR) [16] uses Collaborative Reinforcement Learning (CRL) as a self-organizing technique, which considers the mobility statistics, congestion, and buffer occupancy as feedback in the quality-metric function. QKS [17] uses the Q-Learning algorithm associated with kinematic and sweeping features to explore the routing in an underwater network environment. Delay-Tolerant Reinforcement-Based (DTRB) [10] is a Q-Learning routing protocol, which utilizes multiagent Q-Learning techniques to learn about routes in the network and forwards or replicates messages that produce the best rewards. In the DTRB framework, rewards are associated with the time of the messages reaching the destination. It is shown that DTRB improves the delivery ratio in dense population areas. However, using Q-Learning in DTNs can also suffer a large penalty, because it can produce a positive bias by using the maximum value as the approximation of the maximum expected value. Therefore, we proposed DQLR [11] to solve the above problems. In the DQLR protocol, the Double Q -Learning algorithm is used to decouple the selection from the evaluation to obtain an unbiased estimation. In addition, the dynamic reward and intermediate value mechanisms are proposed to adapt to the node mobility and the change of network topology, which improve the performance of the routing protocol.

However, in the real DTN environment, the delivery ratio and delivery delay of the message forwarding is directly affected by the characteristics of nodes (e.g., node activity, contact interval, and movement speed), which are not taken into account in the DQLR protocol. Besides, these characteristics are complex, dynamic, and uncertain in DTNs. For example, it is difficult to use clear criteria to indicate the speed of node movement and the time of contact interval in DTNs. Therefore, Fuzzy Logic (FL) is introduced to handle these various characteristics and evaluate the quality of nodes as next hop nodes in DTN routings [18]. FL is a classical method for dealing with uncertain and imprecise information, which is present in real-world problems. Additionally, in DTNs, FL is appropriate due to its ability in "implementing" the approximate reasoning. AFSnW [19] is proposed based on the Spray and Wait protocol, which mainly adopts a fuzzy decision that combines the transmission count and message size, and classifies messages into different prioritization levels in a buffer to improve the delivery ratio. EFSnWR [20] employs a fuzzy prioritization-based message scheduling policy along with a random drop policy, which optimizes the mechanism of message priority in AFSnW by aggregating three parameters, i.e., the number of message replicas, message size, and remaining Time-To-Live (TTL). AFRON [21] is also a fuzzy routing protocol for opportunistic networks, which takes the parameters (transmission count, message size, and remaining TTL) as fuzzy input parameters to discover a path to the destination. In [22], a Fuzzy-assisted Position-Based Routing protocol with DTN capability (FPBR-DTN) is proposed for Vehicular Ad hoc NETworks (VANETs). FPBR-DTN is a hybrid routing protocol and has three main modes including greedy, perimeter, and DTN. FPBR-DTN improves the greedy routing by applying the fuzzy logic and parameters such as the number of neighbors, neighboring vehicles' speed, direction, and distance from a destination; a forwarding chance value is calculated for each neighbor node. Then, the node having the highest chance value among the neighbors is selected for greedy forwarding. A fuzzy constraint Q-Learning routing protocol for vehicular networks was proposed in [23], which is the improvement of QLAODV. It employs fuzzy logic to evaluate the link statuses by bandwidth, mobility, and signal strength, and a Q-Learning based approach is provided to select a stable and efficient routing. However, it is based on the Ad hoc On-demand Distance Vector (AODV) routing protocol, which is designed for the mobile ad hoc network rather than DTNs. FQLRP [24] is a novel routing protocol for opportunistic networks, which uses fuzzy-based Q-Learning for efficient routing. FQLRP predicts the next optimal forwarder of a message based on a reward mechanism that considers the node's energy, movement, and buffer space as parameters.

A summary of the related routing protocols discussed in this section is presented in Table 1, where these protocols are compared in terms of routing history, learning method, routing or reward mechanism, fuzzy logic, fuzzy input parameter, drop policy, and network type. We can find that few routing protocols are utilizing both fuzzy logic and Double Q -Learning in DTNs or taking into account the impacts of practical network environment and node characteristics on routing decisions. Therefore, based on the DQLR protocol, we present an enhanced routing protocol called FDQLR, which uses a Double Q-Learning algorithm to select the next hop nodes with an unbiased estimation, and utilizes fuzzy logic to comprehensively evaluate the characteristics such as node activity, contact interval, and movement speed, which directly affect the performance of message delivery. Also, we propose a hot zone mechanism, which consists of the neighbor nodes that meet the destination node within a certain time to improve the message forwarding efficiency. Moreover, we present a drop mechanism, which controls the number of messages carried by nodes and discards the messages that have not been forwarded for a long time.

3. System Model

Reinforcement learning refers to a type of learning that is achieved through interaction. In RL, the learner and decision maker are called an agent, and the surrounding with which they interact is called the environment. The agent selects the actions, and the environment, in return, provides rewards and a new state. The reinforcement learning algorithm is aimed at finding a policy, which means a mapping from state to action, which maximizes the expected cumulative reward (value function) under that policy. To describe our model and algorithm clearly, we provide a list of notations in Table 2.

Based on reinforcement learning, FDQLR can be modeled as follows: DTNs can be regarded as environment E, which includes mobile nodes, messages delivered by nodes, and transmissions among nodes. Each message delivered

Table 1: Comparison of related routing protocols.

Name of protocol	Routing history	Learning method	Routing/reward mechanism	Fuzzy logic	Fuzzy input parameter	Drop policy	Network type
Epidemic [5]	—	—	—	—	—	—	DTNs
PRoPHET [9]	Yes	—	Contact probability	—	—	—	DTNs
PER [12]	Yes	—	Contact time	—	—	—	DTNs
QLAODV [15]	Yes	Q	Link status	—	—	—	Ad hoc
ARBR [16]	Yes	Q	Mobility statistics; congestion; buffer occupancy	—	—	—	DTNs
QKS [17]	Yes	Q	Kinematic and sweeping features	—		—	Ad hoc
DTRB [10]	Yes	Q	Time to destination	—	—	—	DTNs
DQLR [11]	Yes	Double Q	Dynamic reward; intermediate value	—	—	—	DTNs
AFSnW [19]	Yes	—	Prioritization of message	Yes	Forward transmission count; message size	Yes	DTNs
EFSnWR [20]	Yes	—	Prioritization of message	Yes	Number of replicas; message size; remaining TTL	Yes	DTNs
AFRON [21]	Yes	—	Prioritization of message	Yes	Forward transmission count; message size; remaining TTL	—	DTNs
FPBR-DTN [22]	Yes	—	Forwarding chance	Yes	Number of neighbors; speed; direction; distance	—	Ad hoc/DTNs
FQLAODV [23]	Yes	Q	Link status	Yes	Bandwidth; mobility; signal strength	—	Ad hoc
FQLRP [24]	Yes	Q	Reward	Yes	Energy; movement; buffer	—	DTNs
FDQLR	Yes	Double Q	Dynamic reward; intermediate value; hot zone	Yes	Node activity; contact interval; movement speed	Yes	DTNs

from source node to destination node denotes an agent. All nodes in the network can be seen as the set of states S. A node represents the state s ($s \in S$) when the message is delivered to the node. All the processes of the next hop (state) selection for transmitting messages can be taken as a set of actions denoted by A. Each node is equipped with an omnidirectional antenna with a fixed transmission range. Two neighbors can communicate with each other only if they are within the transmission range of each other. We define all neighbors of the current node carrying messages as a state set, and the process of the next hop (state) selection from all neighbors can be regarded as an action, a ($a \in A$). Let d be the destination node of a message; we define $Q_s(d, a)(s, d \in S;\ a \in A)$ as the estimation of the future rewards if a message in node (state) s takes an action a to the destination node d.

The learning task must be done in a distributed way for each node since a global view of the network state is impossible. Thus, the state transition policy of the next hop nodes is essential for delivering messages from the current node c to the destination node d. Suppose all messages have definite destinations and the selection of the next hop to the destinations can influence the network environment. Once the optimal next hop for messages is determined, the action of the current node to next hop i will receive the immediate reward $\tilde{R}_c(d, i)$. According to the information of different next hops, the corresponding rewards will be given, and the best choice can be made as the action with the highest future rewards. By the Double Q-Learning algorithm, we define $Q_c^A(d, i)$ and $Q_c^B(d, i)$ as the two values of future rewards that current node c bound to destination node d through the next hop i ($i \in D_c$), where D_c is the set of neighbors of the current node. Besides, it should be noted that the action of the next hop selection from current node c to next hop i is denoted by i in $Q_c^A(d, i$) and $Q_c^B(d, i)$ for simplicity. Therefore, the selection of the next hop in a greedy strategy can be expressed by Equations (1) and (3), and the update rule in $Q_c^A(d, i)$ and $Q_c^B(d, i)$ can be written as Equations (2) and (4).

$$y^* = \arg \max_{j \in D_i} Q_i^A(d, j), \tag{1}$$

$$Q_c^A(d, i) \leftarrow (1 - \delta) Q_c^A(d, i) + \delta \left[\tilde{R}_c(d, i) + \eta Q_i^B(d, y^*)\right], \tag{2}$$

$$z^* = \arg \max_{j \in D_i} Q_i^B(d, j), \tag{3}$$

$$Q_c^B(d, i) \leftarrow (1 - \delta) Q_c^B(d, i) + \delta \left[\tilde{R}_c(d, i) + \eta Q_i^A(d, z^*)\right], \tag{4}$$

Table 2: List of notations.

