A Buffer Management Strategy on Spray and Wait Routing Protocol in DTNs

En Wang[a,b], Yongjian Yang[a], Jie Wu[b] and Wenbin Liu[c]
[a]Department of Computer Science and Technology, Jilin University, China
[b]Department of Computer and Information Sciences, Temple University, USA
[c]Department of Software, Jilin University, China
wangen0310@126.com, yyj@jlu.edu.cn, jiewu@temple.edu, nike_lwb@126.com

***Abstract*—Due to unpredictable node mobility and the easily-interrupted connections, routing protocols in DTNs commonly utilize multiple message copies to improve the delivery ratio. A store-carry-and-forward paradigm is also designed to assist routing messages. However, excessive message copies lead to rapid consumption of the limited storage and bandwidth. The spray and Wait routing protocol has been proposed to reduce the network overload caused by the storage and transmission of unrestricted message copies. However, there still exist congestion problems when a node's buffer is quite constrained. In this paper, we propose a message Scheduling and Drop Strategy on spray and wait Routing Protocol (SDSRP). To improve the delivery ratio, first of all, SDSRP calculates the priority of each message by evaluating the impact of both replicating and dropping a message copy on delivery ratio. Subsequently, scheduling and drop decisions are made according to the priority. Finally, we conduct extensive simulations based on synthetic and real traces in ONE. The results show that, compared with other buffer management strategies, SDSRP achieves higher delivery ratio, similar average hopcounts, and lower overhead ratio.**

Keywords*—*DTNs, Spray and Wait, Scheduling, Drop strategy.

I. INTRODUCTION

Delay tolerant networks (DTNs) [1] are challenged networks in which end-to-end transmission latency may be arbitrarily long due to a lack of stable connections. Therefore, it is commonly unpractical to forward a message from source to destination utilizing the usual TCP/IP protocol. To solve this problem, messages in DTNs are routed in a store-carry-and-forward paradigm, which usually requires nodes to spawn and store messages. Therefore, there may be multiple copies of the same message at the same moment in DTNs. Successful delivery occurs only when one or more infected nodes encounter the destination. The concept of DTNs can be applied to various applications such as interplanetary networks [2], disaster response networks [3], rural areas [4], wildlife tracking [5], and pocket-switched networks [6].

To maximize delivery ratio, Epidemic [7] utilizes every possible connection to replicate messages to every ever-encountered node. However, excessive message copies are bound to result in network congestion. Therefore, Epidemic is actually impractical in large-scale networks. To overcome this problem, Spray and Wait [8] is proposed to limit the maximum number of message copies, and adopts a binary splitting method to distribute copies into the network. The process goes on until any message holder encounters the destination. However, there is still partial congestion due to the limited buffer size. In other words, the buffer management strategy is still required to further schedule the messages, even in the Spray and Wait routing protocol.

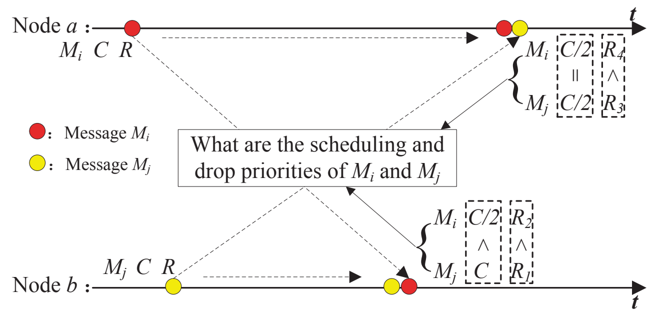


Fig. 1. An illustration of the message scheduling and drop problem (M: message id, C: message copies number, R: message remaining TTL).

An illustration of the message scheduling and drop problem is shown in Fig. 1; it is worth noticing that the abscissa represents the passage of time, while the ordinate indicates the buffer spaces of different nodes. At different times, messages M_i and M_j are generated in nodes a and b, respectively. After a period of time, node a sprays half of its copies of M_i to node b. Soon afterwards, node b also sprays half of its copies of M_j to node a. Therefore, there coexist two kinds of messages (M_i and M_j) in the buffers of both nodes a and b. However, they have different message copy numbers (C_i) and remaining TTLs (R_i). These elements will make a significant effect on priorities of messages. When a connection is established or buffer space overflows, we need to decide which message to send or drop according to the priorities.

In general, the message with a larger number of copies and a longer remaining TTL should be assigned a higher priority, since it requires more transmission opportunities. However, because of the lack of spray opportunities, there may be some messages with a large number of copies, while their TTLs are small, and vice versa. So it is also reasonable to assign a higher priority to the message whose remaining TTL or number of copies is up soon. The above analysis illustrates that the priority is not a simple linear combination, but a complex function of the number of message copies and remaining TTL. Therefore, it is necessary to find an appropriate mapping (i.e., $Priority_i = f(C_i, R_i)$), which could change the number of copies and remaining TTL into priority.

In order to manage buffer space effectively, we need to decide not only which message to send in advance, but also which message to drop. Therefore, we must make a trade-off among messages with different numbers of copies (C_i) and remaining TTLs (R_i), and then decide on a suitable priority. However, it is really challenging to perfectly map number of copies and remaining TTL to priority. The previous methods [9], [10] almost depend on the heuristic algorithms, which usually schedule the messages utilizing a normalization strategy to simply compare the magnitudes among the messages' numbers of copies or remaining TTLs. However, it is impossible to prove that the heuristic algorithm is optimal in terms of any optimization goal. In other words, the previously proposed scheduling and drop strategies commonly depend on the intuitive sense; there is a lack of a strict proof to guarantee the efficiency. For instance, if we attempt to maximize the delivery ratio, which is more important in influencing the performance between number of copies and remaining TTL.

