Research Article

A Probabilistic Spray-and-Wait Routing Algorithm Based on Node Interest Preference in Delay Tolerant Networks

Yanan Chang,1 Ruijie Zhang,1 Jianqun Cui,1 Hao Zhou,2 Qiyun Wan,1 and Jike Wu1

1School of Computer, Central China Normal University, Wuhan, China
2Network and Informatization Center, Wuhan Polytechnic University, Wuhan, China

Correspondence should be addressed to Jianqun Cui; jqcui@126.com

Received 7 January 2022; Accepted 16 April 2022; Published 17 May 2022

Academic Editor: Jian Su

Copyright © 2022 Yanan Chang 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.

How to select proper relay nodes to ensure the successful delivery of messages is still a hot topic in delay tolerant networks (DTN). In this paper, we propose a probabilistic Spray-and-Wait routing algorithm based on node interest preference (called NIP-PSW). Firstly, considering the influence of the social attributes of nodes, we define a metric called node interest preference (NIP) to measure the probability of nodes becoming friends. Secondly, in view of the influence of node quality and connection time between nodes on message forwarding, we define the delivery probability (DP). Finally, according to the historical information of nodes, the node interest similarity (NIS) is proposed. In spray phase, NIP and DP are used to select the relay node and allocate the number of message copies adaptively. In wait phase, it is judged whether to forward the message to the encountering nodes again according to the NIS and the DP. In addition, the concept of message storage value (MSV) and the acknowledgment (ACK) mechanism are introduced to manage the buffer of nodes. The simulation results show that the NIP-PSW not only can significantly improve the delivery rate and reduce the average delay but also shows good performance in network overhead and average number of hops.

1. Introduction

The delay/disruption tolerant network (DTN) generally refers to a kind of network in which nodes are connected intermittently or even interrupted for a long time due to environmental and resource problems, resulting in high latency [1, 2]. Different from the traditional network, the DTN shows the characteristics of frequent splitting of the connection state between nodes, the network topology changes randomly, no stable end-to-end transmission path, and the energy resources of the nodes are limited. So, the transmission protocol of the traditional network cannot be applied to DTN. Therefore, DTN uses the LTP transmission protocol [3] and the “Store-Carry-Forward” mechanism to forward messages [4].

In recent years, many DTN routing algorithms have been proposed. Classical algorithms include Epidemic [5] algorithm based on flooding strategy, Prophet [6] algorithm based on probability, and Spray-and-Wait [7] algorithm to limit the number of message copies. With the development of wireless communication technology and the popularization of communication equipment, the network communication environment has shown regular social attributes due to the social laws of humans. The communication between nodes no longer simply relies on random encounter opportunities to complete message forwarding but presents a stable social attribute. For example, two men who frequently visit the same place indicate that they may have common interests and hobbies, and the probability of encountering between them will be higher. In addition, the more friends in common, the higher the probability of them being friends. Therefore, in DTN, considering the social relationship of nodes, applying it to the routing strategy will greatly improve the efficiency of the routing algorithm. For example, Bubble Rapp [8] algorithm divides each
node into a community, then sorts the rank of each node in the community and forwards messages to a higher rank node. SimBet [9] algorithm calculates the utility function through the intermediate degree and similarity in the community.

However, the classic algorithms mentioned above exist many problems. For example, the Spray-and-Wait algorithm does not consider the performance of the relay node when forwarding messages, only uses a relatively single stop and wait protocol in the wait phase, and does not fully consider the opportunities of encountering other nodes. The Prophet algorithm only considers the historical encounter records between nodes, ignoring the influence of the connection time between nodes and the resource status of the nodes on message forwarding. In response to these problems, many studies have made improvements. In [10], considering the influence of connection time on message delivery, when calculating the delivery probability, the connection time between nodes is introduced. In [11], the number of message copies is dynamically allocated according to nodes' delivery probability. NPSW [12], introducing the activity of node, forwards messages to a node with higher activity. However, the above routing algorithms do not consider that the ability of nodes to deliver messages will change with network conditions. When calculating the delivery probability, each node updates the probability on the basis of the same default probability, which ignores the difference of nodes.

Furthermore, the performance of the routing algorithm will be greatly improved when the social attributes between nodes are considered. In [13], the social circle of nodes is divided according to the historical encounter records of the nodes. The node in the same social circle as the destination node is preferentially selected as a relay node to increase the delivery rate. The paper [14] considers that in society, human life is regular, they often appear in certain places at a certain time. So temporal and spatial attributes of nodes are introduced into the routing algorithm. In [15], it presents a comprehensive research of human mobility issues in three aspects: human mobility characteristics, mobility traces, and mobility prediction. It shows that nodes with higher social similarity are more likely to be divided into the same community, the more opportunities they have to meet, and the more likely they are to forward messages. These studies show that each node possesses many social attributes due to the social connections of the people carrying these devices in social networks. So, reasonable use of these attributes will improve the efficiency of our algorithm to a certain extent.

However, the biggest difficulty is which social attributes to use and how to quantify these attributes. The paper [14] only considers historical encounter nodes to divide social circles. The article [16] introduces a variety of social attributes, but when considering the mobile similarity, differences in nodes’ interest in different areas are ignored. The access frequency of nodes to different areas is different. The more visits to an area, the higher the node’s interest in this area. In addition, most of these algorithms ignore a problem. Although these nodes with high social similarity may have a greater chance of encountering, it does not mean that they can successfully deliver messages between them. Therefore, whether they can successfully forward message is also a problem that needs to be considered.

Based on this background, this paper proposes a probabilistic Spray-and-Wait routing algorithm based on node interest preference (called NIP-PSW). First, we not only record the area that the node has visited but also record the number of visits to the area, so as to distinguish the degree of interest of nodes in different areas. Second, we call the encounter nodes of a node as friend nodes and then use the number of encounters to measure the friendliness between nodes. We quantify the two aspects of data with two vectors to abstract the social attribute of nodes into multidimension. Finally, we estimate the similarity of node interests through cosine similarity. Furthermore, in addition to considering the node interest, we introduce delivery probability to measure the node’s ability to deliver messages. The delivery probability of a node is not only affected by the state of the network but also related to the state of the node. Therefore, when calculating the delivery probability, we not only pay attention to the influence of the network state but also consider the differences between nodes. And we introduce the node quality and probability impact factor to calculate the node delivery probability. The main contributions are summarized as follows.

(i) We propose a new metric called node interest preference (NIP) based on the social attributes of nodes. The NIP, considering both the historical encounter records of nodes and the geographical areas the nodes passed by, is used to determine the possibility of becoming friends between the nodes and predict the possibility of the relay node encountering the destination node.