Notation	Description	Notation	Description
E	DTN environment	n_i	Number of the current node encounters nondestination nodes
S	Set of states	n_d	Number of the current node that encounters the destination node
s	A state of S	ε	Encounter weight
A	Set of actions	n	Total number of the current node that encounters other nodes in the network
a	An action of A	$\Delta t_{c_{now}}$	Average contact interval between the current node and other nodes
d	Destination node	$\Delta t_{c_{\min}}$	The minimum contact interval between the current node and other nodes in the network
$Q_s(d, a)$	Future rewards of s taking an action a to the destination node d	$\Delta t_{c_{\max}}$	The maximum contact interval between the current node and other nodes in the network
c	Current node	$s_{c_{\text{now}}}$	Speed of the current node at the current time
$\tilde{R}_c(d, i)$	Reward function of current node c taking action of selecting i as the next hop to the destination node d	$s_{c_{\min}}$	The lowest speed of the current node in the network
$Q_c^A(d, i)$	Value A of the future rewards	$s_{c_{\max}}$	The highest speed of the current node in the network
$Q_c^B(d, i)$	Value B of the future rewards	γ_1	Aging constant of Q value
D_c	Set of neighbors of node c	μ_1	Number of time units having elapsed
D_d	Set of neighbors of destination node	α	Balance factor
i	Neighbor of node c	β	Scaling constant
j	Neighbor of node i	$\widehat{Q}_c(d, i)$	Mixed future reward
$y*$	Neighbor of node i with the maximum value of $Q_i^A(d, j)$	T_0	Time period of hot zone
$z*$	Neighbor of node i with the maximum value of $Q_i^B(d, j)$	t_i	The contact interval between the node i and the destination node in the hot zone
η	Discount factor	$H_c(t_i)$	Hot zone strength of node i in the hot zone
δ	Learning rate	γ_2	Aging constant of hot zone
h	Number of hops from the source node to the destination node	μ_2	Number of time units having elapsed of hot zone
$F_c(d)$	Fuzzy reward coefficient of current node c relative to destination node d	ω	Joint coefficient
σ	Fuzzy weight	$U_c(d, i, t_i)$	Joint utility function

where η is the discount factor, δ is the learning rate, and D_i is the set of neighbors of node i.

The next hop y^* and z^* in Equations (1) and (3) denote the neighbors of node i with the maximum value of $Q_i^A(d, j)$ and $Q_i^B(d, j)$, respectively. Then, we update $Q_c^A(d, i)$ with $Q_i^B(d, y^*)$ and update $Q_c^B(d, i)$ with $Q_i^A(d, z^*)$ by Equations (2) and (4).

The discount factor η should satisfy that $0 < \eta \le 1$, and it determines the degree of the importance of the future rewards. If η is too low, the immediate rewards are dominated, and if η is too high, the future rewards will be considered with a greater weight.

The learning rate δ ($0 < \delta \le 1$) limits the learning speed. In FDQLR, it affects how quickly the future rewards can change with a network topology change. A low learning rate may cause the routing algorithm unable to adapt to the network mobility, whereas a high learning rate may lead nodes to receive incorrect rewards and reflect the wrong network mobility.

$\tilde{R}_c(d, i)$ represents the immediate reward function that is nonnegative and will have a positive value if c is one of the neighbors of the destination node. To comprehensively consider the characteristics of nodes in DTNs, fuzzy logic is used in the definition of $\tilde{R}_c(d, i)$, which makes the routing algorithm fully consider the status of the network and nodes when evaluating the next hop of messages. In addition, a dynamic reward mechanism is employed for $\tilde{R}_c(d, i)$, which is subjected to exponential decay and related to h, i.e., the number of hops from the source node to the destination node. That is, a larger number of hops give rise to a smaller reward. Hence, $\tilde{R}_c(d, i)$ can be defined as

$$\tilde{R}_c(d, i) = \begin{cases} [(1-\sigma) + \sigma F_c(d)]e^{-h}, & \text{if } c \in D_d, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

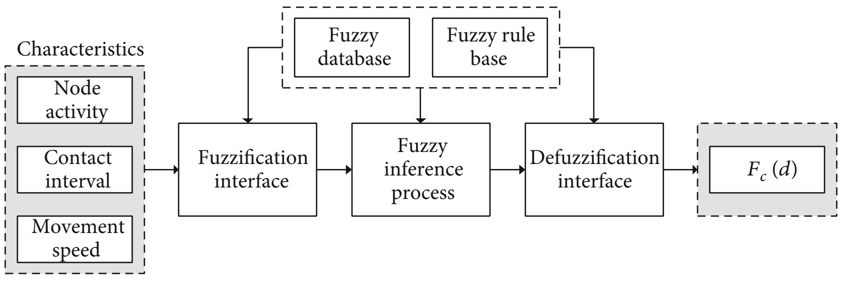


Figure 1: Fuzzy system for generating $F_c(d)$.

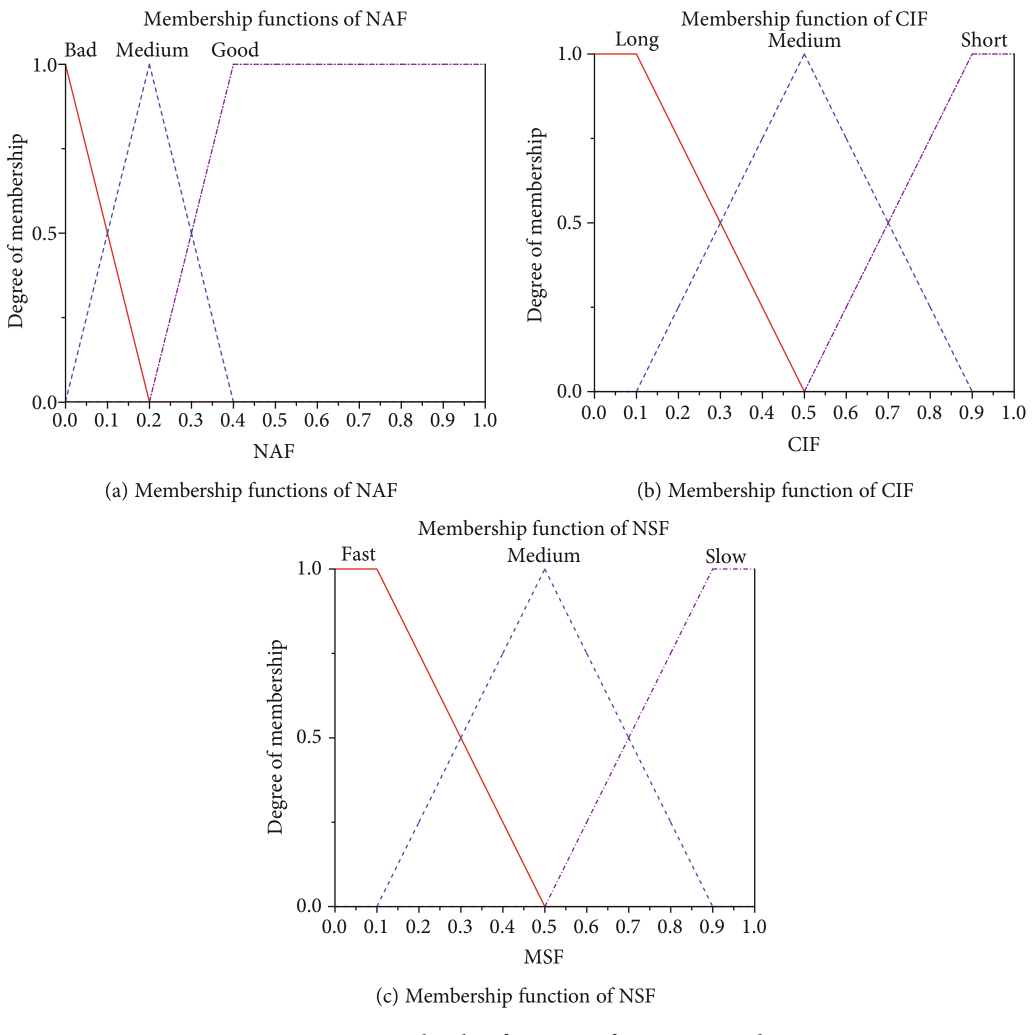


Figure 2: Fuzzy membership functions of NAF, CIF, and MSF.

where D_d is the set of neighbors of the destination node d; $F_c(d)$ represents the fuzzy reward coefficient of the current node c relative to the destination node d, which will be described in the next section, and $\sigma\ (0 \leq \sigma \leq 1)$ is the fuzzy weight, which determines the degree of relationship between $\tilde{R}_c(d, i)$ and $F_c(d)$.

4. Fuzzy Double Q-Learning Routing

In this section, we discuss the proposed routing protocol FDQLR, which considers the characteristics of nodes with fuzzy logic. The rest of the section consists of four parts. We first discuss the fuzzy dynamic reward mechanism. Then, we present the value aging and intermediate value mechanism. Besides, we propose a hot zone and drop mechanisms. Finally, we describe the proposed routing algorithm.

4.1. Fuzzy Dynamic Reward Mechanism. In DTNs, routing is a particular challenge because of the complex, dynamic, and uncertain node characteristics such as node activity, contact interval, and movement speed. Therefore, fuzzy logic is adopted to improve the describing ability for these fuzzy

Table 3: Fuzzy rule base.