To address the aforementioned challenging problem, this paper presents a non-heuristic algorithm SDSRP, which includes two steps. First, SDSRP calculates the priority of each message by evaluating the impact on delivery ratio of both replicating and dropping a message copy. Through this method, the message priority is expressed via number of copies and remaining TTLs. Second, the messages are sorted according to the priority. Dropping the message or not is also determined according to the priority. Finally, we conduct extensive simulations based on synthetic and real mobility traces in ONE. The results show that, SDSRP achieves higher delivery ratio, similar average hopcounts, and lower overhead ratio compared with other buffer management strategies.

The main contributions of this paper are briefly summarized as follows:

- We propose a non-heuristic message scheduling and drop strategy SDSRP, which maps the number of copies and remaining TTLs to the priority by calculating the impact on delivery ratio of both replicating and dropping a message copy. The drop decision and scheduling order are further determined according to each message's priority [11].
- A method to estimate the infection scope of messages (i.e., the number of infected nodes) is presented in the Spray and Wait routing protocol.
- We conduct extensive simulations on both synthetic and real mobility traces. The results show that SDSRP achieves the best performance regarding delivery ratio, overhead ratio, and similar performance of average hopcounts among different buffer management strategies.

The remainder of the paper is organized as follows. We review the related work in Section II. The non-heuristic message scheduling and drop strategy SDSRP is presented in Section III. In Section IV, we evaluate the performance of SDSRP through extensive simulations. We conclude the paper in Section V.

II. RELATED WORK

Buffer Management Strategies in DTNs. Researchers in DTNs have proposed some relatively effective buffer management strategies. In [12], a self-adapting optimal buffer management strategy is proposed. The mobility model is adjusted on the basis of the nodes' historical meeting records, and the message dropping strategies are designed to optimize the delivery ratio and average delay. Zhang $et\ al.$ [13] develop a rigorous and uniform framework based on ordinary differential equations (ODEs) to discuss Epidemic routing and its relevant variations. They also investigate how the buffer space and the number of message copies can be addressed for the fast and efficient delivery. The work in [14] proposes a new message scheduling framework for both Epidemic and two-hop forwarding routings in DTNs; the scheduling and dropping decisions can be made in each contact duration in order to achieve either optimal message delivery ratio or average delay. Krifa and Barakat have published three articles in terms of buffer management in DTNs. In [15], through optimizing delivery ratio and average delay, they achieve the utility value of a given message. Then they drop the message with the smallest utility when buffer overflows. According to the achievement of [15], the work in [16] extends a scheduling strategy to prioritize the message with highest utility. Considering the strategy proposed in [15], where bandwidth overloading easily occurs because excessive information has to be stored and exchanged, Krifa and Barakat in [17] propose an idealized strategy called the Global knowledge-Based Scheduling and Drop strategy (GBSD), in which signal overhead is reduced by optimizing the storage structure and statistics-collection method.

All the aforementioned buffer management strategies are only appropriate for Epidemic routing protocol. However, they are usually unusable in the Spray and Wait routing protocol.

Improvements of Spray and Wait Routing. In recent years, in order to optimize delivery ratio or average delay, researchers in DTNs have also improved the Spray and Wait routing protocol. Spray and Focus [18] is proposed to overcome the passivity of the wait phase, during which it forwards its copy to a relay node with higher utility rather than "Direct Transmission". Kim [19] proposes a combined method consisting of both the utilization of an ACK message and a forwarding method based on the delivery probability. In [20], in order to avoid identical spraying and blind forwarding among mobile nodes, an adaptive spraying scheme is defined based on the delivery predictability of nodes. Subsequently, they propose to utilize multiple spraying techniques. Although the above methods pay attention to improving the Spray and Wait protocol, they just focus on choosing the next appropriate hop [21] and controlling the number of copies. In other words, the above methods ignore the message scheduling and drop problems. For example, there is more than one message in the buffer: which message we should prioritize, and which message we should drop when the buffer overflows.

Motivated by the above drawbacks related to Spray and Wait routing protocol, we propose a message scheduling and drop strategy, which maps the copies number and remaining TTL to priority through calculating the impact on delivery ratio of both replicating and dropping a message copy. The messages are sorted and dropped according to the priority.

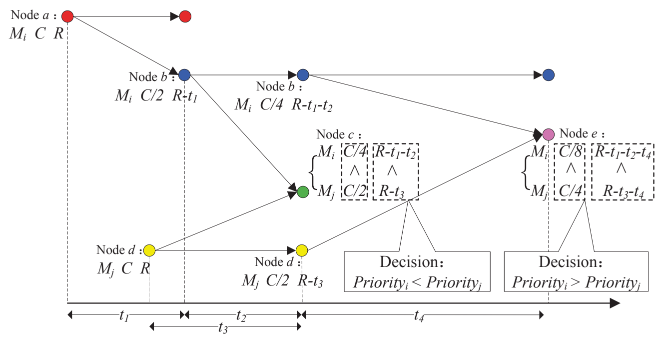


Fig. 2. A detailed example of the message scheduling and drop problem (M: message id, C: message copies number, R: message remaining TTL).

III. MESSAGE SCHEDULING AND DROP STRATEGY ON SPRAY AND WAIT

To deliver a clear problem formulation and gain useful strategy insights, we first introduce the assumptions in this paper and put forward the congestion control problem to be addressed. Next, priority is proposed to reflect the impact of duplicating and dropping a message copy on delivery ratio. Finally, based on the priority, we develop the message scheduling and drop strategy (SDSRP).

A. Problem Formulation

Considering the following network environment, there are N nodes in the fixed area; messages with random sources and destinations are generated periodically. Each message has a given TTL, after which the message is no longer useful and should be dropped. Neither an immunization strategy nor an acknowledgment mechanism is utilized to guarantee the receipt of packets. We use random-waypoint as our mobility pattern. The routing protocol in this paper adopts Spray and Wait. In addition, nodes in the network move independently, the intermeeting times tail off exponentially [22].

To maximize the delivery ratio, this paper primarily addresses the following two problems regarding the Spray and Wait routing protocol. (1) When more than one message coexists in the local buffer and the node cannot ensure whether the contact will last long enough to forward all the messages, we should make a decision regarding which message to send first. (2) If a new message arrives at a node's buffer and overflowing occurs, we should make a drop decision amongst messages already in the local buffer and the new comer.

