Computer Networks 189 (2021) 107901

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

An adaptive multiple spray-and-wait routing algorithm based on social circles in delay tolerant networks

Libing Wu [a,b,c,*], Shuqin Cao [a], Yanjiao Chen [a,**], Jianqun Cui [d], Yanan Chang [d]

[a] School of Computer Science, Wuhan University, Wuhan, 430072, China
[b] School of Cyber Science and Engineering, Wuhan University, China
[c] Shenzhen Research Institute of Wuhan University, China
[d] School of Computer, Central China Normal University, China

ARTICLE INFO

Keywords:
Adaptive routing algorithm
Delay tolerant networks
Spray-and-wait
Social circles

ABSTRACT

In delay tolerant networks (DTN), the social attributes of nodes show long-term stability, which can be leveraged for more effective routing. In this paper, we first present a novel way of constructing social circles based on the node clustering phenomena in DTN. Then, considering that the forwarding capability of nodes is significantly different, we propose a spray strategy based on social circles (named SC-SS) to improve the spray-and-wait routing algorithm. SC-SS selects the next hop based on the social circle of nodes in the spray phase. Instead of fixing the initial number of copies, we design an adaptive multiple spray-and-wait routing algorithm based on social circles (named SC-AMSW) to further improve the performance of SC-SS. SC-AMSW selectively sprays messages multiple times in the wait phase and determines an appropriate number of redundant message copies based on delivery predictability. We conduct extensive simulations to confirm the effectiveness of our proposed routing algorithms in DTN.

1. Introduction

Traditional communication models of computer networks are usually based on some inherent assumptions, such as short round trip delay and a stable end-to-end link between the source and the destination, which may not stand in harsh environment where the network topology is unstable and highly dynamic. Delay tolerant networks (DTN) utilize encounter opportunities from node movement to realize communications [1–3] based on a Store-Carry-Forward scheme, which forwards messages hop by hop. As shown in Fig. 1, at t_1, the source v_s intends to forward a message to v_d. However, the end-to-end path between v_s and v_d is not available. Node v_s first forwards the message to its neighboring node v_a. v_a stores the message in its local cache and waits for a suitable forwarding opportunity. At t_2, v_a forwards the message to the encountered node v_e. At t_3, v_e encounters v_d and forwards the message to v_d, which ends the message forwarding process. The design of an efficient routing algorithm is the key and challenging issue in DTN.

In recent years, many DTN routing algorithms have been proposed. For example, Epidemic [4] is a typical flooding-based routing algorithm. Every node forwards messages to all encountered nodes to maximize the delivery rate. Excessive redundant copies consume network resources and may easily lead to network congestion, so Epidemic is less practical. Prophet [5] assumed that two nodes that often encounter in the past are likely to encounter again in the future. However, Prophet's overhead is still high since the maximum number of copies of a message is unlimited. spray-and-wait [6] is proposed to reduce the overhead. The source node of a message primitively carries a fixed number of copies. The forwarding node will allocate half of the copies to the nodes it encounters. When the number of copies held by a node is 1, the node enters the wait phase and performs a direct transmission until it moves to the destination's communication range. To improve the performance of Spray-and-Wait, Yang et al. [7] proposed GBAS, which selects the next hop based on the activity range of nodes in the spray phase and adopts multiple spray strategy in the wait phase. Since GBAS does not selectively spray the messages multiple times in the wait phase, the overhead is still slightly higher. Moreover, most of the existing routing algorithms do not delete the useless copies that have been successfully delivered to the destination. If these nodes in the network continue to perform forwarding, it may lead to a waste of network resources.

* Corresponding author at: School of Computer Science, Wuhan University, Wuhan, 430072, China.
** Corresponding author.
E-mail addresses: wu@whu.edu.cn (L. Wu), shuqinccao@163.com (S. Cao), chenyanjiao@zju.edu.cn (Y. Chen), jqcui@126.com (J. Cui), ynchang@mail.ccnu.edu.cn (Y. Chang).

https://doi.org/10.1016/j.comnet.2021.107901
Received 11 September 2020; Received in revised form 24 September 2020; Accepted 29 January 2021
Available online 12 February 2021

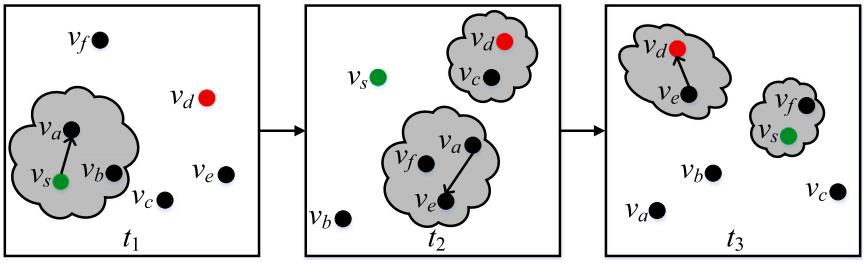

Fig. 1. Message forwarding process in DTN.

Most mobile devices in DTN are carried by humans, so applying social attributes in DTN routing algorithms has raised great concerns among researchers in recent years. SRAMSW [8] selects relay nodes based on the social relationship to improve the chance of message arriving at the destination. In [9], a routing algorithm is proposed considering contact duration and message size, which classifies node contacts into pass-by contacts and social contacts. Small messages are forwarded at pass-by contacts, which generally have short contact duration. Social contacts are used for forwarding large messages. Roy et al. [10] exploited social attributes to design heuristic relay selection strategies. However, the above existing DTN routing algorithms did not adaptively adjust the number of copies according to the network environment. Furthermore, the social attributes considered in routing still need to be further studied.

In this paper, to address the problems mentioned above, we propose a novel routing algorithm named SC-SS for DTN, which leverages social circles to improve the spray strategy. SC-SS makes full use of the social attributes of nodes to achieve high performance in the spray phase and the wait phase. To further improve the performance of SC-SS, instead of a fixed initial number of copies, we present an adaptive multiple spray-and-wait routing algorithm named SC-AMSW. Moreover, we adopt an acknowledgment (ACK) mechanism to reduce redundant copies that have been successfully delivered. The main contributions are summarized as follows.

- We propose a novel way of constructing social circles based on the fact that nodes often show clustering phenomena in most application scenarios of delay tolerant networks.
- We propose a spray strategy based on social circles to improve the spray phase of spray-and-wait. SC-SS adopts different relay node selection strategies according to whether the node carrying messages is in the social circle of the destination, which makes message delivery more effective.
- We propose an adaptive multiple spray-and-wait routing algorithm SC-AMSW, which selectively sprays messages with one copy multiple times to improve the wait phase of SC-SS to adapt to the dynamically changing network environment.

The rest of the paper is organized as follows. We discuss the related works in Section 2. Section 3 gives a brief overview of SC-AMSW. Section 4 describes how to construct the social circles of nodes and gives the details of SC-SS. Section 5 presents the details of SC-AMSW, which improves the wait phase of SC-SS. The details of the ACK mechanism are given in 6. In Section 7, we demonstrate the efficiency of SC-SS and SC-AMSW by comparison with existing algorithms from four aspects. In Section 8, we summarize the paper and point out the directions for future works.

2. Related work

The design of an efficient routing algorithm is an essential and fundamental problem in DTN. A routing algorithm aims to consume fewer network resources and time to deliver messages from a given source to the destination successfully. Recently, a growing number of studies have been done for DTN routing algorithms. According to whether the additional assistance information is needed in the forwarding process, DTN routing algorithms can be divided into zero information and auxiliary information [11]. The zero information routing algorithms do not use any additional information to assist decision-making. For example, Grossglauser et al. [12] proposed Direct Delivery (DD), in which the source node does not forward messages to any relay nodes until it encounters the destination node. Since there is no redundant relay operation, the algorithm has the lowest network overhead, but its performance in the delivery rate and the average delay is poor. Epidemic [4] mimics the spread of infectious viruses in the biological environment and forwards messages to every node it encounters, which is the other extreme of DD. However, the characteristics of blindly spreading messages will lead to buffer overflow in the case of limited resources. The authors in [6] proposed a routing algorithm with a fixed number of copies, called spray-and-wait, to reduce the average delay and the network overhead. However, the zero information routing algorithms mentioned above do not take into account the difference between nodes. They cannot make a reasonable routing strategy based on the difference, resulting in low transmission efficiency.

Factors such as node energy, delivery predictability, and encounter frequency have essential influences on the forwarding strategy. Therefore, researchers have proposed many auxiliary information routing algorithms to improve the performance of spray-and-wait, including dynamically allocating the number of message copies, optimizing the selection of relay candidates, and improving direct transmission strategy in wait phase. Since large-size messages consume more energy, the study of [13] limited the resource consumption of messages and forwarded messages to those nodes with high delivery predictability. Derakhshanfard et al. [14] proposed Sharing spray-and-wait, which calculates the number of copies that a node can allocate to others and selects relay nodes based on delivery predictability and message carrying time. Spaho et al. [15] enhanced spray-and-wait by changing the number of copies the sender sprays to the receiver. Yang et al. [7] proposed GBAS, which selects the next hop based on the activity range of nodes in spray phase and adopts multiple spray strategy in wait phase. Since GBAS does not selectively spray the message multiple times in wait phase, the overhead is still slightly higher. Moreover, we need to discard these redundant copies which have been successfully forwarded to the destination. Because when a message is successfully forwarded to the destination node, many copies of the message still exist in some relay nodes, and these copies are stored in the limited cache resources of these relay nodes, which makes the resource utilization of these nodes' cache much low.

Considering the social attributes of nodes show long-term stability, researchers have proposed many social-based routing algorithms. The works in [16] analyzed the influence of centrality and community social metrics on forwarding. They proposed a social-based forwarding scheme, called Bubble Rap, to select community members of destination and high centrality nodes as the next hops. Jain et al. [17] proposed an improved Bubble Rap, which integrates the reputation-based and incentive-based approaches to motivate selfish nodes to participate in cooperative forwarding. The study of [18] built a social

map to record social structure around nodes, and then applied the map for lightweight routing. In [19], the authors constructed a home-aware community model and then proposed a community-aware routing algorithm. Yu et al. [20] proposed an algorithm to predict mobility probabilities based on forwarding collaboration relationship and social relationship between nodes. The authors in [21] introduced a metric called social energy to evaluate the ability of nodes to forward messages to others, and then proposed a routing algorithm based on social energy. Wei et al. [22] developed a distributed community detection algorithm to dynamically discover bridge nodes and overlapping communities for routing algorithm design. To improve the performance of the forwarding scheme, the study of [23] designed message forwarding metrics to predict the contact capability of mobile nodes. Boc et al. [24] proposed an agenda-based greedy routing algorithm that builds an agenda of locations and contacts based on the history information. The authors in [25] designed criteria to control the trigger of inter-session network coding policies, based on current network load and the social relationships and buffer size of nodes. Then the criteria were used on the simBet routing algorithm to bring gains. In [26], the authors introduced SPARE, which combines four factors (historical encounter information, encounter probability, resources effective consumption, and the number of messages held by nodes) to solve the resource utilization problem. The above routing algorithms do not adaptively adjust the number of message copies according to the network environment.