Rule	NAF	CIF	MSF	Rank
Rule 1	Bad	Long	Fast	Little acceptable (LA)
Rule 2	Bad	Long	Medium	Little good (LG)
Rule 3	Bad	Long	Slow	Bad (B)
Rule 4	Bad	Medium	Fast	Good (G)
Rule 5	Bad	Medium	Medium	Little perfect (LP)
Rule 6	Bad	Medium	Slow	Little good (LG)
Rule 7	Bad	Short	Fast	Acceptable (A)
Rule 8	Bad	Short	Medium	Good (G)
Rule 9	Bad	Short	Slow	Little acceptable (LA)
Rule 10	Medium	Long	Fast	Acceptable (A)
Rule 11	Medium	Long	Medium	Good (G)
Rule 12	Medium	Long	Slow	Little acceptable (LA)
Rule 13	Medium	Medium	Fast	Very good (VG)
Rule 14	Medium	Medium	Medium	Perfect (P)
Rule 15	Medium	Medium	Slow	Good (G)
Rule 16	Medium	Short	Fast	Very acceptable (VA)
Rule 17	Medium	Short	Medium	Very good (VG)
Rule 18	Medium	Short	Slow	Acceptable (A)
Rule 19	Good	Long	Fast	Good (G)
Rule 20	Good	Long	Medium	Little perfect (LP)
Rule 21	Good	Long	Slow	Little good (LG)
Rule 22	Good	Medium	Fast	Perfect (P)
Rule 23	Good	Medium	Medium	Very perfect (VP)
Rule 24	Good	Medium	Slow	Little perfect (LP)
Rule 25	Good	Short	Fast	Very good (VG)
Rule 26	Good	Short	Medium	Perfect (P)
Rule 27	Good	Short	Slow	Good (G)

characteristics. FDQLR uses a fuzzy dynamic reward mechanism in routing decisions. Here, we define the fuzzy reward coefficient $F_c(d)$ in Equation (5) by fuzzy logic.

Figure 1 shows the fuzzy system for generating $F_c(d)$. Firstly, the crisp values of characteristics (i.e., node activity, contact interval, and movement speed) are input into the fuzzification interface and converted into fuzzy values by the membership functions defined in the fuzzy database. Then, the fuzzy inference process is performed according to the fuzzy rule base. Finally, the fuzzy inference output is converted into the crisp value of $F_c(d)$ by the defuzzification interface.

4.1.1. Definitions of Multiple Factors. In this paper, we define the Node Activity Factor (NAF), which represents the degree of activity of the current node in the network. The NAF can be described by the nondestination and destination nodes encountered by the current node, and different weights are assigned to the nondestination nodes and destination nodes. The node will be given higher weights when it encounters the destination node. Hence, the NAF is defined as

$$\mathrm{NAF} = \varepsilon \frac{n_i}{n} + (1-\varepsilon)\frac{n_d}{n}, \tag{6}$$

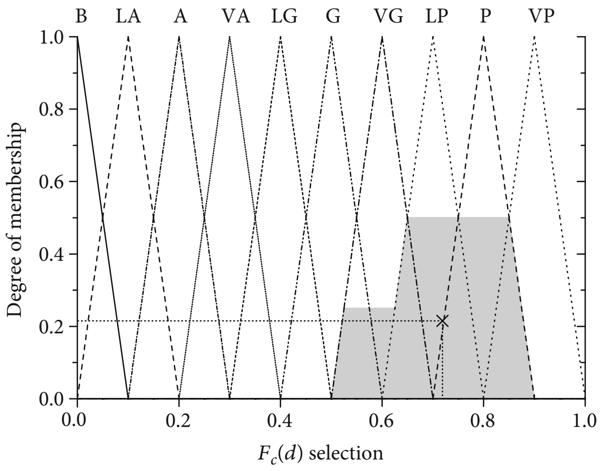


Figure 3: Membership function for output $F_c(d)$.

where n_i is the number of nondestination nodes encountered by the current node, n_d is the number of current nodes encountered by the destination node, and n is the total number of the current nodes encountered by other nodes in the network. ε $(0 \le \varepsilon \le 1)$ is the encounter weight, which indicates the difference of nondestination nodes and destination nodes.

Contact Interval Factor (CIF) represents the normalized factor of the contact interval between the current node and other nodes, which is used to find the most appropriate time interval required for transmitting messages between nodes. CIF is defined as

$$\mathrm{CIF} = \frac{\Delta t_{c_{\mathrm{now}}} - \Delta t_{c_{\mathrm{min}}}}{\Delta t_{c_{\mathrm{max}}} - \Delta t_{c_{\mathrm{min}}}}, \tag{7}$$

where $\Delta t_{c_{\mathrm{now}}}$ denotes the average contact interval between the current node and other nodes and $\Delta t_{c_{\mathrm{min}}}$ and $\Delta t_{c_{\mathrm{max}}}$ represent the minimum and maximum contact intervals between the current node and other nodes in the network, respectively.

Movement Speed Factor (MSF) represents the normalized factor of the relative speed of the current nodes in the network. The MSF also plays an important role in the message transmission process. If the factor is too large, i.e., the relative speed of the node is large, then the message transmission process will be hard to succeed and some messages will be lost. MSF is defined as

$$\mathrm{MSF} = \frac{s_{c_{\mathrm{now}}} - s_{c_{\mathrm{min}}}}{s_{c_{\mathrm{max}}} - s_{c_{\mathrm{min}}}}, \tag{8}$$

where $s_{c_{\mathrm{now}}}$ represents the speed of the current node at the current time and $s_{c_{\mathrm{min}}}$ and $s_{c_{\mathrm{max}}}$ represent the lowest and highest speeds of the current node in the network, respectively.

4.1.2. Fuzzification. Fuzzification is the process of converting crisp values of characteristics to fuzzy values by the fuzzy database. Linguistic variables and membership functions are needed to be defined before the fuzzification. In this

```
Input:
Destination node, $d$
Current node, $c$
The set of neighbors of $c$, $D_c$
One neighbor of node $c$, $i \in D_c$
The set of neighbors of $i$, $D_i$
One neighbor of $i$, $j \in D_i$
Output:
Optimal next hop, $i^T$
1: for all $i \in D_c$ do
2:    Compute multiple factors of NAF, CIF, and MSF by Equations (6)–(8)
3:    Compute $F_c(d)$ by fuzzy set and rules
4:    Compute $\tilde{R}_c(d, i)$ by Equation (5)
5:    Randomly update $Q_c^A(d, i)$ or $Q_c^B(d, i)$ by Equation (2) or Equation (4), respectively
6:    Compute $\widehat{Q}_c(d, i)$ by Equation (11)
7:    if $i$ is $d$ then
8:       $i^T \leftarrow i$
9:       return $i^T$
10:   else
11:   for all $j \in D_i$ do
12:      Use the intermediate value mechanism to compute $\widehat{Q}_c(d, j)$ by Equation (12)
13:      Update $\widehat{Q}_c(d, i)$ by Equation (11)
14:      Compute $H_c(t_i)$ by Equation (13)
15:      Compute $U_c(d, i, t_i)$ by Equation (15)
16:   end for
17:   end if
18: end for
19: $i^T \leftarrow \arg \max_{(c, i \in D_c)} \{U_c(d, c, t_i), U_c(d, i, t_i)\}$
20: return $i^T$
```

Algorithm 1: FDQLR.

paper, three input linguistic variables of NAF, CIF, and MSF are defined, and each input linguistic variable consists of three linguistic terms, which are represented by the membership functions. The results of the membership functions show the degree that linguistic variables belong to their corresponding linguistic terms. Therefore, the fuzzy membership functions of linguistic variables of NAF, CIF, and MSF are defined in Figure 2, where the fuzzy linguistic terms of NAF are {Bad, Medium, Good}, the fuzzy linguistic terms of CIF are {Long, Medium, Short}, and the fuzzy linguistic terms of MSF are {Fast, Medium, Slow}.

4.1.3. Fuzzy Rule Base and Fuzzy Inference. After defining the input linguistic variables of NAF, CIF, and MSF, we define an output linguistic variable called node Rank to represent the quality of a node serving as the next hop, which in fuzzy linguistic terms consists of 10 ranks, i.e., {Bad (B), Little Acceptable (LA), Acceptable (A), Very Acceptable (VA), Little Good (LG), Good (G), Very Good (VG), Little Perfect (LP), Perfect (P), and Very Perfect (VP)}. Then, the fuzzy rule base is defined, which contains fuzzy IF/THEN rules for the inference of a node rank. The rule is "IF (antecedent) part of the rule and then evaluating the THEN (consequent) part of the rule." Table 3 shows the fuzzy rule base for fuzzy linguistic variables. Since there are three input linguistic variables in the fuzzy system, and each of them consists of three linguistic terms, we can obtain 27 (i.e., 3^3) rules for reasoning the node Rank. For example, Rule 3 means "IF the NAF is Bad, CIF is Long, and MSF is Slow, THEN the Rank is Bad (B)," and Rule 23 means "IF the NAF is Good, CIF is Medium, and MSF is Medium, THEN the Rank is Very Perfect (VP)." Rule 3 and Rule 23 express the two extreme cases in the inference of a node rank. It should be pointed out that the fuzzy rule base in Table 3 is determined by our practical experience and intuition in DTN routing. It does not exclude the possibility of other definitions of rules, and the optimal fuzzy rule base will be left as an open problem for future work.