To solve the aforementioned two problems, we attempt to obtain a message priority to decide the scheduling and dropping order. However, it is actually challenging to define a considerate priority, which can reflect the utilities of different messages. In other words, it is really difficult to find a reasonable mapping, which can change number of copies (C_i) and remaining TTLs (R_i) into priority. There must be a bridge to assist the mapping. In this paper, aiming to maximize delivery ratio, we first express the delivery ratio as a function of C_i and R_i. Then, the priority is derived from the effect of both replicating and dropping a message copy on delivery ratio ($\triangle P$). Through this method, we successfully establish a mapping from the number of copies (C_i) and remaining TTLs (R_i) to priority (as shown in Eq. 2).

$$Priority_i = \triangle P = f(C_i, R_i) \tag{2}$$

However, there are an enormous amount of mapping methods; different mapping methods result in different priorities of messages. Fig. 2 is a detailed example regarding the message priority problem. The situation is similar to the one in Fig. 1. There are also two kinds of messages (M_i and M_j) in the buffers of both nodes c and e. In node c, M_j has both a greater number of copies (C) and remaining TTL (R), compared with M_i. It indicates that M_j needs more transmission opportunities. Therefore, the decision made in node c is $Priority_i < Priority_j$. However, after a period of time, the decision in node e is exactly the opposite of that in node c. Although M_j still has both a larger C and R in node e compared with M_i, nevertheless, the number of copies (C) and remaining TTL (R) of M_i are both up soon. Therefore, in node e, a higher priority should be assigned to M_i.

In addition, it is impractical to find a simple mapping which can satisfy all the optimization goals. The priority in this paper can only be used to optimize delivery ratio. Therefore, we make decisions as follows: if the bandwidth is insufficient to forward all messages in its local buffer, the node should preferentially replicate the message with highest priority. If the buffer overflows, the node drops the message with the lowest priority, among messages already in local buffer and the new comer. The main notations are illustrated in Table I. The pseudo-code of SDSRP is described in Algorithm 1.

B. Priority Calculation Model

In DTNs, nodes mainly utilize occasional communication opportunities to transmit messages. Therefore, the intermeeting times will seriously influence the delivery ratio. Aiming to solve the problem, we first define the intermeeting time and minimum intermeeting time as follows:

Definition 1: Intermeeting time I is the elapsed time from the end of the previous contact to the start of the next contact between nodes in a pair.

Algorithm 1 SDSRP

Input:
Copies number: C, Remaining TTL: R,
Number of messages in the buffer: n
The ID of new coming message: m
Output:
Scheduling message: ID_S, Dropping message: ID_D

```
for i =1 to n do
    map C_i, R_i to Priority_i
Sort Priority_i incrementally
Find highest Priority_h, and assign h to ID_S
Find lowest Priority_l, and assign l to ID_D
if connection up then
    return ID_S
if buffer overflows then
    map C_m, R_m to Priority_m
    if Priority_m < Priority_l then
        assign m to ID_D
    return ID_D
```

TABLE I. MAIN NOTATIONS USED THROUGHOUT THE PAPER

Symbol	Meaning
N	Total number of nodes in the network
$K_{(t)}$	Number of distinct messages in the network at time t
TTL_i	Initial time-to-live (TTL) for message i
R_i	Remaining time-to-live (TTL) for message i
T_i	Elapsed time for message i since its generation ($T_i = TTL_i - R_i$)
$n_i(T_i)$	Number of nodes with message i in buffer after elapsed time T_i
$m_i(T_i)$	Number of nodes (excluding source) that have seen message i after elapsed time T_i
$d_i(T_i)$	Number of nodes that have dropped message i after elapsed time T_i ($d_i(T_i) = m_i(T_i) + 1 - n_i(T_i)$)
$E(I)$	Mathematical expectation of intermeeting times
λ	Parameter in the exponential distribution of intermeeting times (λ=$\frac{1}{E(I)}$)
$E(I_{min})$	Mathematical expectation of the minimum intermeeting times
λ_{min}	Parameter in the exponential distribution of minimum intermeeting times (λ_{min}=$\frac{1}{E(I_{min})}$)
C	The initial number of copies of message i in source node
C_i	The copies number of message i in the current node
U_i	Priority of message i
$P(T_i)$	Probability that message i has been successfully delivered after elapsed time T_i
$P(R_i)$	Probability that undelivered message i will reach the destination within time R_i
P_i	Probability that message i can be successfully delivered
P	Global delivery ratio

Definition 2: Minimum intermeeting time I_{min} is the minimum elapsed time for a specific node from the end of the previous contact to the start of the next contact with any other node.

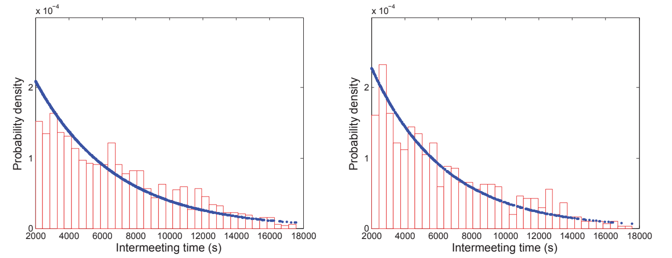
According to the descriptions in Section III-A , the recent researches [22] prove that intermeeting times tail off exponentially in many popular mobilities, such as random walk, random-waypoint, and random direction. Our simulations are based on two scenarios: a synthetic one (the random-waypoint mobility pattern) and real-world trace (EPFL [23], which tracks the taxis in San Francisco over 30 days, we use the data of the first 200 taxis in this paper). We first perform simulations regarding the distribution of the intermeeting times in the aforementioned two scenarios, aiming to examine whether they can fit an exponential distribution.

As can be seen in Fig. 3, the intermeeting times approximately follow an exponential distribution for the above two scenarios: $f(x) = \lambda e^{-\lambda x}$ $(x \geq 0)$. Assume that λ is the parameter for the exponential distribution of intermeeting times and $E(I)$ denotes the mathematical expectation of intermeeting times; then we have $\lambda = \frac{1}{E(I)}$.