(ii) We present a metric called quality of node (QoN) to reflect the ability of the node to process messages. Then, the probability impact factor (PIF) is proposed to measure the condition of the network and the strength of connections. Based on QoN and PIF, we define an adaptive delivery probability. We use this delivery probability and NIP to improve the spray phase of Spray-and-Wait Algorithm. In addition, the delivery probability is used to adjust the distribution of message copies.

(iii) We propose a concept called node interest similarity (NIS) to measure the relationship of nodes’ interests. In wait phase, the messages will be forwarded to a node with higher NIS and delivery probability to the destination node.

(iv) Finally, we define a concept called message storage value (MSV) to manage nodes’ buffer. When nodes’ buffer is insufficient to receive new messages, the message with the smallest MSV value will be deleted.

The rest of this paper is organized as follows. We discuss the related work in Section 2. In Section 3, the main designs of our algorithm are introduced. Section 4 presents the
details of NIP-PSW. The simulation setup and results are discussed in Section 5. In Section 6, some conclusions are provided and the future work is also pointed out.

2. Related Work

Designing an efficient and low-consumption routing algorithm is the key to solving the problems in the delay tolerant network. Since the development of the delay tolerant network, many algorithms have been proposed. Epidemic [5] algorithm is a routing algorithm based on the flooding strategy. It draws on the idea of infectious diseases in biology. Each node carrying a message will replicate the message to the encountering node like an infectious disease, to maximize the message delivery rate. At the same time, the Epidemic algorithm caused more waste of resources and redundant copies of messages. Prophet [6] is also one of the classic algorithms. Relying on the historical information of the nodes, it believes that two nodes that were often encountered in the past are likely to encounter again in the future. Based on this, it introduces the concept of delivery probability to limit the flooding between messages and selects nodes with a higher delivery probability to carry messages. However, the overhead of the Prophet is still very high, and it does not consider the current connection status between nodes. Spray-and-Wait [7] reduces the overhead of network resources by limiting the number of message copies. The source node of a message primitives carries a fixed number of copies and, after encountering other nodes, allocates one copy of the message to the encountering node. When the number of message copies held by a node is 1, the node will enter the wait phase, and the message will not be forwarded until the destination node is encountered. However, the Spray-and-Wait algorithm does not screen the relay nodes in the spray phase, but blindly forwards the message directly, and when assigning copies, it does not fully consider the situation of the nodes and allocates the message copies unreasonably. In addition, the wait phase does not make full use of the opportunity to encounter other nodes, resulting in a high delay.

Many existing studies have also made improvements to the problems of these classic algorithms. Wang et al. [10] considered that the connection time has an important influence on the successful forwarding of messages and proposed a probabilistic routing algorithm based on the connection time. This algorithm introduces the connection time on the basis of the Prophet algorithm, which can screen out better relay nodes and improve the message delivery rate effectively. Kim et al. [11] proposed a probability-based spray and wait protocol in delay tolerant networks, which allocates message copies according to the ratio of the probability of nodes. Dai et al. [12] analyzed that the node in the wait phase did not take advantage of the opportunity to encounter other nodes and just waited for the encounter with the destination node, which brings about the high delay and low delivery rate, so they proposed the NPSW algorithm. This algorithm forwards message to the node with a higher delivery rate in the wait phase. Wu et al. [17] considered that the higher the activity of a node, the more opportunities for the node to exchange information with other nodes, so they proposed a Spray-and-Wait routing algorithm based on node activity, which chose to distribute messages to nodes with higher activity, and the higher the activity, the more copies are allocated. Wu et al. [13] introduced the influence of the social and geographic attributes of the node on routing algorithm and the dead-end problem (A node has a high delivery rate but it will not encounter the destination node and will not forward the message, causing the message to be stored in its buffer until the message is discarded) in the wait phase, so they proposed the SCAW algorithm. This algorithm divides social circle based on the historical encounter records of nodes. Based on the social circle, geographical routing and probabilistic routing are used to select relay nodes, then the method of respraying is used in the wait phase to add message copies and increases the message delivery rate. Cui et al. [18] proposed the concept of quality of node and use the quality of node to dynamically allocate the number of message copies. Jia et al. [14] believed that some nodes that often appear at the same time and in the same place are easier to exchange information, so they proposed the GTSP algorithm, which improves the message delivery rate to a certain extent. Wei et al. [19] also considered the influence of node connection on message forwarding and proposed node connection strength to redefine the delivery probability. They use this probability to select relay nodes and allocate message copies adaptively. Hu et al. [20] considering the problems of the Spray-and-Wait algorithm defined a metric to select the encountered nodes according to the movement state in the spray phase. And the number of message copies is distributed adaptively by the metric. In the wait phase, the message is forwarded to the node with higher activity.

Although these algorithms have made improvements to many problems existing in classic algorithms, many algorithms are not comprehensive enough to solve the problems. Some only consider the influence of connection time on message forwarding; the influence of the current node’s or the relay node’s condition was ignored. Some algorithms ignore the impact of copy allocation on the algorithm. Although some do consider it, they do not consider whether the current node is capable of receiving messages. Other algorithms use the attributes of time and space in social networks as an important metric to select relay nodes, but they only make simple records and do not make full use of these attributes to make decisions.

To address the issues mentioned above, this paper proposes an algorithm based on node interest preference. Because of the social attributes showed by nodes, it is believed that nodes with more mutual friends are more likely to exchange information when they often appear in the same area. So, we propose a new metric for selecting relay nodes, which is defined as node interest preference. At the same time, to ensure that the delivery probability is adjusted adaptively according to the network condition and the node’s condition, the concept of quality of node is introduced. In addition, the connection time and the buffer size of the
encountered nodes are also considered to calculate the delivery probability. In spray phase, we select the appropriate relay node according to NIP and delivery probability and then dynamically adjust the number of message copies according to the proportion of delivery probability. It is also reasonable to use the delivery probability to adjust the message copy because the distribution of the message copy by the delivery probability not only considers the relationship between the source node and the relay node but also considers the buffer size of both nodes. In wait phase, the node interest similarity is introduced as a metric to judge the interest relationship between the relay node and the destination node. If the value is large and the delivery probability is also high, it is considered that the relay node has the same interest as the destination node, and there is a high probability of encountering the destination node, so a copy of the message is replicated to the node. Finally, to reduce the impact of the insufficient buffer of nodes on the entire routing algorithm, the concept of message storage value is introduced, and preferentially discard messages with lower storage value. In addition, we also introduce the ACK mechanism to delete messages that have been successfully forwarded in the node and delete redundant message copies in time.