In the process of fuzzy reasoning, usually, multiple rules are satisfied in parallel and the antecedent of a rule has more than one fuzzy linguistic variables. Therefore, some inference methods should be applied to these rules, such as Min–Max, Prod-Max, and Prod-Sum. Without loss of generality, we adopt the Min–Max method because it is simple and widely used [25]. In this method, for each rule, the minimum value of the antecedent is used as the level of rank. When combining different rules, the maximum value of the antecedent is used as the final rank, for example, when NAF, CIF, and MSF belong to the corresponding linguistic variable with terms {Bad: 0, Medium: 0, Good: 1}, {Long: 0.25, Medium: 0.75, Short: 0}, and {Fast: 0, Medium: 0.5, Slow: 0.5}, respectively. In this case, multiple rules are matched including Rule 20, Rule 21, Rule 23, and Rule 24. For Rule 20, the degree of

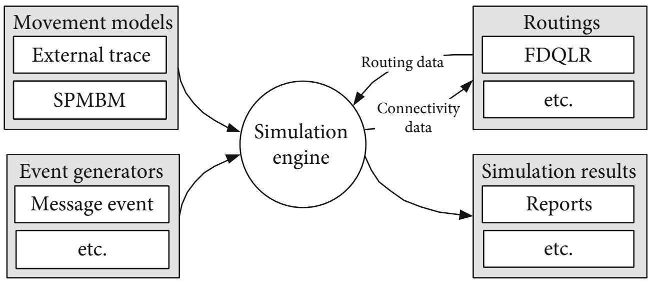


Figure 4: The ONE simulation environment for FDQLR.

NAF for Good is 1, the degree of CIF for Long is 0.25, and the degree of MSF for Medium is 0.5. By the Min–Max method, we take the minimal value and the degree of Rank for Little Perfect is 0.25. Similarly, the degree of Little Good (LG) in Rule 21 is 0.25, the degree of Very Perfect (VP) in Rule 23 is 0.5, and the degree of Little Perfect (LP) in Rule 24 is 0.5. It is easy to find that both Rule 20 and Rule 24 lead to the Rank of Little Perfect (LP), according to the Min–Max method. The maximal value of their consequents is taken, and the degree of Rank for Little Perfect (LP) is 0.5. In this way, all rules are combined, and the fuzzy results are obtained.

4.1.4. Defuzzification. The process of converting the fuzzy inference results into a crisp value based on the output membership function and the corresponding membership degrees is called defuzzification. In this paper, the output membership function of Rank is defined as in Figure 3. There are several common defuzzification methods including Center of Gravity (COG), Bisector of Area (BOA), Mean of Maximum (MOM), and so on. Here, we use the COG method, which returns the center of the area under the curve and is one of the best methods in defuzzification. For example, if the degrees of Rank for Very Good (VG), Little Perfect (LP), and Perfect (P) are 0.25, 0.5, and 0.5, respectively, a shadow shape can be formed according to the output membership function, as shown in Figure 3. Then, we calculate the centroid of this shape and define the x-coordinate of this centroid as the fuzzy reward coefficient $F_c(d)$.

4.2. Value Aging and Intermediate Value Mechanisms. In this paper, the value aging and intermediate value mechanism is based on our previous work [11], which are denoted by $Q_c^A(d, i)$ and $Q_c^B(d, i)$:

$$Q_c^A(d, i) \leftarrow Q_c^A(d, i)_{\text{old}} {\gamma_1}^{\mu_1}, \tag{9}$$

$$Q_c^B(d, i) \leftarrow Q_c^B(d, i)_{\text{old}} {\gamma_1}^{\mu_1}, \tag{10}$$

where $\gamma_1 (0 \le \gamma_1 < 1)$ is the aging constant and μ_1 is the number of time units that have elapsed since the last time the metric was aged.

In the Double Q-Learning algorithm, the two value functions of the future rewards are learned from the separate experience sets, while the action of selecting the next hop should be based on one value. Therefore, we assign a balance factor to the two value functions and define a mixed future reward as $\widehat{Q}_c(d, i)$. We calculate the mixed future rewards for each action and perform the greedy exploration with it. The update of $\widehat{Q}_c(d, i)$ is given as

$$\widehat{Q}_c(d, i) \leftarrow \alpha Q_c^A(d, i) + (1 - \alpha) Q_c^B(d, i), \tag{11}$$

where $\alpha (0 \le \alpha \le 1)$ is a balance factor.

Since the future rewards of encounter are transitive, the intermediate value mechanism is used to calculate the future rewards of the action of delivering a message through an intermediate node, even though the current node and the next hop nodes have not encountered yet. This means that when the current node c encounters node i, and node i will encounter node j $(j \in D_i)$ later, the future rewards of the action from node c to j can be calculated by node i. When node c encounters j in the future, the actions of two nodes already have some future rewards compared to other actions. The update of $\widehat{Q}_c(d, j)$ is given as

$$\widehat{Q}_c(d, j) \leftarrow \widehat{Q}_c(d, j) + \beta \left[\widehat{Q}_c(d, i) + \widehat{Q}_i(d, j) \right], \tag{12}$$

where $\beta \, (0 \le \beta \le 1)$ is a scaling constant and indicates the impact of the transitivity on the FDQLR protocol.

4.3. Hot Zone and Drop Mechanisms. In the FDQLR protocol, we store the information of nodes that the current node has met within a time period T_0 into an encounter table, which includes the node ID and the last encounter interval. When the current node carrying messages meets other nodes, they exchange and update the encounter tables firstly. If the current node finds that its neighbor has met the destination node of the message within T_0, then the neighbor is called a node in the hot zone. The hot zone strength of a node is related to the contact interval between the node and the destination node. The definition of the hot zone strength of node i is shown as

$$H_c(t_i) = \begin{cases} e^{T_0/t_i}, & \text{if } i \in \text{hot zone}, \\ 0, & \text{otherwise}, \end{cases} \tag{13}$$

where $t_i (0 \le t_i \le T_0)$ represents the contact interval between node i and the destination node in the hot zone.

The values of $H_c(t_i)$ should also age over time since node i met the destination node the last time; the aging of $H_c(t_i)$ can be shown as

$$H_c(t_i) \leftarrow H_c(t_i)_{\text{old}} {\gamma_2}^{\mu_2}, \quad (14)$$

where $\gamma_2 (0 \le \gamma_2 < 1)$ is the aging constant of hot zone and μ_2 is the number of elapsed time units since last updated.

In the FDQLR protocol, we consider both the future reward and the hot zone strength when selecting the next hop node and define $U_c(d, i, t_i)$ as the joint utility function whose update rule is shown as

$$U_c(d, i, t_i) \leftarrow \omega \widehat{Q}_c(d, i) + (1-\omega) H_c(t_i), \quad (15)$$

where $\omega (0 \le \omega \le 1)$ represents the joint coefficient.

In addition, the node buffer is an important kind of resource and basis for improving the delivery ratio of routing in DTNs. Therefore, it is necessary to use the drop mechanism to manage the node buffer reasonably.

In this paper, we notice that when the node buffer is full, the message copies that have been sent to the destination node or waited for a long time can be considered invalid message copies. These invalid message copies will waste network resources of DTNs; therefore, they should be dropped. Here, a drop mechanism is proposed to control the number of message copies in the network without reducing the delivery ratio. The drop mechanism is simple and only has the following two rules:

(1) Regardless of whether the node buffer is full or not, each node drops the copies of the message that has been delivered to the destination node. This ensures that the node can have more buffer size to receive new copies of messages

(2) When the node buffer is full and a copy of each message is not delivered to the destination node, the message copy with a longer carrying time should be preferentially dropped, but the message of the source node should be kept in the network

4.4. Routing Algorithm. Based on the mechanisms proposed above, the pseudocode of FDQLR is presented in Algorithm 1.

According to Algorithm 1, the FDQLR protocol uses a fuzzy logic-based approach to evaluate the node status. Each node will periodically update the node information and encounter table. Then, the FDQLR protocol calculates the status of the node according to characteristics of NAF, CIF, and MSF, and uses fuzzy logic to consider these different indicators comprehensively to update the fuzzy reward coefficient $F_c(d)$. In order to improve the adaptability of the fuzzy system, the parameters used in fuzzy logic (e.g., membership functions and fuzzy rules) are stored in external files. Currently, these parameters are configured according to the proposed mechanism aforementioned. In the future, we can also adjust them by experiments to make the system adapt to different network environments. Moreover, we use the Double Q-Learning algorithm to update the two Q values of the future reward values and the intermediate value mechanism to update the Q values of the potential relay nodes. In addition, the hot zone mechanism is proposed to give the relay node in the hot zone a higher priority. Besides, nodes will drop unnecessary message copies according to the drop mechanism. Therefore, the FDQLR protocol can effectively find the best next hop node and improve the efficiency of message forwarding.

Table 4: Simulation parameters.