This paper presented an adaptive multiple spray-and-wait routing algorithm based on social circles, which consists of two phases: spray phase and wait phase. In the spray phase, the algorithm selects relay nodes based on social metrics; in the wait phase, the algorithm adaptively adjusts the number of message copies carried by nodes to improve the delivery rate. Besides, an acknowledgment mechanism is used to improve the utilization of network resources.

3. Overview

In this section, we give a brief overview of the SC-AMSW (adaptive multiple spray-and-wait routing algorithm based on social circles). As shown in Fig. 2, SC-AMSW is divided into spray phase and wait phase. In the spray phase, SC-SS (spray strategy based on social circles) is implemented to select suitable relay nodes. SC-SS constructs the social circle of nodes based on similarity degree and then adopts different relay selection schemes according to whether the node carrying messages is in the destination's social circle, avoiding spraying messages to all encountered nodes. If the node carrying messages is in the social circle of the destination, SC-SS will transmit messages within the social circle of the destination node and adopt DP-RFS (routing forwarding strategy based on delivery predictability) scheme to select relay nodes; if not, SC-SS will select social circle members of the destination or adopt GI-RFS (routing forwarding strategy based on geographical information) scheme to select the nodes with larger transmission utility as the next hops. In this way, messages can be forwarded to the destination's activity range as soon as possible. In wait phase, SC-AMSW adopts DP-AMSS (adaptive multiple spray strategy based on delivery predictability) to spray messages with one copy multiple times selectively. DP-AMSS adaptively sprays the appropriate number of redundant copies based on delivery predictability, which will not bring huge pressure to the cache while improving the routing algorithm's performance in delivery rate and average delay.

In order to efficiently improve the utilization of network resources, SC-AMSW uses an ACK (acknowledgment) mechanism to decrease the number of useless redundant copies. Specifically, when a message is successfully delivered to the destination, the destination generates an ACK message indicating that the message has been successfully forwarded to the destination. When two nodes encounter and form a connected communication area, they exchange each other's ACK message list to determine which messages in the local cache have been successfully forwarded to the destination. Then these two nodes delete these messages from the local cache to reduce unnecessary forwarding operations and free up more cache space for other messages.

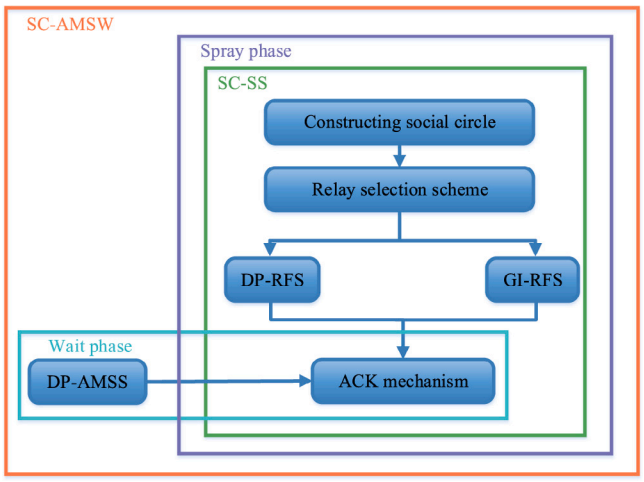

Fig. 2. Overview of SC-AMSW.

Besides, the summary of important used variables in the paper is listed as Table 1.

4. A spray strategy based on social circles

In this section, we proposed constructing social circles and then presented a routing algorithm called spray strategy based on social circles (SC-SS).

4.1. A method of constructing social circles

It is the latest trend in the DTN research field to introduce the relevant theoretical results of social networks to the routing algorithm. Mobile devices with communication functions are mostly used by people in many application scenarios of DTN, so social characteristics are also exhibited in DTN. Although the network topology is relatively unpredictable, the movement mode of humans is not completely random. For example, nodes with similar social characteristics (such as hobbies, interests, research topics, etc.) usually come together and form a social circle. Compared with the nodes outside the social circle, nodes in the same social circle are more closely connected, encounter more frequently, and are far more likely to deliver messages to each other successfully. If we can accurately construct the destination node's social circle, we can forward messages within the social circle of the destination node. Furthermore, the social attributes of nodes are relatively stable, so making reasonable use of the social attributes of nodes can achieve better routing performance.

Based on the above analysis, we presented a method of constructing social circles based on the theory put forward by sociologists that two nodes with many common acquaintances are more likely to become acquaintances. Sociologists have long found that the degree of transitivity is high in social networks. For example, if two people have one or more common friends, these two individuals are more likely to become friends. This phenomenon is called "clustering," that is, nodes with strong social relationships usually come together and form a social circle. Our method measures the closeness of social relationships between nodes according to similarity degree.

Definition 1 (*Similarity Degree (SD)*)**.** The similarity degree between node v_i and node v_j, say $SD_{(i,j)}$, is the count of common encountered nodes between v_i and v_j. $SD_{(i,j)}$ can be calculated by:

$$SD_{(i,j)} = \left| E_i \cap E_j \right|, \tag{1}$$

Table 1
List of notations.

Notation	Explanation
V	Set of nodes in the network
$SD_{(i,j)}$	Similarity degree between node v_i and node v_j
E_i	Set of encounter nodes of node v_i
SCN_i	Social circle of node v_i
$SCN_{threshold}$	Threshold of SCN
SV_i	Set of messages carried by node v_i
$\overline{SV_j}$	Complement of set SV_j
$P_{(i,j)}$	Probability of node v_i successfully forwarding message to node v_j
$P_{threshold}$	Threshold of delivery predictability
$L_i(m_k)$	Number of copies of message m_k carried by node v_i
$nNodes$	Number of neighbor nodes in node v_i's communication range
$forwardList$	Set of messages that node v_i need to forward to node v_j
V_i	Velocity vector of node v_i
$\|V_i\|$	Moving speed of node v_i
$TUoN_i$	Transmission utility of node v_i
RS_i	Remaining cache size of node v_i
S_k	Size of message m_k
$\vartheta_{(i,d)}$	Angle between the direction of motion of node v_i and the direction of motion of the destination node v_d
$D_{(i,d)}$	Distance of node v_i from the destination node v_d
L	The initial number of copies
NRC	Number of the re-sprayed copies
$ST_i(m_k)$	Number of times node v_i has re-sprayed message m_k
AMI_i	ACK message list of node v_i

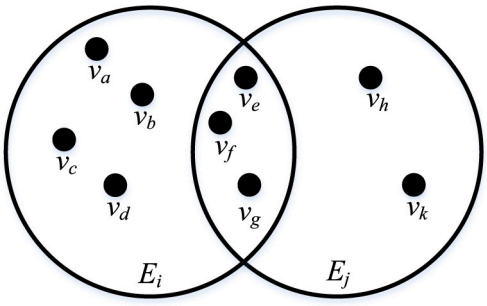

Fig. 3. Node v_i encounters node v_j.

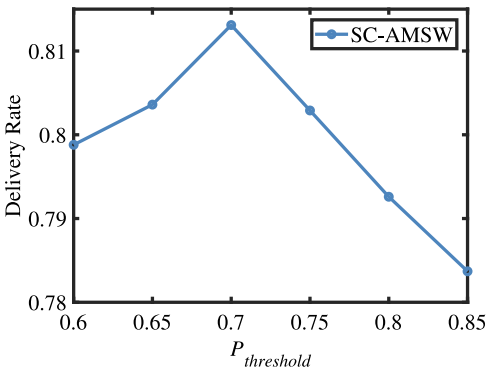

Fig. 4. Message forwarding process in DTN.

where $E_i = \{v_k | n_{ik} \neq 0, 1 \leq k \leq n\}$ is the set of encounter nodes of node v_i, n_{ik} (its initial value is 0) represents the encounter times between v_i and v_j, n_{ik} is added by 1 when v_i encounters v_k, n is the total number of nodes, and E_j is the set of encountered nodes of v_j.

Let us use Fig. 3 as an example to explain the specific computation process of $SD_{(i,j)}$. As shown in Fig. 3, $E_i = \{v_a, v_b, v_c, v_d, v_e, v_f, v_g\}$, $E_j = \{v_e, v_f, v_g, v_h, v_k\}$, it can be calculated that $SD_{(i,j)} = \left|E_i \cap E_j\right| = \left|\{v_e, v_f, v_g\}\right| = 3$.

We use the similarity degree to estimate the closeness of social relationships between nodes. If the previously encountered nodes of two nodes almost coincide, we can infer that these two nodes have many common intimate nodes and are far more likely to become acquaintances. We should add them to each other's social circle.

Definition 2 (*Social Circle of Node (SCN)*). The social circle of node v_i, say SCN_i, is the set of nodes whose similarity degree with node v_i is greater than a certain threshold $SCN_{threshold}$. The construction process of SCN_i is shown in Algorithm 1.

Algorithm 1 The Construction Process of SCN_i

Input: $SCN_{threshold}$, threshold of SCN
$V = \{v_i | 1 < i < n\}$, set of nodes in the network
$E_i = \{v_k | n_{ik} \neq 0, 1 \leq k \leq n\}$, set of encounter nodes of node v_i
$E_j = \{v_k | n_{jk} \neq 0, 1 \leq k \leq n\}$, set of encounter nodes of node v_j
Initialize $SCN_i = Null$, the SCN of node v_i

```
if v_i encounters v_j and v_j ∉ E_i then
    E_i = E_i ∪ {v_j}
    if v_j ∉ SCN_i then
        SD_(i,j) = |E_i ∩ E_j|
        if SD_(i,j) > SCN_threshold then
            SCN_i = SCN_i ∪ {v_j}
        end if
    end if
end if
```

In our algorithm, each node v_i has an attribute named social circle of node (i.e., SCN_i). SCN_i is initially null. If v_i encounters v_j and v_j is not in the set of encountered nodes of v_i (i.e., E_i), we add v_j to the set E_i. Then we judge whether v_j is already included in SCN_i. If not, we use the similarity degree $SD_{(i,j)}$ to estimate the closeness of social relationships between node v_i and v_j. If $SD_{(i,j)}$ is greater than the threshold $SCN_{threshold}$, We add v_j to SCN_i. In summary, if node v_i encounters other nodes, we need to update SCN_i based on similarity degree.