3. System Design

3.1. Network Model. In real society, people who often go to the same place usually indicate that they have common interests and hobbies, the more likely they are to be friends. Based on the stable social attributes that exist between nodes, it is found that nodes often appear in the same area are more likely to have the opportunity to forward messages. Therefore, we divide the whole area into many subareas and use \( R_i = \{ r_k | 1 \leq k \leq m \} \) to record the subareas passed by node \( v_i \), where \( r_k \) denotes the \( k \)th sub-area and \( m \) is the total number of subareas. We use the vector \( W_{R_i} = \langle w_k | 1 \leq k \leq m \rangle \) to record the number of times that node \( v_i \) passes through each subarea in \( R_i \), where \( w_k \) is the number of times that \( v_i \) passed the subarea \( r_k \). In addition, \( V_i^E = \{ v_j | 1 \leq j \leq n \} \) is used to record the nodes that \( v_i \) encountered, where \( n \) is the total number of nodes. And \( WV_i^E = \langle w_j | 1 \leq j \leq n \rangle \) is used to record the times of \( v_i \) encountering the nodes in \( V_i^E \), where \( w_j \) is the times that \( v_i \) encounters \( v_j \). The summary of important used variables in the paper is listed in Table 1.

We take Figure 1 as an example to illustrate how node \( v_1 \) records the data mentioned above from time \( t_0 \) to \( t_3 \). In the figure, each big circle represents a subarea, and each subarea has its own area identifier. At time \( t_0 \), \( v_1 \) is in area \( r_1 \), and no other nodes are encountered. So, \( R_1 = \{ r_1 \}, W_{R_1}(1), V_i^E = \{ \}, WV_i^E = \{ \} \), and \( WV_{V_i} = \{ \} \). At time \( t_1 \), \( v_1 \) is in area \( r_2 \) and encounters \( v_2 \). And update the data, \( R_1 = \{ r_1, r_2 \}, W_{R_1}(1, 1), V_i^E = \{ v_2 \}, WV_i^E = \{ v_2 \} \). At time \( t_2 \), \( v_1 \) is in area \( r_3 \) and encounters \( v_3, v_6 \) and \( v_5 \) again, so update the data to \( R_1 = \{ r_1, r_2, r_3 \}, W_{R_1}(1, 1, 1), V_i^E = \{ v_2, v_3, v_6 \}, WV_i^E = \{ v_2, v_3, v_6 \} \), and \( WV_{V_i} = \{ v_2, v_3, v_6 \} \). At time \( t_3 \), \( v_1 \) returns to area \( r_1 \) and encounters \( v_5 \), so update the data, \( R_1 = \{ r_1, r_2, r_3 \}, W_{R_1}(2, 1, 1), V_i^E = \{ v_2, v_5, v_6 \}, WV_i^E = \{ v_2, v_5, v_6 \} \), and \( WV_{V_i} = \{ v_2, v_5, v_6 \} \).

We take Figure 1 as an example to illustrate how node \( v_1 \) records the data mentioned above from time \( t_0 \) to \( t_3 \). In the figure, each big circle represents a subarea, and each subarea has its area identifier. At time \( t_0 \), \( v_1 \) is in area \( r_1 \), and no other nodes are encountered. So, \( R_1 = \{ r_1 \}, W_{R_1}(1), V_i^E = \{ \}, WV_i^E = \{ \} \). At time \( t_1 \), \( v_1 \) is in area \( r_2 \) and encounters \( v_2 \). And update the data, \( R_1 = \{ r_1, r_2 \}, W_{R_1}(1, 1), V_i^E = \{ v_2 \}, WV_i^E = \{ v_2 \} \). At time \( t_2 \), \( v_1 \) is in area \( r_3 \) and encounters \( v_3 \) again, so update the data to \( R_1 = \{ r_1, r_2, r_3 \}, W_{R_1}(1, 1, 1), V_i^E = \{ v_2, v_3 \}, WV_i^E = \{ v_2, v_3 \} \), and \( WV_{V_i} = \{ v_2, v_3 \} \).

3.2. Node Interest Preference. In many application scenarios of DTN, the movement of nodes is not completely random. In fact, the nodes usually show stable social laws. In addition, finding a suitable relay node to ensure the successful forwarding of messages is the key to improving the performance of the DTN routing algorithm. For the above reasons, the node interest preference is introduced to evaluate nodes’ interests. And it both considers the influence of geographical factor and social factor on nodes’ interest. It is used as a metric to measure the possibility of interaction between nodes. In order to quantify the geographic relationship of nodes, make full use of the impact of geographic relationships on routing, we define a metric called regional similarity.

**Definition 1.** Regional similarity (RS). The similarity of the passed areas between \( v_i \) and \( v_j \) is said RS\(_{ij}\). The calculation of the RS\(_{ij}\) is shown in formula (1):

\[
RS_{ij} = \cos \left( W_{R_i}, W_{R_j} \right),
\]

where \( W_{R_i} \) is active area information vector of node \( v_i \) and \( W_{R_j} \) is active area information vector of node \( v_j \).

However, vector \( W_{R_i} \) and vector \( W_{R_j} \) usually have different dimensions. Therefore, the two vectors’ dimensions must be unified before calculating RS\(_{ij}\). To unify the vector dimension, first, find the union of two nodes’ active area set, and then, modify the dimension of the vector according to the union.

We use the data in Figure 1 to illustrate the calculation of RS\(_{12}\). Table 2 records the data we collected from Figure 1. The calculation process of RS\(_{12}\) is as follows:

1. Calculate the union of \( R_1 \) and \( R_2 \):

\[
R_1 \cup R_2 = \{ r_1, r_2, r_4 \}.
\]

2. Modify the dimension of \( WR_1 \) and \( WR_2 \) according to the union of \( R_1 \) and \( R_2 \):

\[
(2)
\]
Table 1: List of notations.