There are N nodes in the network: a specific node has a series of intermeeting times ($I_i,\ i \in \{1, 2, 3 \ldots, N{-}1\}$); with other $N{-}1$ nodes, the intermeeting times follow an approximately exponential distribution with the parameter λ. Therefore, the minimum intermeeting time is defined as follows: $I_{min} = Min_{i\in\{1,2,3\ldots,N-1\}} I_i$, which follows an approximate exponential distribution with the parameter λ_{min} (as shown Eq. 3).

$$\lambda_{min} = (N-1)\lambda = \frac{1}{E(I_{min})} = \frac{(N-1)}{E(I)} \tag{3}$$

The delivery probability for message i is given by the probability that message i has been delivered and the probability that message i has not yet been delivered, but will be delivered



(a) (b)

Fig. 3. Intermeeting time distribution for random-waypoint (a) and real-world trace EPFL (b).

during the remaining time R_i. Thus, the delivery ratio P_i can be written as Eq. 4.

$$P_i = (1 - P(T_i))P(R_i) + P(T_i) \tag{4}$$

Suppose that all the nodes, including the destination, have an equal chance of being infected by message i, and the number of nodes that have seen message i is expressed as $m_i(T_i)$, while the source node is not included in $m_i(T_i)$. Therefore, the probability $P(T_i)$ that message i has been successfully delivered can be expressed as Eq. 5.

$$P(T_i) = \frac{m_i(T_i)}{N-1} \tag{5}$$

The equation to calculate $P(R_i)$ (Probability that undelivered message i will reach the destination within the remaining time) is more complex compared with $P(T_i)$. Consider the meaning of $1{-}P(R_i)$, which represents the probability that message i not only has not been delivered at T_i, but also will not be delivered in the remaining time R_i (R_i=$TTL{-}T_i$). In other words, $1{-}P(R_i)$ equals the probability that not only the $n_i(T_i)$ nodes with message i in the buffer will not contact the destination during R_i, but also the new infected nodes will not finish the delivery to the destination within R_i. Moreover, we assume that R_i is long enough to spray all the initial copies. Therefore, the C_i copies of message i will keep infecting $\log_2^{C_i}$ nodes until the number of copies is reduced to 1. In addition, the interval time for the adjacent infections can be estimated as $E(I_{min})$. It means $n_i(T_i)$ new infected nodes will be generated every $E(I_{min})$ time units. So $P(R_i)$ can be expressed as Eq. 6.

$$\begin{aligned} P(R_i) &= 1 - \prod_{k=0}^{\log_2^{C_i}} e^{-\lambda n_i(T_i)[R_i - kE(I_{min})]} \\ &= 1 - e^{-\lambda n_i(T_i)[(\log_2^{C_i}+1)R_i - \frac{1}{2(N-1)\lambda}\log_2^{C_i}(\log_2^{C_i}+1)]} \end{aligned} \tag{6}$$

By combining Eqs. 4 – 6, we obtain the final expression for P_i as Eq. 7.

$$\begin{aligned} P_i = \frac{m_i(T_i)}{N-1} + (1 - \frac{m_i(T_i)}{N-1}) \\ (1 - e^{-\lambda n_i(T_i)[(\log_2^{C_i}+1)R_i - \frac{1}{2(N-1)\lambda}\log_2^{C_i}(\log_2^{C_i}+1)]}) \end{aligned} \tag{7}$$

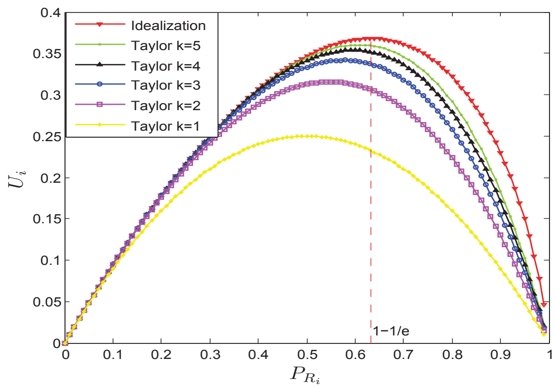


Fig. 4. The functional relationship between U_i and $P(R_i)$.

Note that the global delivery ratio P (expressed as Eq. 8) equals the sum of P_i. According to Eq. 8, we can derive the effect of replicating or dropping a given message i on P. Therefore, ΔP is shown as Eq. 9.

$$P=\sum_{i=1}^{K_{(t)}}[\frac{m_i(T_i)}{N-1}+(1-\frac{m_i(T_i)}{N-1})(1-e^{-\lambda n_i(T_i)[(\log_2^{C_i}+1)R_i-\frac{1}{2(N-1)\lambda}\log_2^{C_i}(\log_2^{C_i}+1)]})] \quad (8)$$

$$\begin{aligned}\Delta P=&\sum_{i=1}^{K_{(t)}}[\frac{\partial P}{\partial n_i(T_i)}\Delta n_i(T_i)]\\ =&\sum_{i=1}^{K_{(t)}}[(1-\frac{m_i(T_i)}{N-1})\lambda[(\log_2^{C_i}+1)R_i-\frac{1}{2(N-1)\lambda}\log_2^{C_i}(\log_2^{C_i}+1)]\\ &e^{-\lambda n_i(T_i)[(\log_2^{C_i}+1)R_i-\frac{1}{2(N-1)\lambda}\log_2^{C_i}(\log_2^{C_i}+1)]}*\Delta n_i(T_i)] \quad (9)\end{aligned}$$