Since there are no loop and recursive statements, the time complexity of algorithm 1 is O(1).

4.2. The details of SC-SS

Since resources such as energy, cache, and bandwidth of a node are very limited, and the ability of nodes to deliver messages to the

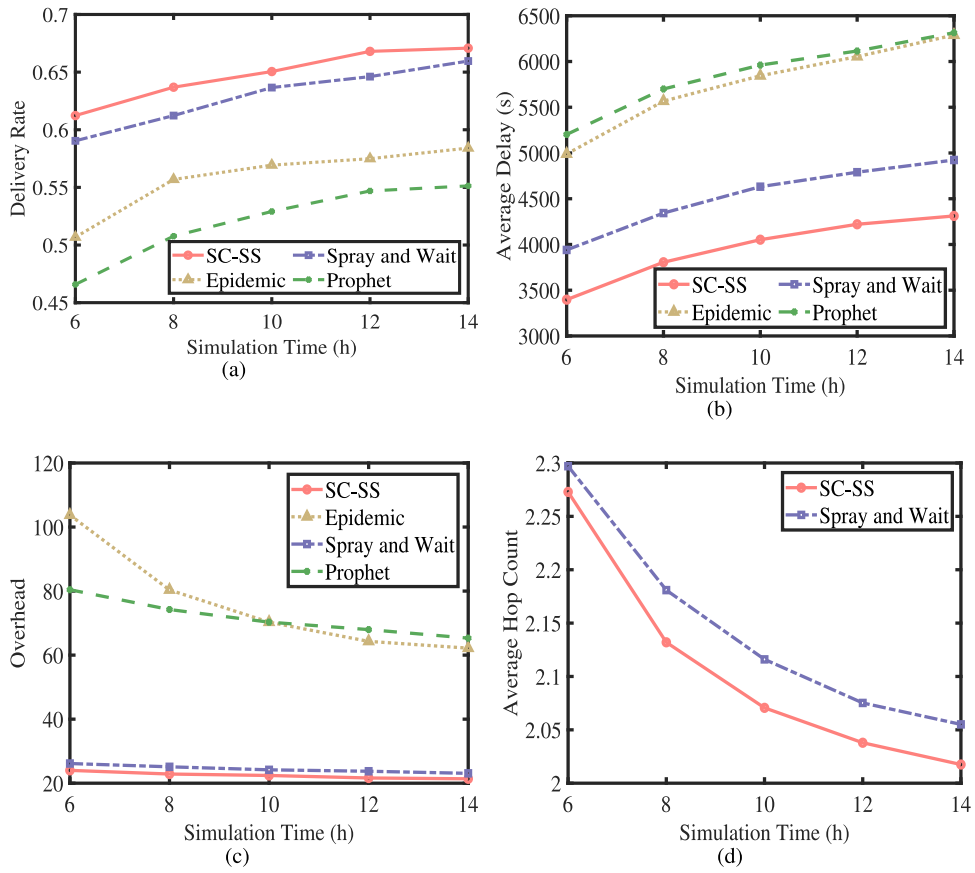

Fig. 5. Comparison of various algorithms under different simulation time.

destination node between nodes is significantly different, we should select suitable relay nodes to avoid blindly spraying messages to all encountered nodes, thereby reducing low efficient relay forwarding times and making message delivery more efficient. For example, those nodes in the same social circle are more likely to forward messages to each other successfully. As another example, if the angle between the direction of motion of the node carrying messages and the direction of motion of the destination node is small, the node carrying messages are more likely to forward messages to the activity range of the destination node. Therefore, the relay selection scheme should be designed according to the characteristics of nodes.

Based on the above analysis, we proposed SC-SS to improve the spray phase of spray-and-wait, in which every node allocates half of the copies to the nodes it encounters. In the spray phase, SC-SS adopts different relay node selection strategies according to whether the node carrying messages is in the destination's social circle, which makes message delivery more effective. Compared with the nodes outside the social circle, nodes in the same social circle are more closely connected, encounter more frequently, and are far more likely to deliver messages to each other successfully. If the node carrying messages is in the social circle of the destination, SC-SS will forward messages within the social circle of the destination node and adopt a routing forwarding strategy based on delivery predictability (DP-RFS); if the node carrying messages is not in the social circle of the destination, SC-SS will deliver messages to the nodes in the social circle of the destination node, or adopt a routing forwarding strategy based on geographical information (GI-RFS). As another special case, if the count of neighbor nodes in a node's communication range is less than 2, it can be inferred that the node density in the area where the node is located is sparse, which means that communication opportunities between nodes are very precious. If the node still carries the message and continues to wait for the suitable relay nodes, it may cause the node not to encounter a suitable relay node until the end of the message's life cycle. Therefore, in this case, SC-SS directly allocates messages to the encountered node to speed up the spread of messages, regardless of whether the encountered node is a suitable relay node.

4.2.1. A routing forwarding strategy based on delivery predictability (DP-RFS)

Nodes in the same social circle have close relationships and are far more likely to deliver messages to each other successfully. Moreover, node mobility is not completely random in DTN. The two nodes often encountered in the past are highly likely to encounter again in the future. Therefore, if the node carrying messages is in the destination node's social circle, we only spread messages in the social circle of the destination node to reduce unnecessary relay forwarding times and adopt DP-RFS.

We use delivery predictability in Ref. [5] to describe the probability of a node successfully forwarding messages to the destination. $P_{(i,j)}$ represents the probability that v_i successfully forwards a given message to v_j. The specific computation of the metric is composed of the following three parts:

- Direct update: The two nodes encounter more frequently, the probability of their encounter is higher, and then the probability of v_i successfully forwarding messages to v_j is higher. When v_i encounters v_j, we update delivery predictability by (2), where $P_{init} \in [0, 1]$ represents an initialization constant.

$$P_{(i,j)} = P_{(i,j)_{old}} + (1 - P_{(i,j)_{old}}) \cdot P_{init}. \tag{2}$$

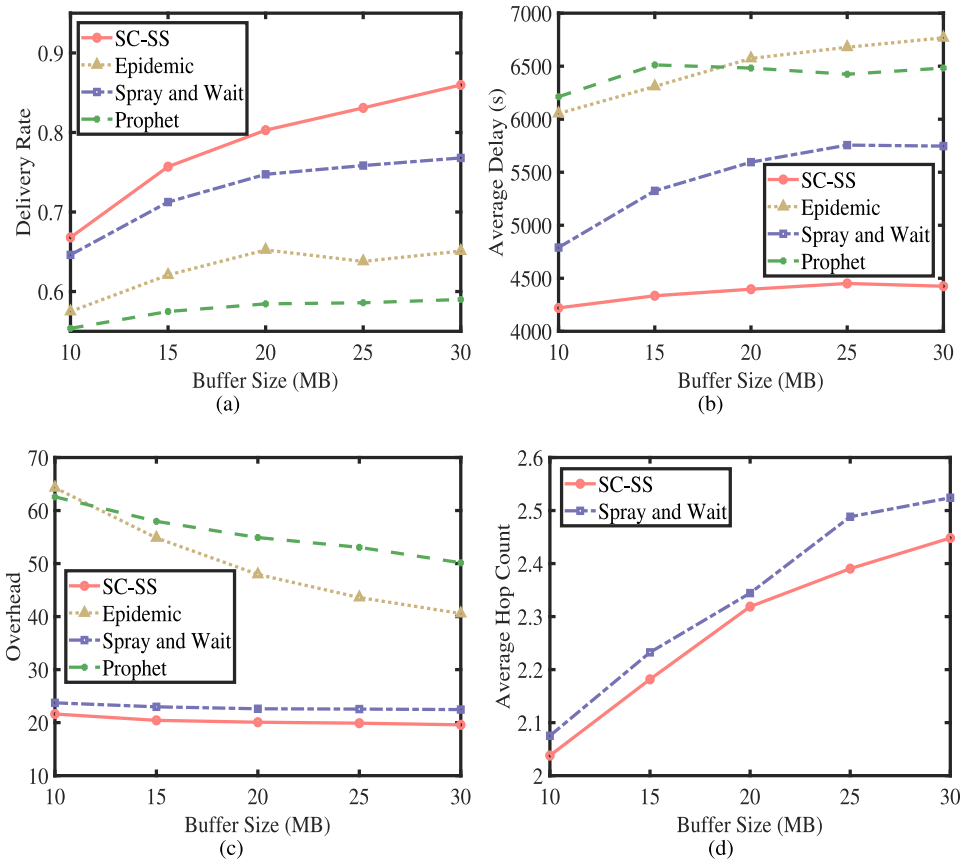

Fig. 6. Comparison of various algorithms under different buffer size.

- Aging: If two nodes do not encounter each other in a while, which means they are less likely to encounter again in the future, then we reduce the delivery predictability according to (3), where $\gamma \in [0,1)$ represents an aging constant, k refers to the count of time units that have elapsed since the last time the delivery predictability is reduced.

$$P_{(i,j)} = P_{(i,j)_{old}} \cdot \gamma^k. \tag{3}$$

- Transitive update: If v_i frequently meets v_j, v_j frequently meets v_r. Then we can infer that v_r may be a suitable relay node to send messages to v_i. So the metric is also transitive, as is shown in (4), where $\beta \in [0,1]$ stands for a scaling constant for measuring the influence of transitive property on the metric.

$$P_{(i,r)} = P_{(i,r)_{old}} + (1 - P_{(i,r)_{old}}) \cdot P_{(i,j)} \cdot P_{(j,r)} \cdot \beta. \tag{4}$$

DR-RFS selects the next hop based on the delivery predictability. If the node's delivery predictability is larger, we can infer that the node encountered the destination node more frequently in the past. Then we can consider that the node is more likely to send messages to the destination successfully in the future. The specific routing forwarding process of DR-RFS is shown in Algorithm 2.