| Notation          | Explanation                                           |
|-------------------|-------------------------------------------------------|
| $V$               | Set of nodes in the network                          |
| $R_i = \{ r_k | 1 \leq k \leq m \}$ | Active area set of node $v_i$                       |
| $WR_i = \{ w_k | 1 \leq k \leq m \}$ | Active area information vector of node $v_i$         |
| $V'_i = \{ v_j | 1 \leq j \leq n \}$ | Friends set of node $v_i$                           |
| $WV'_i = \{ w_j | 1 \leq j \leq n \}$ | Friends information vector of node $v_i$            |
| $RS_{ij}$         | Regional similarity of node $v_i$ and node $v_j$     |
| $FS_{ij}$         | Friends similarity of node $v_i$ and node $v_j$      |
| $NIP_{ij}$        | Node interest preference of node $v_i$ to node $v_j$|
| $QoN_{ij}$        | Quality of node of node $v_i$                        |
| $DP_{ij}$         | Probability of node $v_i$ successfully forwarding message to node $v_j$ |
| $PIF_{ij}$        | Probability impact factor of the probability of node $v_i$ successfully forwarding message to node $v_j$ |
| $NIS_{ij}$        | Node interest similarity of node $v_i$ and node $v_j$|
| $MSV_{m_i}$       | Storage value of message $m_i$                       |
| $SV_i$            | Set of messages carried by $v_i$                     |
| $MSD_i$           | ACK message list of node $v_i$                       |
| $L_i(m_k)$        | Number of copies of message $m_k$ carried by node $v_i$ |
| $forwardList$     | Set of messages that node $v_i$ need to forward to node $v_j$ |
| $\theta_i$        | Threshold of node interest preference                |
| $\delta_i$        | Threshold of node interest similarity                |
| $Con(i)$          | Number of connections of $v_i$                       |

Figure 1: Node movement example.
\[ WR_1 = 2, 1, 1, \quad WR_2 = 0, 2, 1. \]  

(3) Calculate the \( RS(1, 2) \) of the two nodes according to formula (1):

\[ RS_{12} = \cos (WR_1, WR_2) = 0.64. \]  

As we all know, if two people have more mutual friends, the more likely they are to become friends and people are always willing to communicate with their friends. Based on the stable social relationship between nodes, we define a metric called friends similarity (we call historical encounter nodes as friends) to quantify this relationship and measure the similarity of friends between nodes.

**Definition 2.** Friends similarity (FS). The similarity of friends between \( v_i \) and \( v_j \), say \( FS_{ij} \), is the similarity of historical encounter nodes between \( v_i \) and \( v_j \). The calculation of the \( FS_{ij} \) is shown in formula (2):

\[ FS_{ij} = \cos \left( WV_i^E, WV_j^E \right), \]  

where \( WV_i^E \) is friends information vector of node \( v_i \) and \( WV_j^E \) is friends information vector of node \( v_j \).

The vector \( WV_i^E \) and vector \( WV_j^E \) may have different dimensions too. So, we also need to modify the two vectors’ dimension. The method is same as before. First, we find the union of two nodes’ friends set, and then, modify the dimension of vector according to the union. The calculation process of \( FS_{ij} \) is the same as that of \( RS_{ij} \).

Based on Definitions 1 and 2, to introduce geographic and social relationships into routing decisions, we define a metric called node interest preference to evaluate the interest of two nodes in each other.

**Definition 3.** Node interest preference (NIP). The NIP of \( v_i \) and \( v_j \), say \( NIP_{ij} \), is a weighted sum of \( RS_{ij} \) and \( FS_{ij} \). The calculation of the \( NIP_{ij} \) is shown in formula (3):

\[ NIP_{ij} = \alpha \cdot RS_{ij} + (1 - \alpha) \cdot FS_{ij}, \]  

where \( \alpha \in [0, 1] \) is a smoothing parameter, which is used to adjust the proportion of \( RS_{ij} \) and \( FS_{ij} \). NIP can be used as a metric to measure whether two nodes can become friends. The higher the value of NIP, the easier it is for two nodes to become friends. Therefore, when selecting a relay node, if the message is forwarded to a node with a greater NIP value of the destination node, it is easier to forward the message successfully.

3.3. Calculation and Update of the Adaptive Delivery Probability. In Prophet, if two nodes frequently encounter, the delivery probability will be high. However, if the two nodes do not establish a connection or the connection time is short, the buffer size of the encountered node is not enough to accept new messages, the delivery probability should be small. In addition, when Prophet calculates the delivery probability, each node is given the same initial value, and the calculation and update of the delivery probability are based on this initial value. It does not fully consider each node’s situation and the difference between nodes. Therefore, on the basis of Prophet’s delivery probability, we proposed a delivery probability that adaptively changes according to the network status and the node’s situation.

The ability of a node to handle messages is inseparable from the node’s buffer and message size. So, we define a metric called quality of node based on nodes’ buffer and the message size stored in the buffer. The quality of node is used to replace the fixed initial value when calculating the delivery probability in Prophet, which makes the calculation of delivery probability considering the differences of nodes and update adaptively by the node’s condition.

**Definition 4.** Quality of node (QoN). The QoN of node \( v_i \), say \( QoN_i \), calculated by the tanh function, is a normalized value of the ratio of buffer and message size, which can represent a node’s ability to handle messages. The calculation of \( QoN_i \) is shown in formula (4):

\[ QoN_i = \frac{2}{1 + e^{-2 \cdot FS \cdot avgM_i}} - 1, \]  

where \( FS \) represents the remaining buffer size of \( v_i \) and \( avgM_i \) represents the average size of all messages stored in \( v_i \).
The QoNi is calculated and normalized by the tanh function (a commonly used normalization formula). The ratio of buffer to message size is used as the independent variable of the tanh function.

Whether the message can be successfully delivered between nodes is inseparable from the connection time between nodes. In addition, not only the buffer of the source node must be considered but also the buffer of the relay node. Therefore, we define a metric called probability impact factor as a factor for calculating the delivery probability, which both considers the connection time between nodes and the buffer of the relay nodes.

**Definition 5.** Probability impact factor (PIF). The PIF of vi and vj, say PIFij, is a weighted sum of the connection time of vi and vj and the ratio of the remaining buffer size and total buffer size of vj. PIFij can be calculated by formula (5):

\[
PIF_{ij} = (1 - \sigma) \cdot C + \sigma \cdot \frac{FS_j}{S_j},
\]

where

\[
C = \text{sigmoid}(2 \cdot \lambda_{ij}) = \frac{1}{1 + e^{-2\lambda_{ij}}},
\]

\[
\lambda_{ij} = \frac{n \cdot \sum_{k}^{m} T_{ij}(k)}{\sum_{k}^{n} [\sum_{k}^{m} T_{j}(k)]},
\]

Suppose vi is the source node, and vj is the encountering node. In formula (9), \(\lambda_{ij}\) is the ratio of the total connection time of vi and vj to the connection time of all the nodes that vi encountered. It is a metric to measure the strength of this connection. We use a sigmoid function to normalize it. The sigmoid function image is shown in Figure 2. As is shown in the picture, the slope of sigmoid(2x) is larger, and the change of function value is also more obvious. Therefore, we multiply by 2 before \(\lambda_{ij}\), which can highlight the difference in connection time between nodes.

\[\sum_{k}^{m} T_{ij}(k)\] is the total connection time of m times of connections between vi and vj, n is the number of nodes connected to vj, \(\sum_{k}^{n} [\sum_{k}^{m} T_{j}(k)]\) is the total connection time between node vj and other n nodes. \(FS_j\) is the remaining buffer size of vj, and \(S_j\) is the initial buffer size of vj. \(\sigma \in [0, 1]\) is a variable parameter that used to adjust the relative influence of the connection time and the buffer size of the encountering node on the PIF.