The scheduling and drop strategy proposed in this paper attempts to maximize the delivery ratio. Whenever a given message i is replicated during a contact, the number of nodes with message i in the buffer increases by one $[\Delta n_i(T_i)=+1]$; if no operation is performed on message i, the number of nodes with message i in the buffer remains unchanged $[\Delta n_i(T_i)=0]$; when a copy of message i is dropped from the buffer, the number of nodes with message i in the buffer decreases by one $[\Delta n_i(T_i)=-1]$. Therefore, the priority of message i is precisely the derivative of the delivery ratio P. We obtain the following equation for calculating priority:

$$\begin{aligned}U_i=&(1-\frac{m_i(T_i)}{N-1})\lambda[(\log_2^{C_i}+1)R_i-\frac{1}{2(N-1)\lambda}\log_2^{C_i}(\log_2^{C_i}+1)]\\ &e^{-\lambda n_i(T_i)[(\log_2^{C_i}+1)R_i-\frac{1}{2(N-1)\lambda}\log_2^{C_i}(\log_2^{C_i}+1)]} \quad (10)\end{aligned}$$

Eq. 10 gives us an intuitive feeling regarding the influence to delivery ratio of message copies number and remaining TTL, and how these two parameters map to the message priority. It is worth noticing that the priority calculated by Eq. 10 is not a simple linear combination, but a complex function of the message copies number and remaining TTL; therefore, it leads to a more accurate estimation for message priority. In most cases, a larger number of message copies and remaining TTL indicate that the message has a smaller infection scale, and that these messages should have higher priority. However, there is a possibility that a message has both a large number of message copies and a small remaining TTL - and vice versa. Fortunately, SDSRP can schedule the message priority through Eq. 10, even in the above situation. In addition, we can also find that a greater amount of copies of message i in the network ($n_i(T_i)$) leads to lower priority, which is actually both natural and reasonable.

$$U_i=\frac{(1-P(T_i))(P(R_i)-1)\ln(1-P(R_i))}{n_i(T_i)} \quad (11)$$

To further discover the insight of Eq. 10, with the help of Eqs. 5 and 6, the priority of message i can be expressed with $P(T_i)$ (the probability that message i has been successfully delivered) and $P(R_i)$ (probability that an undelivered message i will reach the destination within time R_i) as shown in Eq. 11. It is easy to find that priority decreases monotonously with delivered probability when other variables are fixed. In other words, higher delivered probability leads to lower priority, which perfectly matches our initial thoughts. Next, when the $P(T_i)$ and $n_i(T_i)$ are fixed, the increase-decrease characteristic of priority (as shown in the Idealization of Fig. 4) depends on the derivative of $(P(R_i)-1)\ln(1-P(R_i))$; results show that when $0\leq P(R_i)<1-1/e$, U_i increases monotonously with $P(R_i)$. Otherwise, when $1-1/e\leq P(R_i)<1$, U_i decreases monotonously with $P(R_i)$. In the analysis, it is necessary to assign a higher priority to messages with higher $P(R_i)$ when the estimated $P(R_i)$ is lower than $1-1/e$; this is for the reason that this approach is helpful for delivering the message. However, it is not suitable to assign higher priority to messages with higher $P(R_i)$ when the estimated $P(R_i)$ is larger than $1-1/e$. This is mainly due to that messages with higher $P(R_i)$ can still be delivered even in lower priority. Aiming to trade off the priority, we assign the highest priority to the messages, whose $P(R_i)$ equals to $1-1/e$ (the peak point in Fig. 4). According to the analysis of Eq. 6, if Eq. 12 is satisfied, then $P(R_i)=1-1/e$. In other words, the messages whose expected encounter time with the destination equals the sum of the remaining TTLs are top-priority. Therefore, the priority used in the paper makes sense.

$$\frac{1}{\lambda n_i(T_i)}=\sum_{k=0}^{\log_2^{C_i}}[R_i-kE(I_{min})] \quad (12)$$

According to Taylor expansion ($\ln(1-x)=-\sum_{k=1}^{\infty}\frac{x^k}{k}$), when $P(R_i)\neq 1$, Eq. 11 can also be expressed in polynomial form (as Eq. 13). With the increase of the terms number k, the priority calculated by Eq. 13 gradually tends to be idealization; Fig. 4 shows the changing process. We can determine the different accuracies as required; simultaneously, computation overhead is also saved through this method.

$$U_i=\frac{(1-P(T_i))(1-P(R_i))\sum_{k=1}^{\infty}\frac{P(R_i)^k}{k}}{n_i(T_i)} \quad (13)$$

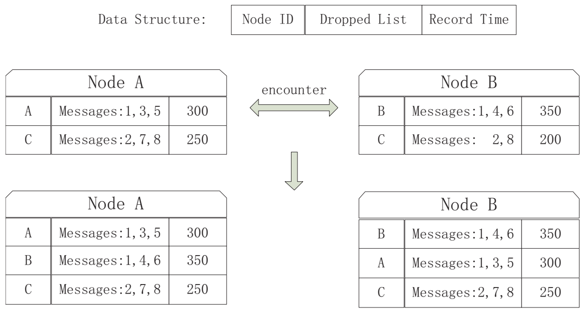


Fig. 5. Data structure and updating process of dropped list.

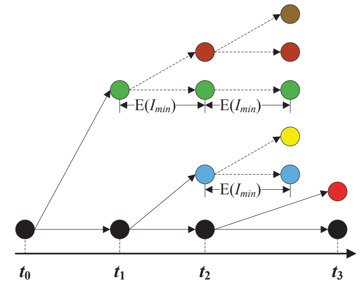


Fig. 6. Binary spray process to estimate $m_i(T_i)$.

Based on the priority calculated by Eq. 10, a successful mapping is established from number of copies (C_i) and remaining TTLs (R_i) to priority (U_i). A scheduling decision which sends the message with the largest priority in advance can be made. At the same time, a drop strategy which drops the message with smallest priority can also be implemented. So far, each node could calculate the priorities of the messages in the buffer. As a result, nodes could schedule the sending order and make the drop decision according to the priorities. It is worth noticing that each node manages its buffer in a distributed fashion, which indicates that each node just cares about the priorities in its own buffer. When two nodes encounter with each other, they simply consider which message to send among the messages in its buffer and which message to drop when overflowing occurs. In conclusion, through the above methods, we achieve a message scheduling and drop strategy on spray and wait routing protocol.