If v_i encounters v_j, the number of copies of m_k carried by node v_i is greater than 1, and the node carrying messages is in the social circle of the destination, the $P_{(i,j)}$ and SCN_i should be updated firstly. Then, we obtain the messages carried by v_i but not carried by v_j, recorded as SV, by calculating the intersection of SV_i and $\overline{SV_j}$. We iterate over each message m_k in SV. If v_j is the destination of m_k, v_i directly forwards m_k to v_j. Otherwise, we need to compare the probability of v_i successfully forwarding m_k to v_d with that of v_j successfully forwarding m_k to v_d, if $P_{(i,d)} < P_{(j,d)}$, or $nNodes < 2$, m_k is added to $forwardList$. When all the messages in SV are traversed, we need to judge whether

Algorithm 2 Pseudo Code for DP-RFS

Input: $V = \{v_i | 1 < i < n\}$, set of nodes in the network
$L_i(m_k)$, the number of copies of m_k carried by node v_i
SCN_d, social circle of the destination
$nNodes$, the number of neighbor nodes in v_i's communication range
SV_i, set of messages carried by v_i
SV_j, set of messages carried by v_j
Initialize $forwardList = Null$

```
if v_i encounters v_j and L_i(m_k) > 1 and v_i ∈ SCN_d then
    update P_(i,j) and SCN_i
    SV = SV_i ∩ ¬SV_j
    for each message m_k in SV do
        v_d = m_k's destination
        if v_d == v_j then
            v_i directly forwards m_k to v_j
        else if P_(i,d) < P_(j,d) or nNodes < 2 then
            L_j(m_k) = ⌊L_i(m_k)/2⌋
            add L_j(m_k) copies of m_k to forwardList
        end if
    end for
    if forwardList! = Null then
        sort forwardList in ascending order of TTL
        v_i forward forwardList to v_j
    end if
end if
```

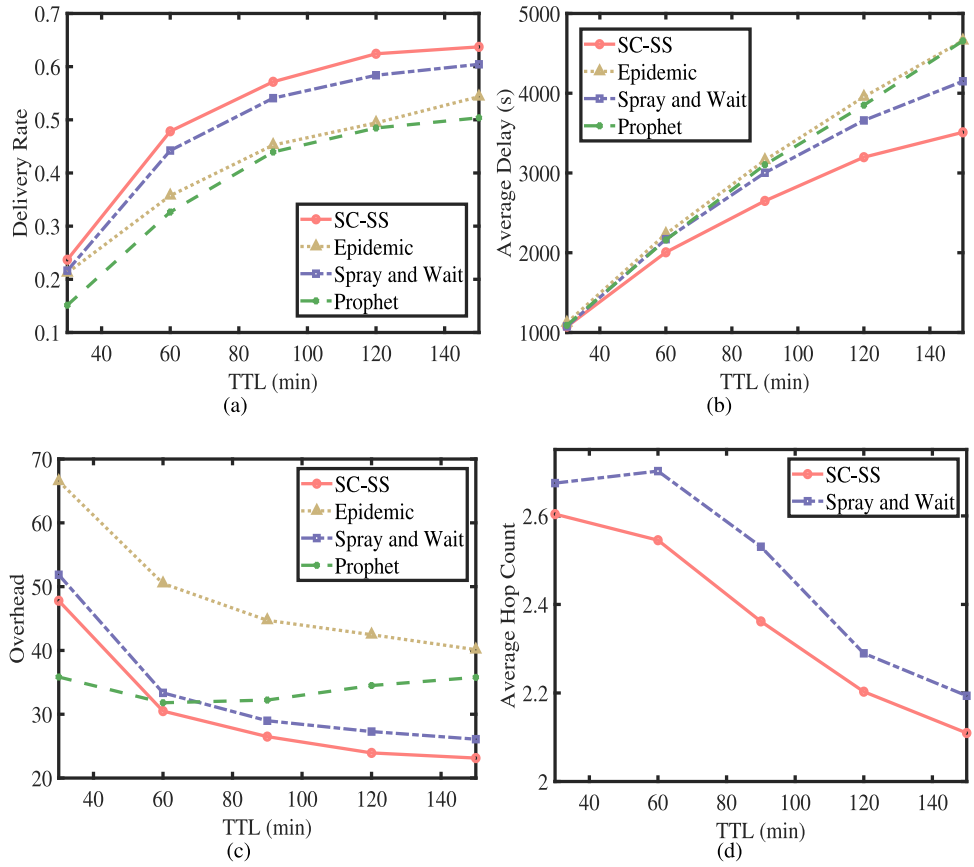

Fig. 7. Comparison of various algorithms under different TTL.

the $forwardList$ is empty. If not, we sort the messages contained in the $forwardList$ according to the remaining TTL and forward the $forwardList$.

In algorithm 2, time is mainly spent on steps 4 to 12. So the time complexity is O(m), where m is the number of messages in the network.

4.2.2. A routing forwarding strategy based on geographical information (GI-RFS)

In this subsubsection, we define a metric called transmission utility of node based on geographic information (location, motion speed, and motion direction) to quickly measure the ability of nodes to deliver messages to the activity range of the destination node. If the node carrying messages is not in the social circle of the destination node, we will select social circle members of the destination, or forward messages to the nodes with larger transmission utility to reduce average delay.

Definition 3 (*Transmission Utility of Node (TUoN)*)**.** The $TUoN$ of node v_i, say $TUoN_i$, is the ratio of the moving speed of node v_i to the product of the angle between the direction of motion of node v_i and the direction of motion of the destination node v_d and the distance of the node v_i from the destination node v_d. $TUoN_i$ can be calculated by:

$$TUoN_i = \frac{|V_i|}{\vartheta_{(i,d)} \cdot D_{(i,d)}}, \tag{5}$$

where $|V_i|$ is the moving speed of v_i, $\vartheta_{(i,d)}$ represents the angle between the direction of motion of v_i and the direction of motion of the destination v_d. Assuming that the velocity vector of v_i is $V_i(v_{xi}, v_{yi})$ and the velocity vector of v_d is $V_d(v_{xd}, v_{yd})$, then $\vartheta_{(i,d)}$ is calculated as (6). $D_{(i,d)}$ represents the distance of v_i from the destination v_d. Assuming that the coordinate of v_i is (x_i, y_i) and the coordinate of v_d is (x_d, y_d), then $D_{(i,d)}$ is computed as (7).

$$\vartheta_{(i,d)} = \frac{V_i \cdot V_d}{|V_i| \cdot |V_d|} = \frac{v_{xi} \cdot v_{xd} + v_{yi} \cdot v_{yd}}{\sqrt{(v_{xi})^2 + (v_{yi})^2} \cdot \sqrt{(v_{xd})^2 + (v_{yd})^2}}. \tag{6}$$

$$D_{(i,d)} = \sqrt{(x_i - x_d)^2 + (y_i - y_d)^2}. \tag{7}$$

GI-RFS uses $TUoN$ to reflect the transmission capacity of the node. If the transmission utility of the node is large, we can infer that the ability of the node to quickly deliver messages to the activity range of the destination node is strong. The specific routing forwarding process of GI-RFS is shown in Algorithm 3.

If v_i encounters v_j, the number of copies of m_k carried by node v_i is greater than 1, and the node carrying messages is not in the social circle of the destination, the $P_{(i,j)}$ and SCN_i should be updated firstly. Then, we obtain the messages carried by v_i but not carried by v_j, recorded as SV, by calculating the intersection of SV_i and $\overline{SV_j}$. If v_j is the destination of m_k, v_i directly forwards m_k to v_j. Otherwise, we need to calculate $TUoN_i$ and $TUoN_j$ as (5). If $v_j \in SCN_d$ or $TUoN_i < TUoN_j$, or $nNodes < 2$, we add m_k to $forwardList$. When all the messages in SV are traversed, we need to judge whether the $forwardList$ is empty or not. If the $forwardList$ is not empty, we sort the messages contained in the $forwardList$ according to the remaining TTL and forward the $forwardList$.

In algorithm 3, time is mainly spent on the steps 4 to 13. So the forwarding process of GI-RFS can be completed in time O(m), where m is the number of messages in the network.

5. An adaptive multiple spray-and-wait routing algorithm based on social circles (SC-AMSW)

In this section, we presented a routing algorithm called adaptive multiple spray-and-wait routing algorithm based on social circles (SC-AMSW) to further improve the performance of SC-SS. Since the network

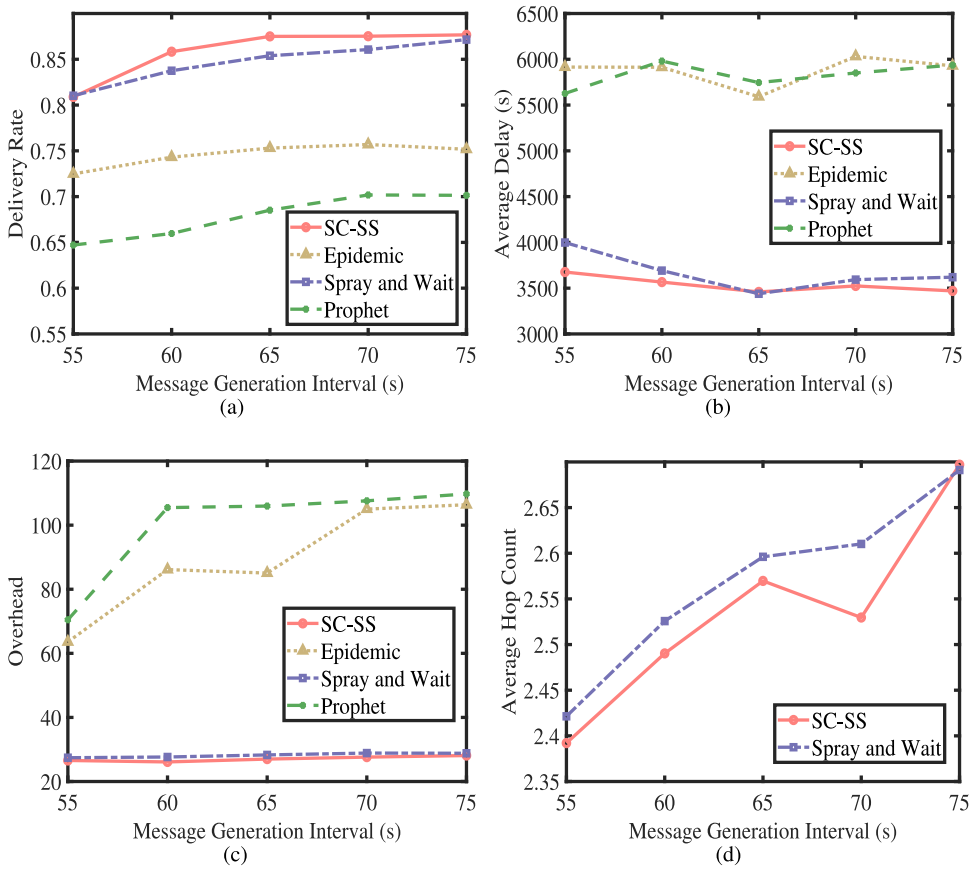

Fig. 8. Comparison of various algorithms under different message generation interval.