Based on Definition 4 and Definition 5, we have improved the delivery probability of Prophet, so that the delivery probability can be updated adaptively according to the state of node and network.

**Definition 6.** Delivery probability (DP). The DP of vi to vj, say DPij, is the probability of vi delivering messages to vj successfully.

According to Prophet’s ideas, we also divide the calculation of DP into three phases: update, aging, and transitive update.

Update: when vi and vj encounter, calculate the delivery probability of vi to vj through the QoNi and PIFij. The update formula is shown in formula (8):

\[
DP_{ij}^{\text{new}} = DP_{ij}^{\text{old}} + \left(1 - DP_{ij}^{\text{old}}\right) \cdot QoNi \cdot PIF_{ij},
\]

In the formula, \(DP_{ij}^{\text{new}}\) represents the probability that vi successfully delivers message to vj now, and \(DP_{ij}^{\text{old}}\) represents the delivery probability of vi to vj before this encounter.

Aging: if vi and vj do not encounter in a while, the delivery probability of vi to vj decays according to

\[
DP_{ij}^{\text{new}} = DP_{ij}^{\text{old}} \cdot e^{-\tau},
\]

where \(\tau\) is the aging parameter; the number of time units that have elapsed from the last time they encountered to the current time they encountered is denoted as \(t\). In addition, the use of exponential aging can quickly lower the delivery probability between nodes that have not connected for some time, making the delivery probability between nodes have obvious differences, and can also reduce the possibility of source nodes choosing failed paths to forward messages.

Transitive update: if vi can transmit the message to vj, and vj can transmit the message to vr, then vi can transmit the message to vr through node vj. The calculation is shown in formula (10):

\[
DP_{ij}^{\text{new}} = DP_{ij}^{\text{old}} + \left(1 - DP_{ij}^{\text{old}}\right) \cdot DP_{ij} \cdot DP_{jr} \cdot \gamma,
\]

where \(\gamma\) is the transmission influence factor, which represents the proportion of the influence of transitivity on DP.

3.4. Calculation of Node Interest Similarity. When measuring the interest of nodes, in addition to introducing node interest preference, we also define a metric called node interest similarity. It is used to measure whether the node with higher NIS still needs to forward the message to prevent the dead-end problem.

**Definition 7.** Node interest similarity (NIS). The NIS of vi and vj, say NISij, is a metric to measure the interest relationship of vi and vj. The NISij can be calculated according to formula (11):

\[
NIS_{ij} = \mu \frac{|V^f_i \cap V^f_j|}{\min \left(|V^f_i|, |V^f_j|\right)} + (1 - \mu) \frac{|R_i \cap R_j|}{\min \left(|R_i|, |R_j|\right)},
\]

where \(\mu \in [0, 1]\) is a variable parameter, which is used to adjust the proportion of the historical encounter nodes and
the passed areas. So, if the \( NIS_{ij} \) is larger, it is proved that the interests of \( v_i \) and \( v_j \) are more similar.

3.5. Buffer Management. The delivery rate of messages and network performance will be severely affected because of the congestion of nodes. Therefore, an effective buffer management strategy can alleviate the congestion of nodes, lower network overhead, increase message delivery rate, and improve network performance. So, the ACK mechanism and message storage value are proposed to manage nodes’ buffer.

3.5.1. Acknowledgment (ACK) Mechanism. To delete redundant copies that have been successfully delivered to the destination, the ACK mechanism was introduced. For a message that has been successfully forwarded, it will not only occupy the node’s buffer but also affect the node’s ability to receive and forward other messages when the node’s buffer is insufficient and ultimately affect the entire network. Therefore, we let each node maintain a list \( MSD \) to record the ID of the message that has been successfully forwarded. Whenever two nodes establish a connection, before exchanging information, they will exchange their \( MSD \) list, add message ID which is not in their own \( MSD \) but in the encountered node’s \( MSD \) to their own \( MSD \), and delete these messages stored in nodes’ buffer.

3.5.2. Message Storage Value. When the node’s buffer is not enough to receive new messages, deleting the messages in the buffer reasonably and ensuring the node has sufficient buffer to receive new messages can improve the overall performance of the network.

The more times a message is forwarded, the wider the spread of the message in the network. If a node gives priority to discarding the message with more relay times when the buffer is insufficient, it will not have much impact on the delivery of the message to the entire network. Based on this, we define a metric called message storage value to evaluate the importance of storing a message.

**Definition 8.** Message storage value (MSV). The MSV of message \( m_i \), say \( MSV_{m_i} \), is a metric to measure the importance of the message stored in nodes, and the specific calculation formula is shown in

\[
MSV_{m_i} = \frac{TTL \cdot e^{-RT}}{S_{m_i}}.
\]

In the formula, TTL is the time to live of the message \( m_i \), RT is the number of times the message \( m_i \) forwarded in the entire network, and \( S_{m_i} \) is the size of the message \( m_i \). When a node does not have enough buffer to receive new messages, the message with the smallest MSV in the buffer is deleted first. The specific algorithm process is shown in Algorithm 1. First, the messages in the queue need to be sorted. The time complexity of the sorting algorithm is \( O(n\log n) \). Then, it is necessary to traverse the messages in the queue and forward the messages to the nodes, so the time complexity \( O(n) \).

Therefore, the time complexity of the whole algorithm is \( O(n\log n) \).

4. NIP-PSW

An probabilistic Spray-and-Wait routing algorithm based on node interest preference is divided into spray and wait phases.