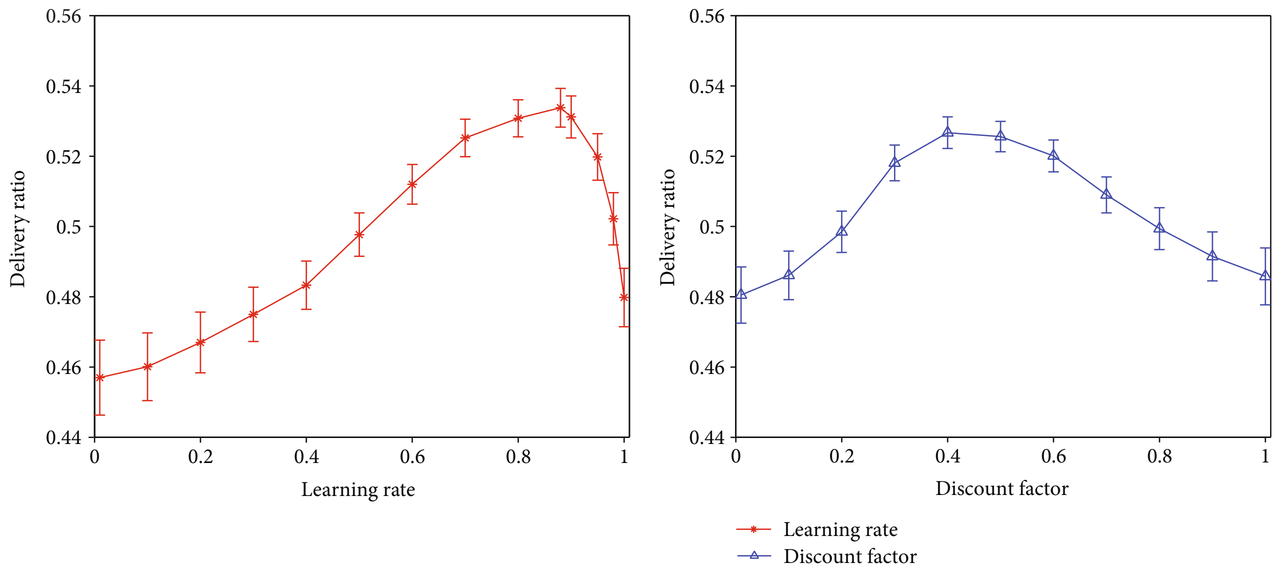
Parameters	RioBuses	SPMBM
Simulate time	477,000 s	43,200 s
Terrain size	35 km × 25 km	4.5 km × 3.2 km
Transmission range	50 m	10 m
Transmission speed	250 kbytes	25 kbytes
Message size	500 kbytes, 1 M	1 kbytes
Number of nodes	55	80

Next, we analyze the computational complexity of Algorithm 1. Since the algorithm adopts the distributed learning scheme, for forwarding a single message, the current node c needs to maintain the set of its neighbors (i.e., D_c). Hence, the outer loop statement of Algorithm 1 (Line 1) will iterate at most $|D_c|$ times. Furthermore, node c also needs to maintain the neighbor set of its neighbor node i (i.e., D_i) for the intermediate value mechanism, and thus, the inner loop statement of Algorithm 1 (Line 11) will iterate $|D_i|$ times. Usually, the nested loop in the algorithm will iterate $|D_c| \cdot |D_i|$ times. However, in the worst case, all n nodes in the network can become neighbors, that is, $|D_c| = |D_i| = n$. Therefore, the computational time and space complexity of the algorithm are both $O(n^2)$. By comparing with other algorithms, we can find that FDQLR has the same computational time and space complexity $O(n^2)$ as DQLR, which adopts the intermediate value mechanism as well. Although the algorithm of PRoPHET only maintains a table of delivery predictability for each node and has the computational time and space complexity of $O(n)$, the performance evaluation in Section 5 shows that PRoPHET cannot be as efficient as the proposed algorithm.

5. Performance Evaluation

5.1. Experiment Setup. In this paper, we use the Opportunistic Network Environment (ONE) [26] network simulator to simulate the FDQLR protocol. The ONE simulator is a Java-based tool offering a broad set of DTN protocol simulation capabilities in a single extensible framework, which mainly consists of movement models, event generators, routings, and simulation results, etc.

Figure 4 shows the ONE simulation environment for FDQLR. Two movement models are used in our simulation experiments. The first movement model is the external trace model, which is derived from the RioBuses dataset on CRAWDAD and contains the real mobile trajectories covering 1,200 square kilometers and involving 17,723 buses in Rio de Janeiro, Brazil. The second movement model is the Shortest Path Map-Based Movement (SPMBM) model, where the nodes move on the shortest path decided by the Dijkstra algorithm in the simulation map. Additionally, the



(a) Delivery ratio vs. learning rate

(b) Delivery ratio vs. discount factor

Figure 5: Performance of delivery ratio against learning rate and discount factor.

message event generator is used to create messages to be delivered randomly. Then, both the mobility traces and message events are fed to the simulation engine to produce the connectivity data that drives the routing modules including FDQLR and other routings. Finally, by collecting the feedback routing data, the simulation engine can obtain the simulation results and generate reports, which contain the message stats (e.g., delivery ratio, delivery delay, overhead, and hop count) for different scenarios of the performance evaluation. In this paper, FDQLR is implemented as a class by extending the ActiveRouter class in the ONE. The node characteristics, the set of neighbors, the double Q values, and related parameters are all maintained as members of the FDQLR class. And the fuzzy system of FDQLR is as defined as a separate class called Fuzzy Attribute, where all components of the fuzzy logic are implemented. Based on this, all functions of the FDQLR algorithm are complete (e.g., the fuzzy inference, double Q value update, hot zone, and intermediate value mechanisms) to select the optimal node of the next hop.

We run the ONE simulation of performance evaluation on a computer with a Windows operation system, 3.4 GHz CPU main frequency, and 6.0 GB memory. In the simulated network environment, we employ the RioBuses dataset as the main movement model and concentrate on a 12-hour period from 7 am to 7 pm and 55 buses as nodes in the area of 35 km × 25 km of the dataset. Furthermore, we also adopt the SPMBM model as a different network scenario for comparison. In running the simulation, when any two nodes meet each other, i.e., move within the transmission range, they exchange and update the node states and information firstly. If any of them carries a message to be forwarded, the corresponding routing protocol (e.g., FDQLR or other related protocols) is invoked to determine the next hop of the message. The process of message forwarding will continue until the message reaches the destination node. We let the simulator run long enough that at least 500 messages were generated and delivered. Then, at the end of the simulation time, we calculate the mean and 95% confidence interval for the performance metrics reported by the simulator. Considering the real network scenario, the settings of simulation parameters of the two movement models are shown in Table 4.

The following metrics are used in our simulations:

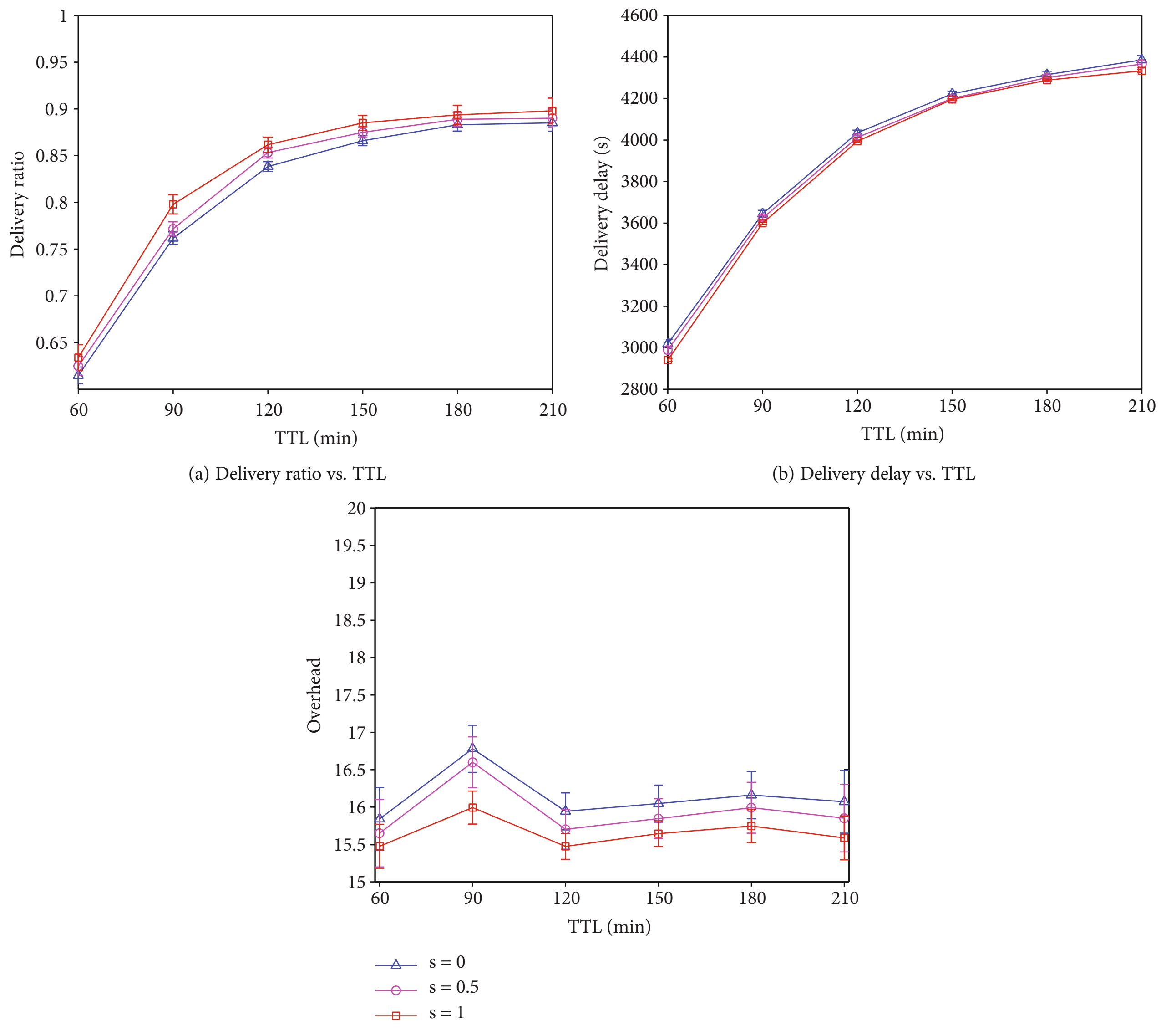
(i) *Delivery ratio*: the ratio of the number of messages that reached the destination and the number of messages generated in the system