Algorithm 3 Pseudo Code for GI-RFS

Input: $V = \{v_i | 1 < i < n\}$, set of nodes in the network
$L_i(m_k)$, the number of copies of m_k carried by node v_i
SCN_d, social circle of the destination
$nNodes$, the number of neighbor nodes in v_i's communication range
SV_i, set of messages carried by v_i
SV_j, set of messages carried by v_j
Initialize $forwardList = Null$

if v_i encounters v_j and $L_i(m_k) > 1$ and $v_i \notin SCN_d$ **then**
  update $P_{(i,j)}$ and SCN_i
  $SV = SV_i \cap \overline{SV_j}$
  for each message m_k in SV **do**
    calculate $TUoN_i$ and $TUoN_j$ as (5)
    $v_d = m_k$'s destination
    if $v_d == v_j$ **then**
      v_i directly forwards m_k to v_j
    else if $v_j \in SCN_d$ or $TUoN_i < TUoN_j$ or $nNodes < 2$ **then**
      $L_j(m_k) = \lfloor L_i(m_k)/2 \rfloor$
      add $L_j(m_k)$ copies of m_k to $forwardList$
    end if
  end for
  if $forwardList$!= NULL **then**
    sort $forwardList$ in ascending order of TTL
    v_i forward $forwardList$ to v_j
  end if
end if

topology, the number of communication links, and the communication area in DTN are dynamically changing, the algorithm with a fixed number of copies is not flexible enough. Node's energy, cache, bandwidth, and other resources are very limited. Once the number of copies that a node hold is 1, simply spraying messages with one copy multiple times in the wait phase can effectively improve the algorithm's performance. However, it may generate excessive redundant copies, easily lead to network congestion, and reduce the routing algorithm's overall performance. For example, the number of copies of a certain message that a node hold is 1, but the probability that the node successfully delivers the message to the destination node is high. We do not have to spray the message multiple times in this case to reduce unnecessary, redundant copies.

Based on the above analysis, we implement an adaptive multiple spray strategy based on delivery predictability (DP-AMSS) in the wait phase. DP-AMSS selectively sprays messages with one copy multiple times based on delivery predictability, which will not bring huge pressure to the cache while inheriting the advantages of multi-copy strategy in delivery rate and average delay. Specifically, when v_i encounters v_j with larger delivery predictability, and the number of copies of m_k with destination v_d hold by v_i is 1, v_i will respray m_k if $P_{(i,d)} < P_{threshold}$. The number of re-sprayed copies (NRC) is calculated as (8). Moreover, to avoid redundant copies of the same message existing in a certain local area, we stipulated that each node re-spray the same message no more than twice.

$$NRC = \begin{cases} \lfloor L \cdot (1 - P_{(i,d)}) \rfloor, if \lfloor L \cdot (1 - P_{(i,d)}) \rfloor \cdot S_k < RS_i \\ \lfloor \frac{RS_i}{2S_k} \rfloor, \ others, \end{cases} \tag{8}$$

where RS_i is the remaining cache size of v_i, S_k refers to the size of the message m_k, L represents the initial number of copies of m_k,

$P_{(i,d)}$ denotes the probability of v_i successfully delivering a message to the destination v_d. If $P_{(i,d)}$ is smaller, NRC calculated by (8) is larger. Because the smaller $P_{(i,d)}$ indicates that the probability of v_i successfully delivering m_k to the destination v_d is smaller, we should spray more copies of m_k to improve the probability of m_k being successfully delivered to v_d. Node v_i with larger $P_{(i,d)}$ is more likely to successfully forward m_k to v_d, we do not need to respray the message to reduce unnecessary redundant copies and unnecessary relay forwarding times. So v_i is allowed to respray m_k only when the probability that v_i successfully forwards m_k to v_d is less than $P_{threshold}$.

In wait phase, when v_i encounters v_j, we will adopt DP-AMSS to selectively spray messages with one copy multiple. The specific routing process is shown in Algorithm 4.

Algorithm 4 Pseudo Code for DP-AMSS

Input: $V = \{v_i | 1 < i < n\}$, set of nodes in the network
$L_i(m_k)$, the number of copies of m_k carried by node v_i
$nNodes$, the number of neighbor nodes in v_i's communication range
SV_i, set of messages carried by v_i
SV_j, set of messages carried by v_j
$P_{threshold}$, threshold of delivery predictability
$ST_i(m_k)$, the number of times v_i has re-sprayed m_k

```
1: if v_i encounters v_j and L_i(m_k) = 1 then
2:   update P_(i,j) and SCN_i
3:   SV = SV_i ∩ ¬SV_j
4:   for each message m_k in SV do
5:     v_d = m_k's destination
6:     if v_d == v_j then
7:       v_i directly forwards m_k to v_j
8:     else if P_(i,d) < P_threshold and ST_i(m_k) < 2 and P_(i,d) < P_(j,d) then
9:       NRC = ⌊L · (1 − P_(i,d))⌋
10:      if NRC · S_k > RS_i then
11:        NRC = ⌊RS_i / 2S_k⌋
12:      end if
13:      if NRC > 1 then
14:        ST_i(m_k) = NRC
15:        ST_i(m_k) = ST_i(m_k) + 1
16:      end if
17:    end if
18:  end for
19: end if
```

If v_i encounters v_j, and the number of copies of m_k carried by node v_i is equal to 1, the $P_{(i,j)}$ and SCN_i should be updated firstly. Then, we obtain the messages carried by v_i but not carried by v_j, recorded as SV, by calculating the intersection of SV_i and $\overline{SV_j}$. We traverse each message m_k in SV. If v_j is the destination of m_k, v_i directly forwards m_k to v_j. Otherwise, if $P_{(i,d)} < P_{threshold}$, $ST_i(m_k) < 2$, and $P_{(i,d)} < P_{(j,d)}$ are satisfied, we calculate NRC according to Eq. (8). If NRC is greater than 1, we update the number of copies of m_k to NRC and add spray times by 1.

The main time consumption of algorithm 4 is on the steps 4 to 18, and these steps can be completed in time O(m), where m is the number of messages in the network. So the time complexity is O(m).

6. Acknowledgment (ACK) mechanism

In this section, we presented an ACK mechanism to delete redundant copies that have been successfully delivered to the destination. Each node maintains an ACK message list AMI for recording the set of message IDs that has been successfully forwarded to the destination. When two nodes encounter, they exchange each other's AMI, add the message ID which is not in their own AMI but in the encountered node's AMI to their own AMI, and then delete the messages that have been recorded in their AMI. When v_i encounters v_j, the specific execution process of the ACK mechanism is shown in Algorithm 5.

Algorithm 5 Pseudo Code for ACK Mechanism

Input: $V = \{v_i | 1 < i < n\}$, set of nodes in the network
AMI_i, ACK message list of v_i
AMI_j, ACK message list of v_j
SV_i, set of messages carried by v_i
SV_j, set of messages carried by v_j

```
1: if AMI_i != NULL then
2:   add all the message IDs in AMI_i to AMI_j
3:   for each ID in AMI_j do
4:     if ID ∈ SV_j and the message with id ID is not being sent then
5:       v_j delete the message with id ID from SV_j
6:     end if
7:   end for
8: end if
9: if AMI_j != NULL then
10:  add all the message IDs in AMI_j to AMI_i
11:  for each ID in AMI_i do
12:    if ID ∈ SV_i and the message with id ID is not being sent then
13:      v_i delete the message with id ID from SV_i
14:    end if
15:  end for
16: end if
```

Table 2
Simulation parameters.

Parameters	Default values	Variation range
Simulation time (h)	12	6~14
Scenario size (m^2)	4500 × 3400	–
Transmit speed (KB/s)	250	–
Buffer size (MB)	10	10~30
Number of nodes	126	–
TTL (min)	240	30~240
Message generation interval (s)	30	25~75
L	32	–
$P_{threshold}$	0.7	–
$SCN_{threshold}$	$\frac{1}{2min(\lvert E_i\rvert,\lvert E_j\rvert)}$	–
P_{init}	0.75	–
β	0.25	–
γ	0.98	–

When v_i encounters v_j, if the AMI of v_i is not empty, v_i sends AMI_i to the encountered node v_j. Node v_j performs the union operation between AMI_i and AMI_j, and uses the operation result as new AMI_j. We traverse each message ID in AMI_j to determine whether the message is stored in the cache of v_j. If it is true, and the message is not being transmitted, v_j deletes the message from the local cache. Similarly, if the AMI of the v_j is not empty, v_j sends the AMI_j to the encountered node v_i, v_i performs the union operation between AMI_j and AMI_i, and uses the operation result as a new AMI_i. Then we traverse each message ID in AMI_i to determine whether the message is stored in the cache of the v_i. If it is true, and the message is not being transmitted, v_i deletes the message from its local cache.

The main time consumption of algorithm 5 is on steps 3 to step 7 and the steps 11 to 15. So the time complexity is O(m), where m is the number of messages in the network.

7. Simulation

7.1. Simulation setup

We use the ONE [27] simulation to create the scenario and use the Helsinki Map [28] as the mobile area. The Helsinki Map can be obtained in the ONE simulator. We evaluate the routing algorithms from four metrics: delivery rate, average delay, overhead, and average hop count. P_{init}, β, and γ are tunable parameters, with reference to [5],

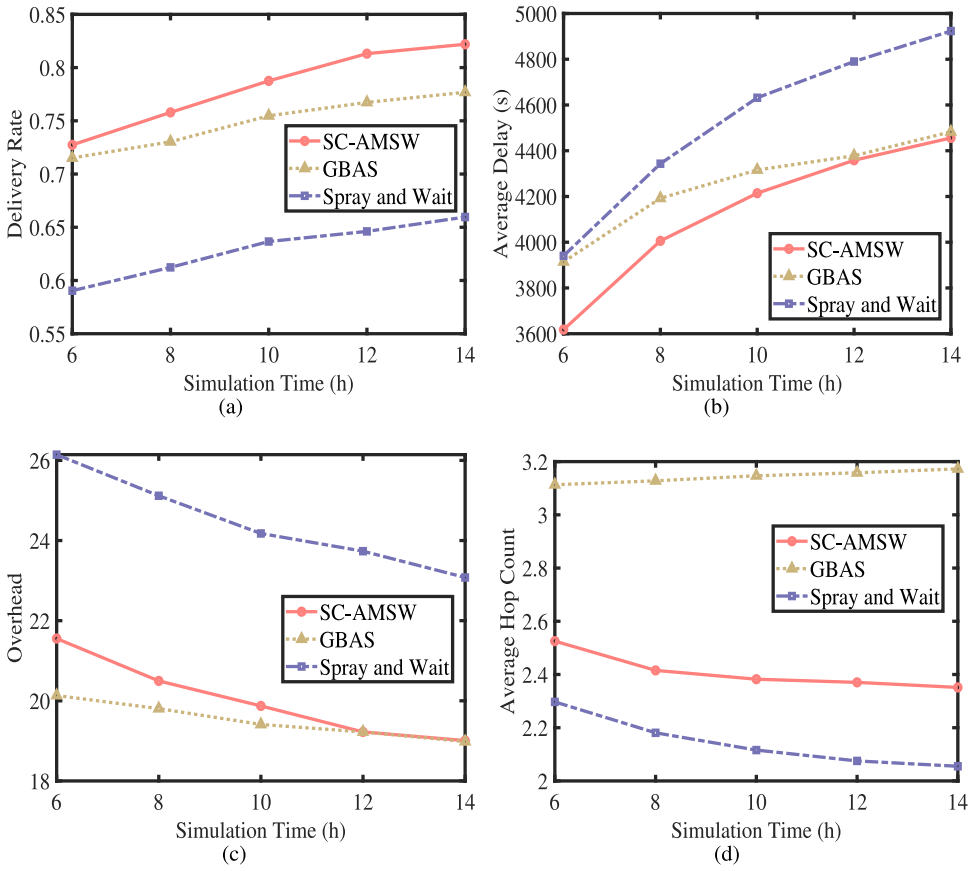

Fig. 9. Comparison of various algorithms under different simulation time.