4.1. Spray Phase. The traditional Spray-and-Wait algorithm only allocates one replica of the message to the relay node, whether the relay node can successfully forward the message to the destination node is not considered. The binary Spray-and-Wait algorithm forward half of the message copies to the relay nodes, which can make the message quickly spread and improve the performance of the algorithm to a certain extent. However, the transmission capacity of each node and the buffer of the node are not considered in the two algorithms. In the spray phase, the NIP-PSW uses \( NIP \) and \( DP \) as metrics to select relay nodes, because selecting nodes that are easy to become friends with the destination node to carry messages can increase delivery rate. At the same time, considering the difference in nodes’ buffer and message handling capabilities, the number of copies of messages are adaptively allocated according to the node’s delivery probability, which ensures that the node can receive the allocated message. When two nodes encounter, the number of message copies is allocated according to

\[
L_{i}^{\text{new}}(m_d) = \frac{DP_{id}}{DP_{id} + DP_{jd}} \cdot L_{i}^{\text{old}}(m_d),
\]

\[
L_{j}^{\text{new}}(m_d) = L_{i}^{\text{old}}(m_d) - L_{i}^{\text{new}}(m_d),
\]

where \( L_{i}^{\text{old}}(m_d) \) represents the number of copies of \( m_d \) carried by \( v_i \) before it encounters \( v_j \), \( L_{i}^{\text{new}}(m_d) \) represents the number of message copies of \( m_d \) that the \( v_j \) should be allocated, and \( L_{i}^{\text{new}}(m_d) \) represents the number of copies of \( m_d \) remaining after \( v_j \) allocates message copies to \( v_j \). The specific algorithm process is shown in Algorithm 2.

If source node \( v_i \) encounters \( v_j \), first update their delivery probability \( DP_{ij} \) and \( MSD \) list, calculate the messages carried by \( v_j \) but not carried by \( v_i \) through \( SV_i \cap SV_j \), and store them in \( SV \). \( v_i \) and \( v_j \) delete the message recorded in the \( MSD \) list. If \( v_i \) only set a connection with a node, it indicates that the nodes near \( v_i \) are relatively sparse, and the chance of encountering other nodes is very slim. To ensure the successful delivery of the message, the message is directly forwarded to \( v_j \). Otherwise, all messages in \( SV \) are traversed, denoted by \( m_k \). If \( v_j \) is the destination node of \( m_k \), the message is directly forwarded to \( v_j \). Take out messages with more than 1 copy, and calculate \( NIP_{id} \) and \( NIP_{jd} \) according to formula (6). If both \( NIP_{id} \) and \( NIP_{jd} \) are larger than the NIP’s threshold \( \theta_i \) or \( NIP_{jd} \) and \( NIP_{id} \) are both less than \( \theta_j \), then use \( DP \) to determine whether to forward messages. If \( DP_{jd} \) is greater than the \( DP_{id} \), calculate the distribution of message copies according to formula (16) and formula
Input:

- $v_i$, the node that is ready to receive the message
- $L_i(m_k)$, number of copies of $m_k$ that need to forward to $v_i$
- $forwardList$, set of messages need to forward to $v_i$

1: if $forwardList! = Null$ then
2: $forwardList.sort(ascending, TTL)$
3: for each message in $forwardList$ do
4: if $v_i$'s remaining buffer < $L_i(m_k) \times size(m_k)$ then
5: delete the messages with smallest $MSV$ in $v_j$ until has enough buffer size to receive $m_k$
6: end if
7: forward $m_k$ to $v_i$
8: end for
9: end if

Algorithm 1: Pseudocode for buffer management.

Input:

- $V = \{v_i | 1 \leq i \leq n\}$, set of nodes in the network
- $\Theta_{threshold}$, the threshold of node interest preference ($NIP$)
- $Con(i)$, the number of connections of $v_i$
- $SV_i$, set of messages carried by $v_i$
- $SV_j$, set of messages carried by $v_j$

1: if $v_i$ encounters $v_j$ then
2: $SV = SV_i \cap SV_j$, update $DP(i, j)$ and $MSD$
3: delete messages in $v_i$ and $v_j$ according to their own $MSD$
4: if $Con(i) = 1$ then
5: for each message in $SV$ do
6: $v_i$ directly forwards $m_k$ to $v_j$
7: end for
8: end if
9: for each message in $SV$ do
10: if $NIP_{id} > \Theta_i \& \& NIP_{jd} > \Theta_j \& \& DP_{id} > DP_{jd}$ then
11: calculate $L^\text{new}_{ij}(m_k)$ according to formulas (16) and (17)
12: forward $L^\text{new}_{ij}(m_k)$ copies of $m_k$ to $forwardList$
13: end if
14: else if $NIP_{id} < \Theta_i \& \& NIP_{jd} > \Theta_j$ then
15: calculate $L^\text{new}_{ij}(m_k)$ according to formulas (16) and (17)
16: forward $L^\text{new}_{ij}(m_k)$ copies of $m_k$ to $forwardList$
17: end if
18: else if $NIP_{id} < \Theta_i \& \& NIP_{jd} < \Theta_j \& \& DP_{id} > DP_{jd}$ then
19: calculate $L^\text{new}_{ij}(m_k)$ according to formulas (16) and (17)
20: forward $L^\text{new}_{ij}(m_k)$ copies of $m_k$ to $forwardList$
21: end else
22: end for
23: end if

Algorithm 2: Pseudocode for NIP-PSW’s Spray phase.
In the whole algorithm, when calculating \( SV_j \), it is necessary to traverse the messages carried by the nodes, so the time complexity is \( O(n) \). Then, each message in the \( SV_i \) needs to be traversed, so the time complexity is \( O(n) \). In addition, the calculation of \( NIP, DP \), and the number of message copies in the algorithm are all constant levels, so the time complexity of the whole algorithm is \( O(n) \).

### 4.2. Wait Phase

In Spray-and-Wait algorithm, when the number of message copies is 1, the node enters the wait phase. Only when the destination node is encountered, the message will be forwarded to the destination node. In this process, a high delay is caused, and the chances of encountering other nodes are not fully utilized. Therefore, to reduce network delay and increase the delivery rate, this paper uses the \( NIS \) and \( DP \) to improve the wait phase. The specific algorithm is shown in Algorithm 3.