C. Estimation of $m_i(T_i)$ and $n_i(T_i)$

It is obvious that U_i, as illustrated in Eq. 10, is calculable if and only if $m_i(T_i)$ and $n_i(T_i)$ are known. A majority of researchers [12] make a strong assumption that the unknown parameters can be obtained through the centralized control channel. However, the mechanism is difficult to implement in DTNs. According to the definition of $d_i(T_i)$ in Table I, $m_i(T_i)$ and $n_i(T_i)$ can be associated through Eq. 14.

$$n_i(T_i) = m_i(T_i) + 1 - d_i(T_i) \tag{14}$$

The U_i turns to be calculable when $m_i(T_i)$ and $d_i(T_i)$ can be achieved. In order to accurately estimate $d_i(T_i)$, every node maintains a data structure (as shown in Fig. 5) including Node id, Dropped message list, and Record time to collect the information regarding dropped messages. We assume that the size of above data structure could be negligible compared with the message size. The dropped list contains all dropped messages, and the record time is the generation time of the record. When nodes encounter each other, they exchange and update the records in their own as shown in Fig. 5. It is worth noticing that only the source node can modify the record time, which happens if and only if a new drop action occurs in its buffer. When two nodes with the same records encounter each other, a simple update action is implemented according to the record time (updating the record with the nearest record time). Moreover, nodes reject receiving the message already in their dropped lists, which avoids duplication of the dropped action. After a period of time, every node can estimate $d_i(T_i)$.

The estimation method of $m_i(T_i)$ is shown in Fig. 6, which describes the message transmission process of the Spray and Wait routing protocol. During the whole process, we record the time when the message is binary sprayed (i.e., t_0 to t_3). Assuming that the current time is t_3, we can estimate the message transmission process of each node as shown in Fig. 6. Furthermore, we can estimate the value of $m_i(T_i)$.

We assume that the current number of copies for message i is C_i, and the initial number of copies is C, then we can get the height of the tree: $n = \log_2^{C/C_i}$. The solid line in Fig. 6 represents the real transmission process, and the dotted line represents estimated transmission process. Considering that messages are binary sprayed after a period of $E(I_{min})$, we get the estimation for $m_i(T_i)$ as Eq. 15.

$$m_i(T_i) = \sum_{k=1}^{n-1} 2^{\lfloor \frac{t_n - t_k}{E(I_{min})} \rfloor} + 1 \tag{15}$$

To sum up, we develop an estimation strategy to achieve $m_i(T_i)$ and $n_i(T_i)$, furthermore, each node could calculate the messages' priorities utilizing the calculation results of $m_i(T_i)$ and $n_i(T_i)$. Scheduling and drop decisions in terms of buffer-management are made according to the priorities. In order to verify the accuracy of the proposed scheduling and drop strategy, we conduct simulations based on synthetic and real traces in ONE. The results show that, compared with other buffer management strategies, SDSRP achieves higher delivery ratio, similar average hopcounts, and lower overhead ratio.

IV. PERFORMANCE EVALUATION

A. Simulation Setup

Aiming to demonstrate the performance of the proposed SDSRP, an Opportunistic Network Environment (ONE) simulator [24] is employed in this paper. We have carried out simulations using both the synthetic random-waypoint mobility pattern and the real-world trace EPFL [23] (i.e., GPS data of San Francisco taxis). In the former scenario, each node repeats its own behavior, selecting a destination randomly and walking along the shortest path to reach the destination. In the latter dataset, the GPS information of taxis is collected for 30 days, we use the data of the first 200 taxis in this paper (as shown in Fig. 7). Four buffer management strategies (Spray and Wait, Spray and Wait-O, Spray and Wait-C and SDSRP) are implemented in order to compare their performances. Spray and Wait adopts the FIFO (first in first out) buffer management

TABLE II. SIMULATION PARAMETERS UNDER RANDOM-WAYPOINT MOBILITY PATTERN

Parameter	Random-Waypoint
Simulation Time	18000s
Simulation Area	4500m×3400m
Number of Nodes	100
Moving Speed	2m/s
Transmission Speed	250Kbps
Transmission Range	100m
Buffer Size	2MB,2.5MB,3MB,3.5MB,4MB,4.5MB,5MB
Message Size	0.5MB
Message generation rate	[10,15][15,20][20,25]· · · [35,40][40,45][45,50]
TTL	300mins
Initial Copies Number	16,20,24,28,32,36,40,44,48,52,56,60,64

TABLE III. SIMULATION PARAMETERS UNDER REAL-WORLD TRACE EPFL

Parameter	EPFL-Dateset
Simulation Time	18000s
Number of Nodes	200
Transmission Speed	250Kbps
Transmission Range	100m
Buffer Size	2MB,2.5MB,3MB,3.5MB,4MB,4.5MB,5MB
Message Size	0.5MB
Message generation rate	[10,15][15,20][20,25]· · · [35,40][40,45][45,50]
TTL	300mins
Initial Copies Number	16,20,24,28,32,36,40,44,48,52,56,60,64

strategy. Spray and Wait-O regards the ratio between the remaining TTL and initial TTL as the priority. Similarly, Spray and Wait-C treats the ratio between the current message copies number and initial copies number as the priority. In order to reflect the efficiency of proposed buffer management strategy, we set a small buffer size. The detailed simulation parameters are given in Table II.

While a range of data is gathered from the simulation, we take the following three main performance metrics into consideration.

(1) Delivery ratio, which is the ratio between the number of messages successfully delivered to the destination and the total number of messages generated in the network.
(2) Average hopcounts, which is the average number of hops for the successful message delivery from source to destination.
(3) Overhead ratio, which is the ratio between the result of the successfully forwarded message number minus the successfully delivered message number and successfully delivered message number.