(ii) *Delivery delay*: the average time a message takes to reach the destination node

(iii) *Overhead*: the average number of copies of a message in the system, at the time of reaching the destination

(iv) *Hop count*: the average hops a message takes to reach the destination node

5.2. Impact of Parameters. The learning rate δ and the discount factor η are critical parameters in our routing protocol. Next, we evaluate the impact of the two parameters on the delivery ratio of the FDQLR protocol using the RioBuses dataset. From Figure 5(a), we can find that when δ is larger than 0.9, the delivery ratio drops. Because δ determines the ability of new information to cover old information, if we choose $\delta = 1$, it means that agents only consider the newest information and ignore old information. Therefore, we choose $\delta = 0.88$ for the FDQLR protocol. In Figure 5(b), the delivery ratio rises with the increase of η. Because η determines the importance of future rewards, if η is too low, the immediate rewards are mainly optimized in learning; if η is high, the future rewards are obtained. Therefore, we choose $\eta = 0.4$ for the FDQLR protocol.



(a) Delivery ratio vs. TTL

(b) Delivery delay vs. TTL

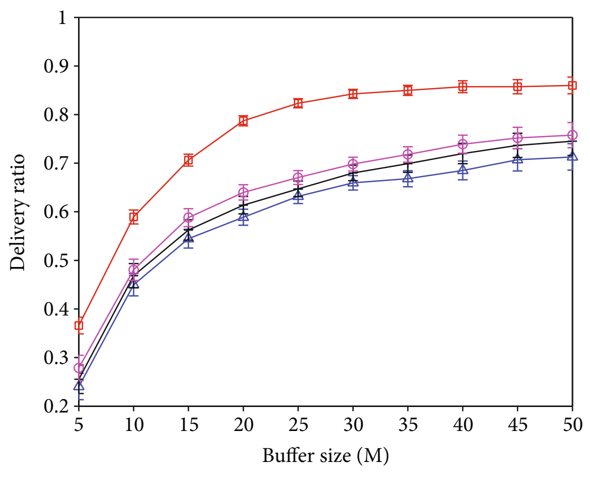
(c) Overhead vs. TTL

FIGURE 6: Performance of FDQLR with different σ.

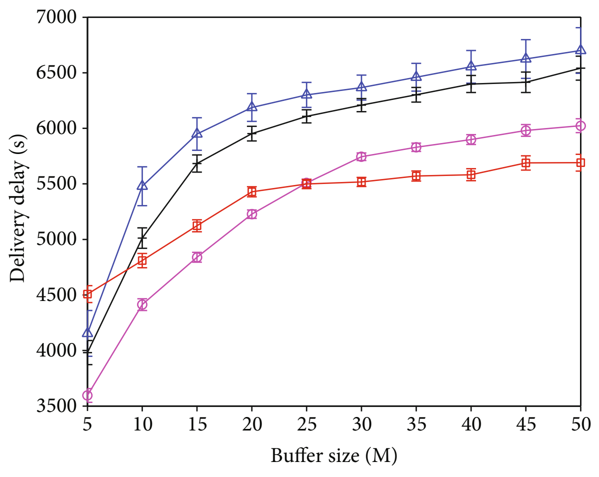
TABLE 5: Protocol parameters.

Routing protocol	Q-Learning	DQLR	FDQLR
δ (learning rate)	0.88	0.88	0.88
η (discount factor)	0.4	0.4	0.4
ε (encounter weight)	—	—	0.1
α (balance factor)	—	0.75	0.75
β (scaling constant)	—	0.1	0.1
γ_1 (aging constant of Q value)	0.13	0.13	0.13
γ_2 (aging constant of hot zone)	—	—	0.35
ω (joint coefficient)	—	—	0.82
σ (fuzzy weight)	—	—	1

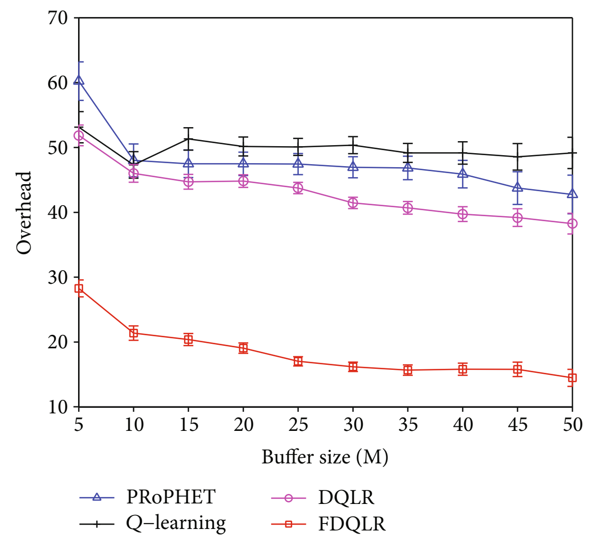
The relationship between the fuzzy reward coefficient $F_c(d)$ and the immediate reward $\tilde{R}_c(d,i)$ is observed by adjusting the fuzzy weight σ. When $\sigma = 1$ and $\sigma = 0$, $\tilde{R}_c(d,i)$ is completely related and unrelated to $F_c(d)$, respectively. In particular, $\tilde{R}_c(d,i)$ is partially related to $F_c(d)$ when $\sigma = 0.5$. In order to study the effect of fuzzy logic, we evaluate the performance of FDQLR with different values of σ. It can be seen from Figure 6 that the delivery ratio, delivery delay, and overhead of the FDQLR protocol are improved continually with the increase of σ. That is, the fuzzy logic has a greater impact on the performance of the protocol. Moreover, notice that the results of FDQLR with $\sigma = 0$ are equivalent to those of DQLR with a hot zone and a drop mechanism added. These phenomena validate fuzzy logic as well. Hence, we take $\sigma = 1$ in the following performance evaluation of the FDQLR protocol.