In the wait phase, when two nodes encounter, first update their \( DP \) and the \( MSD \) list, and use \( SV_i \cap SV_j \) to calculate the messages carried by \( v_i \) but not carried by \( v_j \) and store them in \( SV \). \( v_i \) and \( v_j \) delete the messages recorded in the \( MSD \). Then, traverse all the messages in \( SV \), denoted by \( m_k \). If \( v_j \) is the destination node \( v_d \) of \( m_k \), the messages are directly forwarded to \( v_j \). Take out the messages with only 1 copy. If the \( NIS \) between \( v_i \) and \( v_d \) is greater than the threshold and the \( DP_{vd} \) is greater than the \( DP_{id} \), then the message will be stored in the forwarding queue \( forwardList \) and wait for the forwarding of the message. During the message forwarding process, if the buffer size of the node \( v_i \) is not enough to receive the message \( m_k \) in the \( forwardList \), the message with the smallest \( MSV \) in the buffer of the \( v_i \) will be deleted, until there is enough buffer to receive the message \( m_k \). When calculating \( SV_j \), the messages carried by the nodes will be traversed, so the time complexity is \( O(n) \). Then, the messages in the \( SV_j \) will be traversed, so the time complexity is \( O(n) \). In addition, when calculating \( NIS \) and \( DP \), the time complexity is \( O(1) \), so the time complexity of the whole algorithm is \( O(n) \).

### 5. Simulation

In this paper, the ONE [21] is used to create the scenario and simulate the NIP-PSW algorithm. We take Helsinki Map [22] as the simulation area and divide it into 5 * 6 small areas on average. We evaluate the algorithms from four metrics: delivery rate, average delay, average hop count, and overhead. And we use three algorithms: Spray-and-Wait, Prophet, MASS [20], SCSS [14], and DPN-ASW [19] to
Figure 3: Continued.
compare with NIP-PSW. The main parameters of the simulation are listed in Table 3.

5.1. Impact of Simulation Time. Figure 3 shows the performance of NIP-PSW, MASS, Spray-and-Wait, and Prophet algorithms at different simulation times. Figure 3(a) shows the impact of simulation time on the delivery rate. From the figure, we can see that the delivery rate of each algorithm increases significantly as the simulation time increases. Among them, NIP-PSW has the highest delivery probability, up to 90%. NIP-PSW, respectively, outperforms Spray-and-Wait, MASS, Prophet, SCSS, and DPN-ASW by 10.45%, 9.2%, 39.9%, 8%, and 21.6% on average. This is because NIP-PSW forwards the message to a node that is closer to the destination node’s interest and has a larger delivery probability. Then, considering the node’s ability, NIP-PSW allocates the number of message copies dynamically, which ensures the node can deliver the message to the destination node. In addition, in the wait phase, opportunities for encountering other nodes are also taken into consideration. When encountering a node with a higher interest similarity and delivery probability to the destination node, the message is forwarded again. The delivery rate of the Prophet is relatively low because there is no restriction on message copy.
Figure 4: Continued.
In the process of replicating and forwarding messages, nodes will quickly fall into a congested state, causing many messages to be discarded.

From Figure 3(b), we can see that the average delay of NIP-PSW is the lowest. This is because NIP-PSW selects nodes that are easier to become friends with the destination node to carry messages. The closer the interest is, the easier it is to encounter, which speeds up the encounter between the relay node and destination node and reduces the network delay. From Figure 3(c), we can see that the overhead of NIP-PSW is also relatively low, and the average overhead value remains at 14, which is basically the same as MASS. Spray-and-Wait and SCSS have the lowest overhead. In Spray-and-Wait algorithm, the number of message copies is limited, and then, in the spray phase, only one copy of the message is forwarded to the relay node, and the relay node will enter wait phase and will not forward messages to other nodes until it encounters the destination node. However, NIP-PSW will also forward messages in wait phase, so the overhead will be slightly higher. In Figure 3(d), the average hop count of NIP-PSW, MASS, Prophet, Spray-and-Wait, SCSS, and DPN-ASW are 2.5, 2.8, 2.3, 1.8, 2.4, and 2.5, respectively. The hop count of NIP-PSW is not high. The reason for the low hop count of Spray-and-Wait is that all relay nodes will enter the waiting phase after receiving the message from the source node and only wait for the encounter with the destination node, so the hop count will remain within two hops.

5.2. Impact of Buffer Size. Figure 4 shows four algorithms under different buffer sizes. As shown in Figure 4(a), as the buffer size increases, the delivery rate gradually increases. This is because, as the node buffer size increases, more messages are stored, the probability of discarding messages decreases, and the node’s ability to handle messages increases. NIP-PSW has an excellent delivery rate, always staying above 80%, which is 13.8%, 14.3%, 45.5%, 12.3%, and 23.5% higher than Spray-and-Wait, MASS, Prophet, SCSS, and DPN-ASW on average. This is due to the fact that NIP-PSW preferentially selects nodes with similar interests to the destination node.

Figure 4(b) shows the average latency of these four algorithms under different buffer sizes. From the figure, we can see that the average delay of NIP-PSW becomes lower and lower as the buffer size increases. At first, when the buffer size is small, the delay of NIP-PSW is higher than Spray-and-Wait and MASS. This is because when NIP-PSW selects a relay node, it needs to calculate the node interest preference, interest similarity, and delivery probability from the node’s historical information. The storage and calculation of this information will take up the node’s buffer and consume time. So, when the buffer size is small, the latency is slightly higher than these two algorithms. However, when the buffer size gradually increases, the delay is also decreased, and after 16 M, the NIP-PSW’s delay is the lowest.

Figure 4(c) shows the changes in the overhead of the four algorithms when the buffer size changes. The overhead of NIP-PSW is also low, with an average value of about 15, which is nearly the same as the MASS algorithm. The overhead of the Spray-and-Wait and SCSS is still the lowest. This is because they only forward messages to destination nodes in wait phase.

Figure 4(d) shows the change of the hop count of the four algorithms. The average hop count of NIP-PSW is about 2.5 hops, which is slightly higher than the SCSS algorithm. The hop count of Spray-and-Wait algorithm is maintained within two hops. This is because the source node only
allocates a copy of the message to the relay node. After receiving the message, the relay node enters the wait phase and waits to encounter the destination node. In this process, the message only passes two hops. NIP-PSW also forwards messages in the wait phase, so the hop count will increase slightly, which is reasonable.

5.3. Impact of TTL. Figure 5 shows the performance of the four algorithms under different TTL. Figure 5(a) shows the effect of TTL on the delivery rate. From the figure, we can see that the delivery rates of NIP-PSW, MASS, and Spray-and-Wait all increase with the increase of TTL, only the delivery rate of Prophet increases first and then decreases. This reason is that Prophet does not limit the number of message copies. In the beginning, as the TTL increases, the message can live to get the chance of being forwarded, so the delivery rate is improved. As the TTL becomes larger and larger, the live time of messages in the node is longer and longer. In addition, due to the replication strategy, more and more copies of messages are stored in the node, causing congestion, so the delivery rate will decrease. The delivery rate of NIP-PSW is also the highest, 12.2%, 10%, 33.5%, 9%, and 21.6% higher than Spray-and-Wait, MASS, Prophet, SCSS, and DPN-ASW on average.