$P_{init} = 0.75$, $\beta = 0.25$, and $\gamma = 0.98$. After many experiments, SC-AMSW can achieve better performance when $P_{threshold} = 0.7$. Table 2 gives the specific parameters of the simulation.

7.2. Discussion about the threshold $P_{threshold}$

In the wait phase of SC-AMSW, we leverage $P_{threshold}$ to decide whether respray messages or not. The parameter $P_{threshold}$ will directly affect the performance of SC-AMSW. Thus, we have done a lot of simulations under different values of $P_{threshold}$. Fig. 4 shows the delivery rate of SC-AMSW with an increase of $P_{threshold}$. We can observe that when $P_{threshold}$ is 0.7, SC-AMSW achieves better performance than the other. This is because if $P_{threshold}$ is small, the restrictions on the spray of messages are too strict, and then only a few messages have chances to be sprayed multiple times in the wait phase. If $P_{threshold}$ is large, most messages with one copy will be sprayed multiple times. Then there will be excessive redundant copies in the network, which may cause network congestion and lead to a low delivery rate. In this paper, $P_{threshold}$ is set to 0.7.

7.3. Demonstrating the effectiveness of SC-SS

For demonstrating the effectiveness of SC-SS, we compare SC-SS with three algorithms: Epidemic, Prophet, and Spray-and-Wait by changing simulation time, buffer size, TTL, and message generation interval. Since the maximum number of message copies in Epidemic and Prophet is unlimited, the maximum number of message hops is also unlimited. Therefore, we do not draw their average hop count in figures.

7.3.1. Impact of simulation time

Fig. 5 presents the performance of the four routing algorithms under different simulation times. As shown in Fig. 5(a), we observe that SC-SS respectively outperforms Epidemic, Prophet, and Spray-and-Wait by 16.10%, 24.73%, and 3.00% in terms of the delivery rate. That is because SC-SS constructs the social circle of nodes based on the similarity degree between nodes and adopts different relay node selection strategies according to whether the node carrying messages is in the destination's social circle. In this way, messages are always transmitted along with the relay candidates with better performance. The delivery rate of Epidemic and Prophet is low because neither Epidemic nor Prophet limits the maximum number of message copies, which may easily cause network congestion, then many messages will be discarded.

In Fig. 5(b), we observe that the average delay of SC-SS is the lowest, and the average delay of SC-SS is about 31.18%, 32.53%, and 12.60% lower than that of Epidemic, Prophet, and Spray-and-Wait. The main reason is that SC-SS selects nodes with better performance as the next hops, which speeds up the arrival of messages at the destination. From Fig. 5(c), we observe that the overhead of SC-SS is respectively about 69.70%, 68.58%, and 8.14% lower than Epidemic, Prophet, and Spray-and-Wait. This is because SC-SS optimizes the selection of relay nodes in spray phase, which reduces low efficient relay forwarding times. As depicted in Fig. 5(d), the average hop count of SC-SS is reduced by 1.81% compared with Spray-and-Wait, which proves the validity of the relay node selection strategy adopted by SC-SS in the spray phase.

7.3.2. Impact of buffer size

Fig. 6 shows the impact of buffer size. As depicted in Fig. 6(a), SC-SS respectively outperforms Epidemic, Prophet, and Spray-and-Wait by 24.71%, 35.44%, and 7.71% in terms of the delivery rate. We can

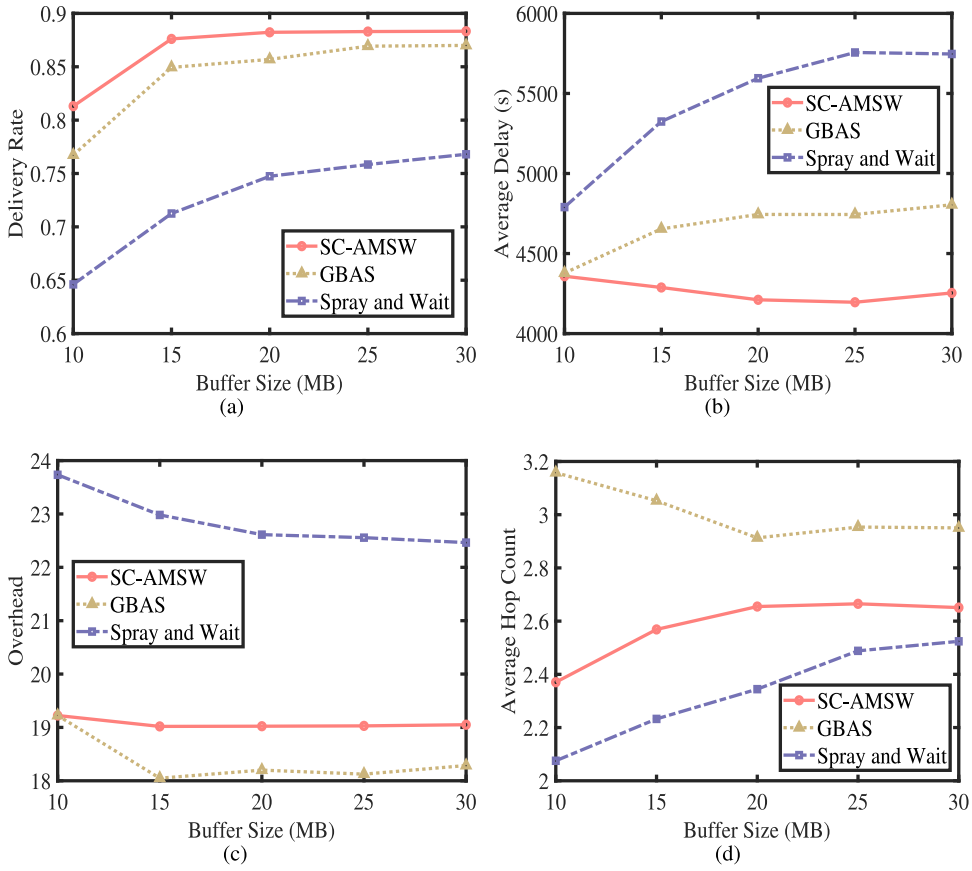

Fig. 10. Comparison of various algorithms under different buffer size.

also observe that these four routing algorithms' delivery rate increases as buffer size increases from the figure. This is because the larger the buffer, the more messages a node can carry, and then the situation that many messages are discarded due to network congestion will gradually ease. Fig. 6(b) shows the average delay, where it can be seen that the average delay of SC-SS is respectively about 32.53%, 32.02%, and 19.51% lower than that of Epidemic, Prophet, and Spray-and-Wait.

From Fig. 6(c), the overhead of SC-SS can reduce 58.67%, 63.42%, and 11.16% on average compared with Epidemic, Prophet, and Spray-and-Wait. Moreover, Epidemic and Prophet's overhead shows an obvious downward trend as buffer size increases, which reflects that these two algorithms are more practical in a resource-rich scenario. Fig. 6(d) presents the average hop count under different buffer size. We note that the average hop count of SC-SS is about 2.42% lower on average than Spray-and-Wait. Moreover, the average hop count of Spray-and-Wait and SC-SS has an obvious growing trend when buffer size increases. This is because increasing the node's cache allows the discarded messages due to network congestion can continue to survive in the network. These messages will have more opportunities to be successfully forwarded to the destination after being forwarded multiple times.

7.3.3. Impact of TTL

Fig. 7 shows the impact of TTL. From Fig. 7(a), we observe that SC-SS respectively outperforms Epidemic, Prophet, and Spray-and-Wait by 23.07%, 37.68%, and 7.14% in terms of the delivery rate. In addition, the figure also shows that the delivery rate of these algorithms increases as TTL increases. Because if TTL is increased, message copies can survive for a longer time in nodes' buffer, then nodes can have more time to forward messages to the destination. In Fig. 7(b), we observe that the average delay of SC-SS is about 14.89%, 12.96%, and 9.51% lower than that of Epidemic, Prophet, and Spray-and-Wait. If TTL is increased, then many messages that cannot be transmitted to the destination due to short TTL have more chances to be delivered. So the average delay of the four algorithms has a growing trend with increasing TTL.

As shown in Fig. 7(c), the overhead of SC-SS is reduced by 38.90%, 10.90%, and 9.72% compared with Epidemic, Prophet, and Spray-and-Wait. We can also observe that Epidemic's overhead is much higher than other routing algorithms because flooding-based routing algorithms need to consume more cache space, and it is applicable for scenarios with sufficient cache resources. As depicted in Fig. 7(d), the average hop count of SC-SS is decreased by 4.54% compared with Spray-and-Wait.

7.3.4. Impact of message generation interval

Fig. 8 shows the impact of message generation interval. As depicted in Fig. 8(a), SC-SS significantly outperforms the other three algorithms on the delivery rate, which can be illustrated by the advantages of SC-SS mentioned above. We observe that SC-SS respectively outperforms Epidemic, Prophet, and Spray-and-Wait by 15.11%, 26.50%, and 1.42% in terms of the delivery rate. In addition, the number of message copies stored in the buffer is reduced if the message generation interval is increased, so there will be more available cache resources can be obtained, and the delivery rate will be increased. From Fig. 8(b), we note the average delay of SC-SS is about 39.74%, 39.23%, and 3.38% lower than Epidemic, Prophet, and Spray-and-Wait respectively.