B. Simulation Results

1) Performance evaluation under random-waypoint mobility pattern: In the $4500m \times 3400m$ fixed area, we place 100 nodes, whose mobility patterns are random-waypoint. Moreover, the message generation rate is one message per 25-35 seconds; we also set the number of initial copies to 32, and the buffer size to 2.5MB. We vary the initial copies number, buffer size, and message generation rate to examine their impacts on delivery ratio, average hopcounts, and overhead ratio, respectively.

For the first set of simulations, we set buffer size to 2.5MB, and generation rate to one message every 25-35 seconds. The trends of delivery ratio, average hopcounts, and overhead ratio as a function of initial copies number are shown from Fig. 8-(a) to Fig. 8-(c).

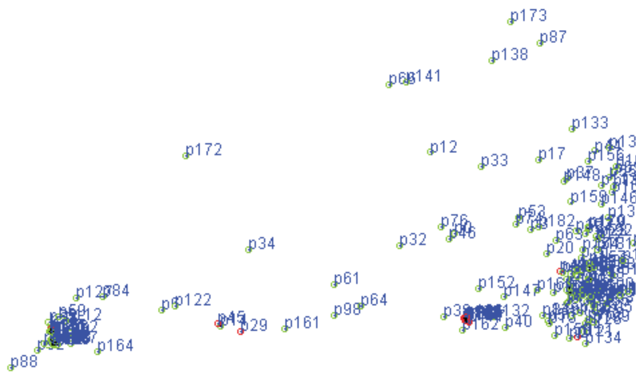


Fig. 7. Real-world movement trace of EPFL.

Fig. 8-(a) shows the changes in delivery ratio over the initial copies number from 16 to 64. The simulation results lead us to the conclusion that the delivery ratio of Spray and Wait-C remains at the lowest level over the period from 16 to 64, compared with other management strategies. Subsequently, this phenomenon becomes more obvious, especially when the initial number of copies is small. In the analysis, the phenomenon is reasonable because a small initial number of copies results in different messages having almost the same number of copies. Therefore, the scheduling and drop strategy is equivalent to the random selection. However, there is a downward trend in the delivery ratio of Spray and Wait-O, along with the growth of the initial number of copies. According to the analysis, the growth of the initial copies number leads to the occurrence of buffer overflow; in other words, the buffer size cannot undertake the overhead in DTNs. In addition, it is worth noticing that the proposed message scheduling and drop strategy SDSRP appears to have a slightly upward trend, and it achieves the best performance regarding delivery ratio. According to the above analysis, we can make a conclusion that SDSRP does a good job facing different initial numbers of copies.

Fig. 8-(b) describes the variation trend of average hopcounts as a function of an initial number of copies. It is easy for us to get the result that Spray and Wait consumes the most average hopcounts to deliver a message. Moreover, there is an upward trend of average hopcounts along with the growth of the initial copies number on Spray and Wait, and it matches our understanding. It is worth noticing that Spray and Wait-C achieves the lowest average hopcounts. It is mainly due to that messages with fewer copies (more hopcounts) are dropped, therefore, all the successfully delivered messages have fewer hopcounts in Spray and Wait-C. However, it is a very pleasant surprise that SDSRP still achieves better performance regarding average hopcounts, compared with Spray and Wait. It is mainly caused by the reasonable scheduling and drop strategy.

Fig. 8-(c) provides some important data regarding overhead ratio performance. Overhead ratio is exploited to measure the amount of effective links; a higher overhead ratio indicates fewer effective links. Therefore, it is not difficult to find that Spray and Wait-C still gets the worst overhead ratio performance, due to unreasonable buffer management. The curve shapes of Spray and Wait and Spray and Wait-O are almost the same. It is worth noticing that SDSRP can achieve