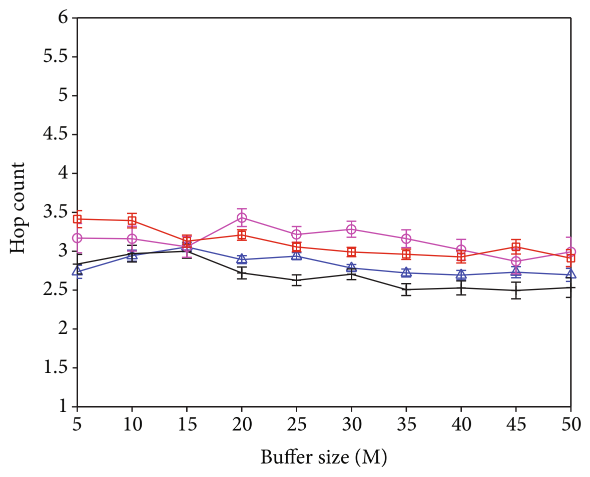
(a) Delivery ratio vs. buffer size



(b) Delivery delay vs. buffer size



(c) Overhead vs. buffer size

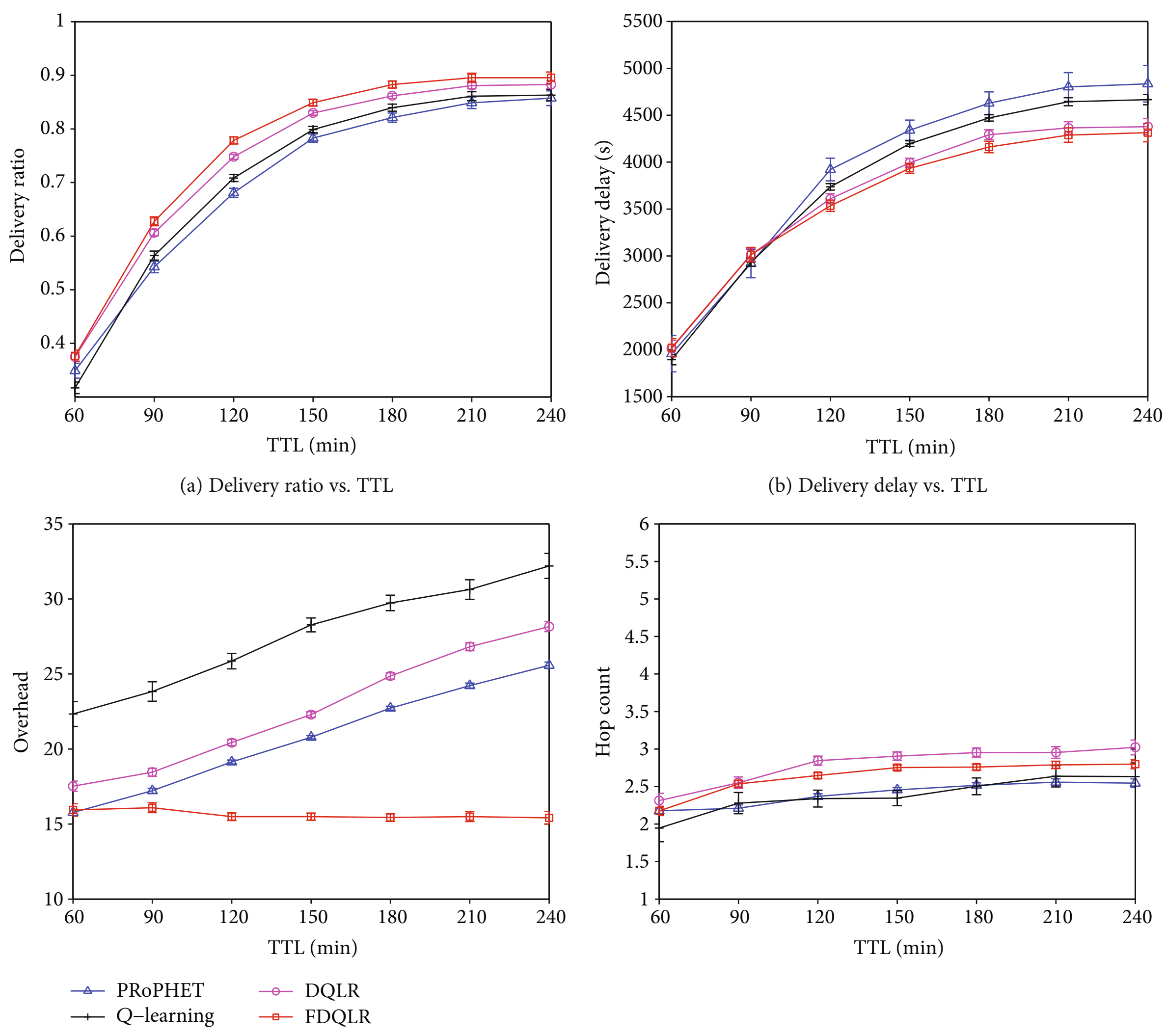


(d) Hop count vs. buffer size

Figure 7: Performance of protocols with different buffer sizes (RioBuses).

5.3. Performance Comparison. Firstly, we compare the performance of the FDQLR protocol with the PRoPHET, Q-Learning, and DQLR protocols with regard to the delivery ratio, delivery delay, overhead, and hop count while changing the buffer size of nodes and TTL of messages. The PRoPHET protocol is one of the early routing protocols in DTNs, which determines whether to forward messages based on the historical information about the encounters between nodes and deliveries of messages. The Q-Learning protocol utilizes a simple forwarding strategy, which determines whether to forward messages according to future rewards. The DQLR protocol is our previous work that uses the Double Q-Learning algorithm to decouple the selection from the evaluation, and it also uses the dynamic reward mechanism and intermediate value mechanism to improve forwarding efficiency. The FDQLR protocol extends the DQLR protocol, which is based on fuzzy logic, and proposed the hot zone mechanism and the drop mechanism additionally. The parameters in the Q-Learning, DQLR, and FDQLR protocols are shown in Table 5, where the values of δ, η, and σ are determined according to the simulation results in Section 5.2, and other parameters are set by our experiences that make FDQLR perform better. For comparison under the same condition, the same parameters of different protocols are set to the same values. Besides, the PRoPHET protocol has four parameters, and we use the default PRoPHET protocol parameters as recommended in the ONE simulator.

Figures 7(a)–7(d) show the simulation results of the delivery ratio, delivery delay, overhead, and hop count versus buffer size for different protocols using the RioBuses dataset. It can be seen from Figure 7(a) that the delivery ratio of different protocols increases with the buffer size because there can be more message copies stored in the network as the buffer size increases, which makes more messages to be delivered successfully. The delivery ratio of PRoPHET is lower than that of the other three protocols since PRoPHET forwards messages only according to the historical information, but the other Q-Learning-based protocols utilize the future rewards further. In addition, DQLR obtains a larger delivery ratio than the Q-Learning protocol due to the unbiased



(a) Delivery ratio vs. TTL

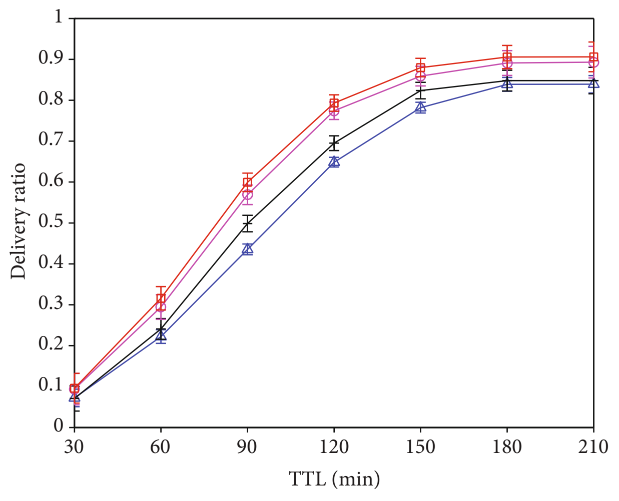
(b) Delivery delay vs. TTL

(c) Overhead vs. TTL

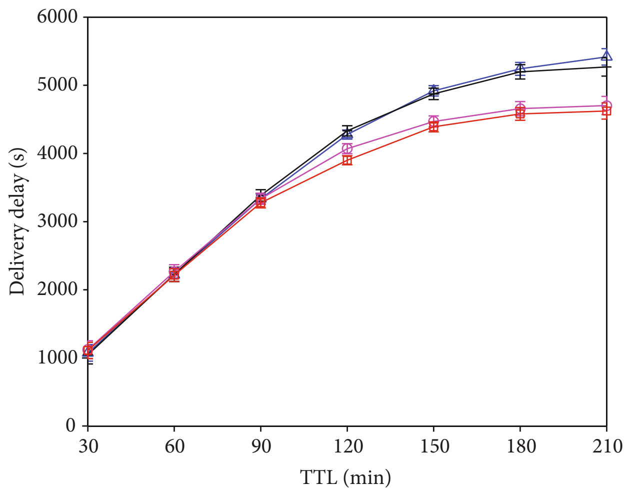
(d) Hop count vs. TTL

FIGURE 8: Performance of protocols with different TTL (RioBuses).

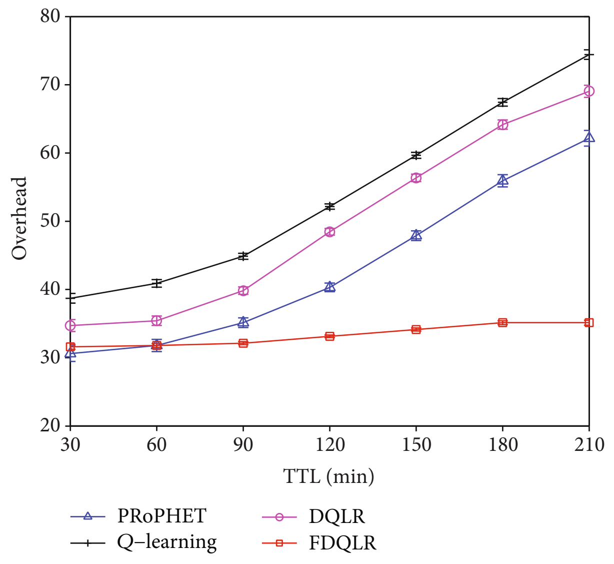
Double Q-Learning algorithm. Furthermore, the delivery ratio of FDQLR is obviously higher than that of other protocols, e.g., when the buffer size is 25 M, FDQLR can effectively improve the delivery ratio by about 18%, 20%, and 22% compared with the DQLR, Q-Learning, and PRoPHET protocols, respectively. This is because FDQLR uses fuzzy logic to jointly consider the node characteristics with the dynamic reward mechanism and largely avoids forwarding messages to nodes with poor status. Besides, it has been shown in Figure 6 that FDQLR is better than DQLR with the hot zone and drop mechanisms added, which verifies the effectiveness of fuzzy logic as well. Figure 7(b) shows the delivery delay of different protocols. Their delivery delay also increases with the buffer size, because when the buffer increases, the number of messages to be forwarded also increases, and the average delivery delay will increase as more messages are delivered. Furthermore, the queuing delay may increase when more messages are stored in the buffer. It can be seen from Figure 7(b) that the delivery delay of Q-Learning and DQLR is smaller than that of PRoPHET. However, for FDQLR, when the buffer size is small (e.g., ≤25 M), the delivery delay of FDQLR is relatively high, since the larger delivery ratio of FDQLR will lead to more messages with higher delay to be delivered. Otherwise, when the buffer size is increased (e.g., >25 M), the delivery delay of FDQLR becomes the lowest compared with the other three protocols. In this case, the drop mechanism of FDQLR plays an important role to discard those older messages and reduce the delivery delay. In Figure 7(c), the overhead of different protocols decreases with the increase of the buffer size as a whole. The reason is that the average number of copies of each message decreases with the increase of delivered messages in the network, and thus, the overhead is reduced. Besides, we can see that the overhead of FDQLR protocol is greatly lower than other protocols because FDQLR uses the drop mechanism, which discards both the delivered and the older message copies in the network to provide buffer space for new coming messages. Figure 7(d) shows the hop count of different protocols. Their hop count is maintained at about 3, which does not change obviously with the increase of the buffer size. Thus, all of



(a) Delivery ratio vs. TTL



(b) Delivery delay vs. TTL



(c) Overhead vs. TTL

Figure 9: Performance of protocols with different TTL (SPMBM).

these four protocols perform very well on the hop count metric, while none of them shows the advantage obviously.