Fig. 8(c) shows the overhead under different message generation interval. The overhead of SC-SS is reduced by 68.68%, 72.16%, and 4.02% compared with Epidemic, Prophet, and Spray-and-Wait. As depicted in Fig. 8(d), the average hop count of SC-SS is reduced by 1.30% compared with Spray-and-Wait. In addition, the average hop count increases with increasing message generation interval. This is because more resources can be obtained when message generation interval

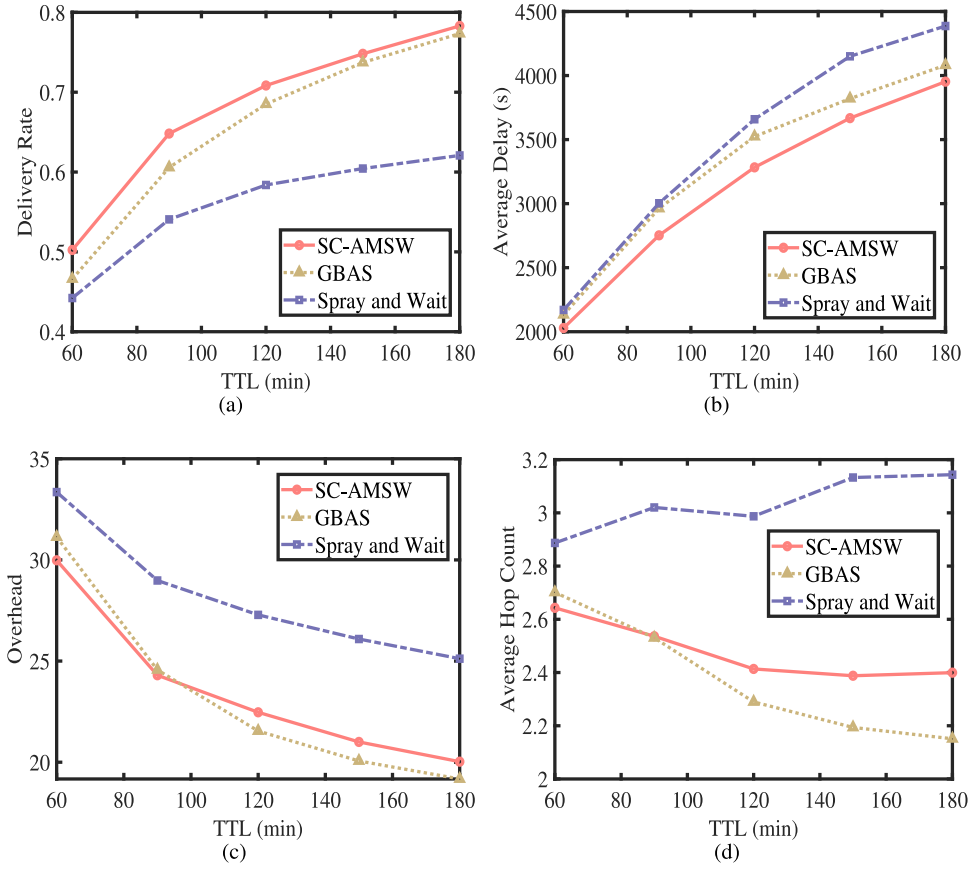

Fig. 11. Comparison of various algorithms under different TTL.

increases, then some messages discarded due to network congestion can be successfully forwarded to the destination after multi-hop forwarding.

7.4. Demonstrating the effectiveness of SC-AMSW

In this subsection, to prove the validity of SC-AMSW, we evaluate the performance of SC-AMSW, Spray-and-Wait, and GBAS under different simulation time, buffer size, TTL, and message generation interval.

7.4.1. Impact of simulation time

Fig. 9 shows the impact of simulation time. The delivery rate under different simulation time is shown in Fig. 9(a), where it can be seen that SC-AMSW respectively outperforms Spray-and-Wait and SC-AMSW by 24.23% and 4.32% in terms of the delivery rate. This is attributed to two reasons. Firstly, SC-AMSW selects suitable relay nodes from many candidate relays. The message can be forwarded to the active range of motion of the destination as soon as possible. Secondly, SC-AMSW adopts DP-AMSS to spray messages with one copy multiple times selectively, then the overall performance of the algorithm is improved by increasing the count of message copies. From Fig. 9(b), we observe that the average delay of SC-AMSW is respectively about 8.69% and 3.08% lower than Spray-and-Wait and GBAS. The result is attributed to wait phase, in which SC-AMSW selectively spray messages with one copy multiple times to speed up the spread of messages so that messages can be quickly forwarded to the destination.

As depicted in Fig. 9(c), the overhead of SC-AMSW is reduced by 18.08% compared with Spray-and-Wait. This is because SC-AMSW avoids allocating message copies to all encountered nodes, then the unnecessary relay forwarding times are reduced greatly. The overhead of SC-AMSW is slightly higher than that of the GBAS, because GBAS will spray each message with one copy multiple times in wait phase. However, network resources are very limited, blindly spraying messages multiple times may generate excessive redundant copies, easily causing network congestion. Then many messages that have just been generated or received by the receiver will be discarded before they have a chance to be forwarded. Fig. 9(d) presents the average hop count. We note that Spray-and-Wait has the lowest average hop count. This is because Spray-and-Wait does not have any optimization strategy in wait phase. The average hop count of SC-AMSW is reduced by 23.35% compared with GBAS. This is because SC-AMSW allows the message to be re-sprayed only when the probability of a node successfully forwarding messages to the destination node is less than the given threshold $P_{threshold}$ in wait phase.

7.4.2. Impact of buffer size

Fig. 10 compares the performance of the three routing algorithms under different buffer size. From Fig. 10(a), we note that SC-AMSW has an obvious advantage in the delivery rate. The figure shows that SC-AMSW outperforms Spray-and-Wait and GBAS by 19.66% and 3.04% respectively in terms of the delivery rate, which demonstrates the effectiveness of SC-AMSW once again. We can also see that each algorithm's delivery rate has an obvious growing trend when buffer size increases. Because the larger the buffer, the more messages a node can carry, and then the situation that many messages are discarded due to network congestion will gradually ease.

In Fig. 10(b), we observe that the average delay of SC-AMSW is respectively about 21.26% and 8.52% lower than that of Spray-and-Wait and GBAS. As depicted in Fig. 10(c), the overhead of SC-AMSW is about 16.59% lower on average than Spray-and-Wait. Fig. 10(d) presents the average hop count under different buffer size. We can note that the average hop count of SC-AMSW is decreased by 13.91% compared with GBAS. We do not repeat the reasons.

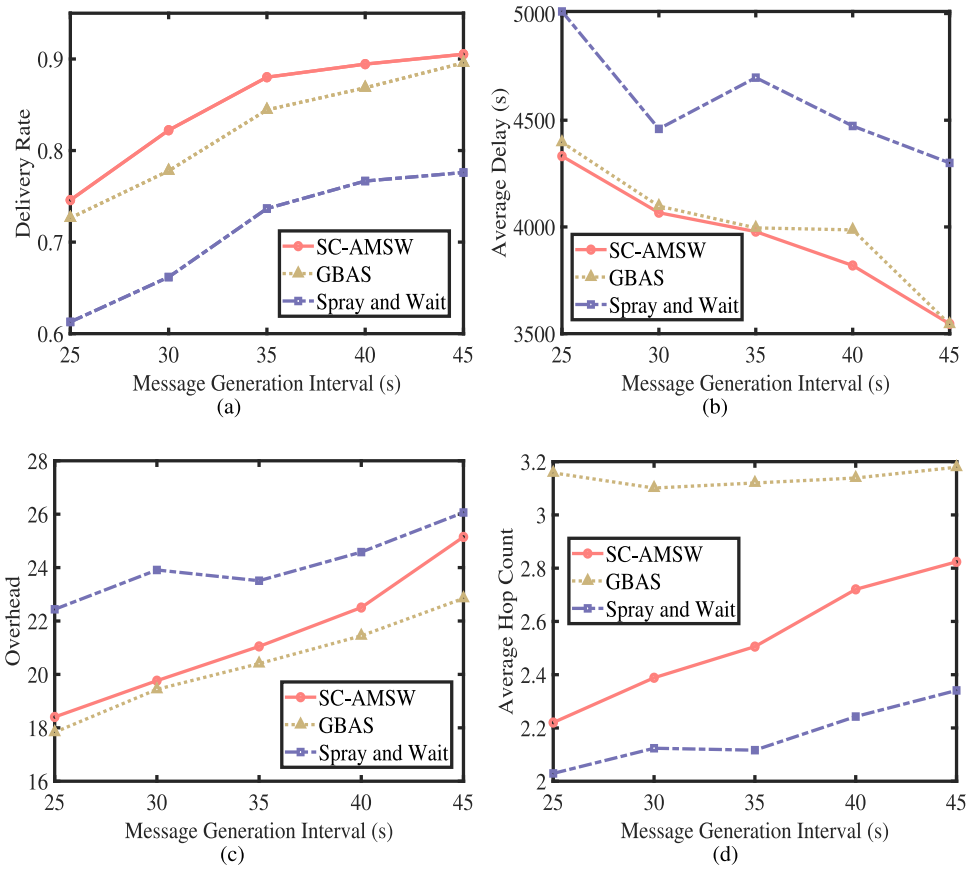

Fig. 12. Comparison of various algorithms under different message generation interval.

7.4.3. Impact of TTL

Fig. 11 shows the impact of TTL. From Fig. 11(a), we observe that SC-AMSW respectively outperforms Spray-and-Wait and GBAS by 20.95% and 4.19% in terms of the delivery rate, which proves the validity of SC-AMSW once again. From Fig. 11(a), we can also observe that the delivery rate of these three algorithms increases as TTL increases. This is because a large number of messages that cannot be successfully forwarded to the destination node due to short TTL have more chances to encounter the destination when TTL increases.

From Fig. 11(b), we observe that the average delay of SC-AMSW is respectively about 9.33% and 5.18% larger than Spray-and-Wait and GBAS. Moreover, as the TTL increases, the average delay of these three algorithms shows a significant growth trend. This is because when TTL increases, many messages that cannot be successfully forwarded to the destination node due to short TTL have chances to be successfully forwarded, then it takes a longer average time to forward a message from source to destination. Fig. 11(c) shows the overhead under different TTL. We can observe that the overhead of SC-AMSW is reduced by 16.74% compared with Spray-and-Wait. In Fig. 11(d), the average hop count of SC-AMSW is lower than GBAS by 18.22%, which demonstrates the effectiveness of DP-AMSS implemented in wait phase of SC-AMSW.