As shown in Figure 5(b), with the increase of TTL, the average delay of each algorithm is increasing. Where the average delay of NIP-PSW is the lowest. This is because with the increase of TTL, messages that had no chance to be
forwarded because of the short TTL before have the opportunity to be forwarded. The reason for the lower latency of NIP-PSW is that it forwards the message to a relay node with similar interests to the destination node, so the probability of encountering the destination node will be high, and it will take less time to forward messages. In addition, messages will be forwarded in the wait phase, so the latency is low.

In Figure 5(c), with the change of TTL, the overhead of the other three algorithms has not changed much except for the Prophet’s. This is caused by network congestion in Prophet. The overhead of NIP-PSW is not high; the average value is about 15.

As shown in Figure 5(d), the average hop count of the four algorithms is relatively stable. The hop count of NIP-PSW is maintained within 2.5 hops.

6. Conclusion

In this paper, we propose a method for calculating node interest preference (NIP) based on the historical encounter nodes and the geographical areas the nodes passed by. Then, according to the source node’s situation and the buffer state of the relay node, as well as the connection time between the two, the adaptive delivery probability (DP) is defined. We combine the DP with NIP as a metric for selecting the relay node. To allocate message copy according to the node’s own capabilities and ensure that each node can handle these messages, the message copy is allocated dynamically by the DP. In the wait phase, the concept of node interest similarity (NIS) is presented. The node forwards the message to the node with the larger NIS of the destination node. In addition, we also introduce the acknowledgment (ACK) mechanism and message storage value (MSV) to manage the buffer of nodes and alleviate the impact of network congestion on the algorithm to a certain extent. The simulation results show that NIP-PSW is superior in delivery rate and delay, while the overhead and average hop count are not high, and the overall performance of the algorithm is also better.

In future work, we plan to divide the area more finely by analyzing the historical data information of the nodes. In addition, more efficient methods can be used to calculate the interest preferences of the nodes to ensure a high probability of encounter between the selected relay node and the destination node. At the same time, an improved buffer management strategy can be used to further increase the delivery rate and reduce network overhead.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (nos. 61702210, 61672257, and 61772377).

References

[1] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, and H. Weiss, Delay Tolerant Network Architecture, 2004.
[2] K. R. Fall, “A delay-tolerant network architecture for challenged internets, in Proceedings of the ACM SIGCOMM,” in Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, New York, NY, United States, 2003.
[3] M. Ramadas, S. Burleigh, and S. Farrell, Rfc 5326, Licklider Transmission Protocol Specification, 2008.
[4] Y. Kuronuma, H. Suzuki, and A. Koyama, “An adaptive dtm routing protocol considering replication state,” in 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA), Taipei, Taiwan, 2017.
[5] A. Vahdat and D. Becker, Epidemic routing for partially-connected ad hoc networks, Handbook of Systemic Autoimmune Diseases, 2000.
[6] A. Lindgren, A. Doria, and O. Schelén, “Probabilistic routing in intermittently connected networks,” International Workshop on Service Assurance with Partial and Intermittent Resources, vol. 7, 2004.
[7] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Efficient routing in intermittently connected mobile networks: the single-copy case,” IEEE/ACM Transactions on Networking, vol. 16, no. 1, pp. 63–76, 2008.
[8] S. P. Palazzo, L. G. Galluccio, M. D. de Amorim Amorim, A. F. Fladenmuller, and M. B. Boc, “Price: hybrid geographic and contact-based forwarding in delay-tolerant networks,” 2011.
[9] E. M. Daly and M. Haahr, “Social network analysis for routing in disconnected delay-tolerant manets,” in Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc, Montreal, Quebec, Canada, 2007.
[10] H. C. Wang Guizhu and W. Bingting, “Probabilistic routing algorithm considering connection time in dtn,” Computer Applications, vol. 31, no. 5, pp. 1170–1172, 2011.
[11] E. H. Kim, J. C. Nam, J. I. Choi, and Y. Z. Cho, “Probability-based spray and wait protocol in delay tolerant networks,” in International Conference on Information Networking, Phuket, Thailand, 2014.
[12] C. Dai, J. Li, L. You, and J. Xu, “An improved spray and wait routing algorithm based on node performance in delay tolerant networks,” Information Technology Journal, vol. 12, no. 19, pp. 5036–5043, 2013.
[13] L. Wu, S. Cao, Y. Chen, J. Cui, and Y. Chang, “An adaptive multiple spray-and-wait routing algorithm based on social circles in delay tolerant networks,” Computer Networks, vol. 189, no. 12, article 107901, 2021.
[14] J. X. Jia, G. Z. Liu, and X. U. Ming, “Probability routing algorithm in dtm based on time and space and sociality,” Computer Science, vol. 43, 2016.
[15] P. Pirozmand, G. Wu, B. Jedari, and F. Xia, “Human mobility in opportunistic networks: characteristics, models and prediction methods,” Journal of Network Computer Applications, vol. 42, pp. 45–58, 2014.
[16] K. Liu, Z. Chen, J. Wu, and L. Wang, “Fcns: a fuzzy routing-forwarding algorithm exploiting comprehensive node similarity in opportunistic social networks,” Symmetry, vol. 10, no. 8, 2018.

[17] J. Wu, K. Ye, Y. Zhu, and L. Liu, “New spray and wait protocol with node activity in heterogeneous delay tolerant networks,” The Journal of China Universities of Posts and Telecommunications, vol. 22, 2015.

[18] J. Cui, S. Cao, Y. Chang, L. Wu, 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.

[19] R. W. C. S. Wei Zhuhua, L. Dan, and K. Shaoping, “An adaptive spray and wait routing algorithm based on delivery probability of node in dtn,” Journal of Wuhan University: Science Edition, vol. 65, no. 6, p. 9, 2019.

[20] H. Hu, D. W. Yang, H. Wang, and J. M. Kuang, “A motion awareness based routing algorithm for delay tolerant network,” Beijing Daxue Xuebao/Transaction of Beijing Institute of Technology, vol. 39, no. 1, pp. 68–74, 2019.

[21] A. Keränen, J. Ott, and T. Kärkkäinen, “The one simulator for dtn protocol evaluation,” in International Conference on Simulation Tools Techniques, Rome, Italy, 2009.

[22] E. Bulut and B. K. Szymanski, “Exploiting friendship relations for efficient routing in mobile social networks,” Parallel Distributed Systems IEEE Transactions on, vol. 23, no. 12, pp. 2254–2265, 2012.