Figures 8(a)–8(d) show the simulation results of the delivery ratio, delivery delay, overhead, and hop count versus TTL for these four protocols using the RioBuses dataset. As shown in Figure 8(a), with the increase of the TTL, the delivery ratio of all protocols increases, because messages will have a longer time to be delivered when the TTL becomes larger. The delivery ratio of FDQLR is also higher than that of other protocols, e.g., when the TTL is 120 min, FDQLR can effectively improve the delivery ratio by about 3%, 5%, and 8% compared with the DQLR, *Q*-Learning, and PRoPHET protocols, respectively. Furthermore, in Figure 8(b), we can see that the delivery delay increases with the TLL since the average lifetime of the delivered messages increases as the TTL increasing, and FDQLR protocol shows better performance in delivery delay as a whole. Additionally, as shown in Figure 8(c), the overhead of the *Q*-Learning, DQLR and PRoPHET protocols increases with the TTL because the number of copies of each message will increase when the TTL becomes large. On the contrary, FDQLR protocol shows very low overhead compared with others. In addition, it is interesting that the overhead of FDQLR protocol decreases with the TTL. The main reason for this phenomenon is that the FDQLR protocol uses a drop mechanism, which will actively discard older message copies with the increase of the TTL, and so the number of older messages in the network will decrease. Next, Figure 8(d) also shows that these four protocols have a similar performance on hop count, which is between 2 and 3 within the change range of TTL.

To evaluate the behaviour of the proposed protocols with different scenarios, node densities, and mobility models, the SPMBM model is used to compare the performance of different protocols. Figures 9(a)–9(c) show the simulation results of delivery ratio, delivery delay, and overhead versus TTL for these four protocols using the SPMBM model. It can be seen that the results of the SPMBM model are similar to the previous experiments, that is, FDQLR can achieve the highest delivery ratio and lowest delivery delay and overhead compared with the PRoPHET, Q-Learning, and DQLR protocols. In conclusion, the FDQLR protocol can improve the delivery ratio with lower delivery delay and overhead through the proposed mechanisms in this paper.

6. Conclusion

In this paper, the routing protocol FDQLR is proposed, which considers the complexity, dynamics, and uncertainty of characteristics in the DTNs and integrates the fuzzy logic and Double Q-Learning algorithm to optimize messages forwarding. Then, the fuzzy-based dynamic reward mechanism is proposed to comprehensively evaluate the status of the nodes with fuzzy logic, and the fuzzy reward coefficient is obtained with the fuzzy rules in the fuzzy process. Besides, the hot zone mechanism is proposed based on the encounter table to improve the efficiency of message forwarding. Furthermore, the drop mechanism is proposed to discard the delivered and older message copies to reduce the buffer occupation. The simulation results show that the fuzzy logic can obviously improve the performance of the FDQLR protocol, which can achieve a satisfactory delivery ratio with low forwarding overhead compared with other routing protocols of DTNs.

In the future, we are going to evaluate the proposed fuzzy system with different methods of fuzzy inference and defuzzification as well as fuzzy rule base and membership functions to increase its efficiency. We are also planning to optimize the parameters of the proposed protocol to improve its performance by more extensive simulations under different mobility models and traces. Last but not least, we will research on the adaptive learning mechanism of the protocol so that the FDQLR protocol can be more suitable for the real network environment of DTNs.

Data Availability

The datasets generated and analyzed during the current study are available from the corresponding author on reasonable request (Email: jgwu@njupt.edu.cn).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is supported by the National Natural Science Foundation of China under Grant Nos. 61872191 and 41571389; Open Research Fund from the Key Laboratory of Computer Network and Information Integration (SEU), Ministry of Education, China, under Grant No. K93-9-2014-05B; and Scientific Research Foundation of NUPT under Grant No. NY214063.

References

[1] D. Zhou, M. Sheng, B. Li, J. Li, and Z. Han, "Distributionally robust planning for data delivery in distributed satellite cluster network," *IEEE Transactions on Wireless Communications*, vol. 18, no. 7, pp. 3642–3657, 2019.

[2] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017.

[3] S. Zhang, J. Chen, F. Lyu, N. Cheng, W. Shi, and X. Shen, "Vehicular communication networks in the automated driving era," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 26–32, 2018.

[4] K. Sakai, M. Sun, and W. Ku, "Data-intensive routing in delay-tolerant networks," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pp. 2440–2448, Paris, France, April-May 2019.

[5] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," in *Tech. Rep. CS-200006, Duke University, Department of Computer Science*, Durham, NC, USA, 2000.

[6] J. Cui, S. Cao, Y. Chang, L. Wu, D. Liu, and Y. Yang, "An adaptive spray and wait routing algorithm based on quality of node in delay tolerant network," *IEEE Access*, vol. 7, pp. 35274–35286, 2019.

[7] S. Jain, K. Fall, and R. K. Patra, "Routing in a delay tolerant network," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 145–158, 2004.

[8] P. Raveneau, R. Dhaou, E. Chaput, and A.-L. Beylot, "DTNs back: DTNs broadcasting ACK," in *2014 IEEE Global Communications Conference*, pp. 2789–2794, Austin, TX, USA, December 2014.

[9] S. D. Han and Y. W. Chung, "An improved PRoPHET routing protocol in delay tolerant network," *The Scientific World Journal*, vol. 2015, Article ID 623090, 7 pages, 2015.

[10] V. G. Rolla and M. Curado, "A reinforcement learning-based routing for delay tolerant networks," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 10, pp. 2243–2250, 2013.

[11] F. Yuan, J. Wu, H. Zhou, and L. Liu, "A double Q-learning routing in delay tolerant networks," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pp. 1–6, Shanghai, China, May 2019.

[12] Q. Yuan, I. Cardei, and J. Wu, "An efficient prediction-based routing in disruption-tolerant networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 1, pp. 19–31, 2012.

[13] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, A Bradford Book, 2nd edition, 2018.

[14] H. V. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-Learning," in *The 30th AAAI Conference on Artificial Intelligence (AAAI'16)*, pp. 2094–2100, Phoenix, USA, March 2016.

[15] C. Wu, K. Kumekawa, and K. Kato, "Distributed reinforcement learning approach for vehicular ad hoc networks," *IEICE Transactions on Communications*, vol. E93-B, no. 6, pp. 1431–1442, 2010.

[16] A. Elwhishi, P. H. Ho, K. Naik, and B. Shihada, "ARBR: adaptive reinforcement-based routing for DTN," in *2010 IEEE 6th International Conference on Wireless and Mobile Computing, Networking and Communications*, pp. 376–385, Niagara Falls, ON, Canada, October 2010.

[17] R. Plate and C. Wakayama, "Utilizing kinematics and selective sweeping in reinforcement learning-based routing algorithms for underwater networks," *Ad Hoc Networks*, vol. 34, pp. 105–120, 2015.

[18] L. A. Zadeh and R. A. Aliev, *Fuzzy logic theory and applications: part I and part II*, World Scientific, 2018.

[19] J. Makhlouta, H. Harkous, F. Hutayt, and H. Artail, "Adaptive fuzzy spray and wait: efficient routing for opportunistic networks," in *2011 International Conference on Selected Topics in Mobile and Wireless Networking (iCOST)*, Shanghai, China, October 2011.

[20] S. Jain, M. Chawla, V. N. G. J. Soares, and J. J. Rodrigues, "Enhanced fuzzy logic-based spray and wait routing protocol for delay tolerant networks," *International Journal of Communication Systems*, vol. 29, no. 12, pp. 1820–1843, 2016.

[21] P. Nabhani and S. Radmanesh, "Adaptive fuzzy routing in opportunistic network (AFRON)," *International Journal of Computer Applications*, vol. 52, no. 18, pp. 7–11, 2012.

[22] S. Rahimi and M. A. J. Jamali, "A hybrid geographic DTN routing scheme based on fuzzy logic in vehicular ad hoc networks," *Peer-to-Peer Networking and Applications*, vol. 12, no. 1, pp. 88–101, 2019.

[23] C. Wu, S. Ohzahata, and T. Kato, "Routing in VANETs: a fuzzy constraint Q-learning approach," in *2012 IEEE Global Communications Conference (GLOBECOM)*, pp. 195–200, Anaheim, CA, USA, December 2012.

[24] S. K. Dhurandher, J. Singh, M. S. Obaidat, I. Woungang, S. Srivastava, and J. J. P. C. Rodrigues, "Reinforcement learning-based routing protocol for opportunistic networks," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, Dublin, Ireland, June 2020.

[25] Y. Liu, C. M. Eckert, and C. Earl, "A review of fuzzy AHP methods for decision-making with subjective judgements," *Expert Systems with Applications*, vol. 161, article 113738, 2020.

[26] A. Keränen, T. Kärkkäinen, and J. Ott, "Simulating mobility and DTNs with the ONE," *Journal of Communications*, vol. 5, no. 2, pp. 92–105, 2010.