7.4.4. Impact of message generation interval

Fig. 12 shows the impact of the message generation interval. From Fig. 12(a), we can observe that SC-AMSW respectively outperforms Spray-and-Wait and GBAS by 19.75% and 3.33% in terms of the delivery rate, which proves the validity of SC-AMSW once again. From Fig. 12(b), we observe that the average delay of SC-AMSW is respectively about 13.97% and 1.36% lower than Spray-and-Wait and GBAS. As depicted in Fig. 12(c), the overhead of SC-AMSW is reduced by 11.55% compared with Spray-and-Wait. Fig. 12(d) presents the average hop count. We can observe that the average hop count of SC-AMSW is reduced by 19.37% compared with GBAS. We do not repeat the reasons again.

As mentioned above, SC-AMSW outperforms all the other algorithms on the delivery rate and the average delay under different simulation time, buffer size, TTL, and message generation interval. The overhead of SC-AMSW is slightly higher than GBAS, but lower than Spray-and-Wait. The average hop count is higher than spray-and-wait, but lower than GBAS. Therefore, the overall performance of SC-AMSW is optimal, which demonstrates the effectiveness of SC-AMSW.

8. Conclusion

In this paper, we presented a method of constructing social circles based on the clustering phenomenon of nodes. To make more effective routing decisions, we introduced a spray strategy based on social circles (SC-SS) to improve the spray phase of Spray-and-Wait. SC-SS uses different relay node selection strategies according to whether the message-carrying node is in the social circle of the destination node. If the message-carrying node is in the social circle of the destination node, SC-SS will forward messages within the social circle of the destination node and adopt a routing forwarding strategy based on delivery predictability (DP-RFS). If the node carrying messages is not in the social circle of the destination, SC-SS will transmit messages to the nodes in the social circle of the destination node, or adopt a routing forwarding strategy based on geographical information (GI-RFS). Besides, we presented an adaptive multiple spray-and-wait routing algorithm based on social circles (SC-AMSW) to improve the wait phase of SC-SS. SC-AMSW can adaptively adjust the number of message copies to adapt to the dynamically changing network environment. Simulation results show that SC-AMSW has the best overall performance on the delivery rate, the average delay, the overhead, and the average hop count.

We will continue this work in the following ways. Firstly, SC-AMSW uses a flooding-based mechanism to notify nodes to delete useless, redundant copies, but the flooding-based way may increase the overhead. A natural avenue for future research is to propose a method of quickly deleting useless, redundant copies. Secondly, a message scheduling mechanism must be proposed to further improve the algorithm in the future.

CRediT authorship contribution statement

Libing Wu: Conceptualization, Supervision. **Shuqin Cao:** Conceptualization, Methodology, Software, Writing - original draft. **Yanjiao Chen:** Writing - review & editing, Supervision. **Jianqun Cui:** Supervision. **Yanan Chang:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. U20A20177, 61772377, 91746206, 61972296, 61672257, 61702210), the Fundamental Research Funds for the Central Universities (2042020kf0217), Wuhan Advanced Application Project (2019010701011419), and Science and Technology planning project of ShenZhen (JCYJ20170818112550194).

References

[1] A. Roy, T. Acharya, S. DasBit, Quality of service in delay tolerant networks: A survey, Comput. Netw. 130 (2018) 121–133.
[2] K. Sakai, M.-T. Sun, W.-S. Ku, J. Wu, F.S. Alanazi, Performance and security analyses of onion-based anonymous routing for delay tolerant networks, IEEE Trans. Mob. Comput. 16 (12) (2017) 3473–3487.
[3] Y. Li, P. Hui, D. Jin, S. Chen, Delay-tolerant network protocol testing and evaluation, IEEE Commun. Mag. 53 (1) (2015) 258–266.
[4] A. Vahdat, D. Becker, Epidemic routing for partially-connected ad hoc networks, Tech. rep., Technical Report CS-200006, Duke University, 2000.
[5] A. Lindgren, A. Doria, O. Schelén, Probabilistic routing in intermittently connected networks, in: Service Assurance with Partial and Intermittent Resources, Springer, 2004, pp. 239–254.
[6] T. Spyropoulos, K. Psounis, C.S. Raghavendra, Spray and wait: an efficient routing scheme for intermittently connected mobile networks, in: ACM SIGCOMM Workshop on Delay-Tolerant Networking, ACM, 2005, pp. 252–259.
[7] H. Yang, Geography-Based Adaptive Spray Routing Algorithm in Delay Tolerant Network (Master's thesis), Central China Normal University, 2018.
[8] J. Guan, Q. Chu, I. You, The social relationship based adaptive multi-spray-and-wait routing algorithm for disruption tolerant network, Mob. Inf. Syst. 2017 (2017).
[9] S.-H. Kim, S.-J. Han, Distinguishing social contacts and pass-by contacts in DTN routing, Telecommun. Syst. 68 (4) (2018) 669–685.
[10] A. Roy, S. Bose, T. Acharya, S. DasBit, Social-based energy-aware multicasting in delay tolerant networks, J. Netw. Comput. Appl. 87 (2017) 169–184.
[11] P. Yuan, M. Song, MONICA: One simulator for mobile opportunistic networks, in: International Conference on Mobile Multimedia Communications, European Alliance for Innovation, 2018, p. 21.
[12] M. Grossglauser, D.N.C. Tse, Mobility increases the capacity of ad hoc wireless networks, IEEE/ACM Trans. Netw. 10 (4) (2002) 477–486.
[13] Q. Ayub, S. Rashid, Resource refrain quota based routing protocol for delay tolerant network, Wirel. Netw. 25 (8) (2019) 4695–4704.
[14] N. Derakhshanfard, M. Sabaei, A.M. Rahmani, Sharing spray and wait routing algorithm in opportunistic networks, Wirel. Netw. 22 (7) (2016) 2403–2414.
[15] E. Spaho, K. Dhoska, L. Barolli, V. Kolici, M. Takizawa, Enhancement of binary spray and wait routing protocol for improving delivery probability and latency in a delay tolerant network, in: International Conference on Broadband and Wireless Computing, Communication and Applications, Springer, 2019, pp. 105–113.
[16] P. Hui, J. Crowcroft, E. Yoneki, Bubble rap: Social-based forwarding in delay-tolerant networks, IEEE Trans. Mob. Comput. 10 (11) (2010) 1576–1589.
[17] S. Jain, A. Verma, Bubble rap incentive scheme for prevention of node selfishness in delay-tolerant networks, in: Smart Innovations in Communication and Computational Sciences, Springer, 2019, pp. 289–303.
[18] K. Chen, H. Shen, SMART: Utilizing distributed social map for lightweight routing in delay-tolerant networks, IEEE/ACM Trans. Netw. 22 (5) (2013) 1545–1558.
[19] M. Xiao, J. Wu, L. Huang, Community-aware opportunistic routing in mobile social networks, IEEE Trans. Comput. 63 (7) (2013) 1682–1695.
[20] G. Yu, Z. Chen, J. Wu, J. Wu, Predicted encounter probability based on dynamic programming proposed probability algorithm in opportunistic social network, Comput. Netw. (2020) 107465.
[21] F. Li, H. Jiang, H. Li, Y. Cheng, Y. Wang, SEBAR: social-energy-based routing for mobile social delay-tolerant networks, IEEE Trans. Veh. Technol. 66 (8) (2017) 7195–7206.
[22] K. Wei, D. Zeng, S. Guo, K. Xu, On social delay-tolerant networking: Aggregation, tie detection, and routing, IEEE Trans. Parallel Distrib. Syst. 25 (6) (2013) 1563–1573.
[23] W. Gao, G. Cao, T. La Porta, J. Han, On exploiting transient social contact patterns for data forwarding in delay-tolerant networks, IEEE Trans. Mob. Comput. 12 (1) (2011) 151–165.
[24] M. Boc, A. Fladenmuller, M.D. de Amorim, L. Galluccio, S. Palazzo, Price: Hybrid geographic and co-based forwarding in delay-tolerant networks, Comput. Netw. 55 (9) (2011) 2352–2360.
[25] N. Shrestha, L. Sassatelli, Inter-session network coding-based policies for delay tolerant mobile social networks, IEEE Trans. Wireless Commun. 15 (11) (2016) 7329–7342.
[26] T. Wang, M. Tang, H. Song, Y. Cao, Z. Khan, Opportunistic protocol based on social probability and resources efficiency for the intelligent and connected transportation system, Comput. Netw. 149 (2019) 173–186.
[27] A. Keränen, J. Ott, T. Kärkkäinen, The ONE simulator for DTN protocol evaluation, in: International Conference on Simulation TOOLS and Techniques, European Alliance for Innovation, 2009, p. 55.
[28] E. Bulut, B.K. Szymanski, Exploiting friendship relations for efficient routing in mobile social networks, IEEE Trans. Parallel Distrib. Syst. 23 (12) (2012) 2254–2265.

Libing Wu received his Ph.D. degree in Computer Science from Wuhan University. Now he is a Professor in Computer School of Wuhan University. His current research interests include network communication, grid computing and Internet of Things.

Shuqin Cao received her M.S. degrees in computer science from Central China Normal University, Wuhan, China, in 2016. She is currently pursuing Ph.D. degree in computer science from Wuhan University, Wuhan, China. Her current research interests include delay tolerant network, vehicular ad hoc networks, and wireless communication.

Yanjiao Chen received her B.E. degree in Electronic Engineering from Tsinghua University in 2010 and Ph.D. degree in Computer Science and Engineering from Hong Kong University of Science and Technology in 2015. She is currently a Professor in Wuhan University, China. Her research interests include computer networks, wireless system security, and network economics.

Jianqun Cui received her Ph.D. degree in Computer Science from Wuhan University, China and was a Visiting Scholar at Department of Computer Science and Engineering, SUNY at Buffalo, USA, from 2017.8 to 2018.8. She is currently a Professor in Computer School, Central China Normal University. Her research interests include opportunistic networks, Internet of Things, mobile networks and application layer multicast.

Yanan Chang received her Ph.D. degree in Computer Science by the joint Ph.D. program between Wuhan University and City University of Hong Kong and was a Visiting Scholar in Tandon School of Engineering, New York University, from 2018.1 to 2019.1. She is currently an Assistant Professor in Computer School of Central China Normal University. Her research interests include wireless mesh networks, delay tolerant networks and social networks.