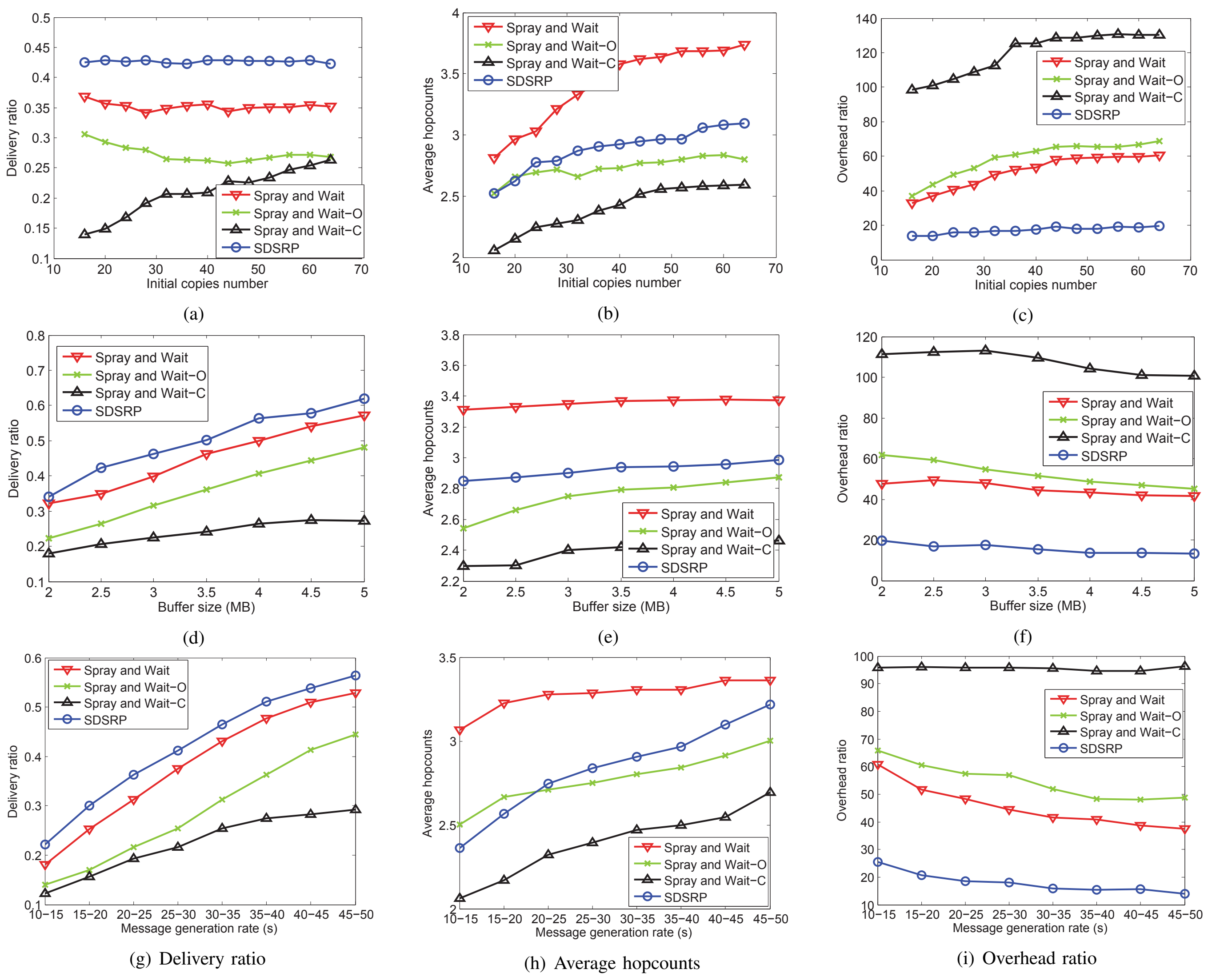


Fig. 8. Delivery ratio, Average hopcounts, and Overhead ratio as a function of initial number of copies, buffer size, and message generation rate under the random-waypoint mobility pattern.

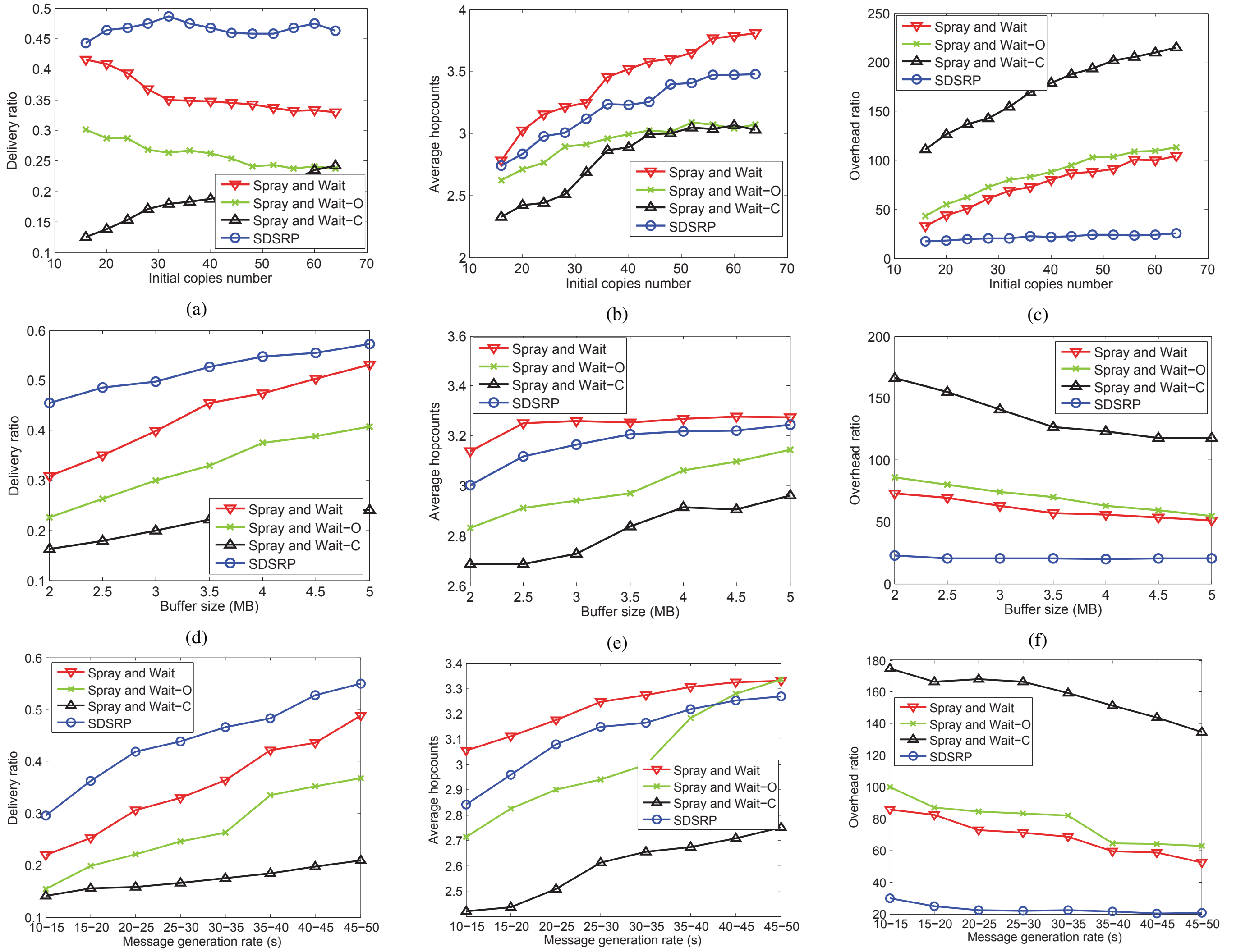
the best overhead ratio performance, and the overhead ratio of SDSRP falls far below that of the other three buffer management strategies.

For the second group of simulations, we set the initial number of copies to 32, and the generation rate to one message per 25-35 seconds. The changes of delivery ratio, average hopcounts, and overhead ratio as a function of buffer size are shown in Fig. 8-(d) through Fig. 8-(f) .

Fig. 8-(d) displays the variation of delivery ratio along with the growth of the buffer size. We can make a conclusion that there are four kinds of upward trends in varying degrees regarding delivery ratio. The tendency of Spray and Wait-C is not obvious. However, there is a significant upward trend over the buffer size from 2MB to 5MB for SDSRP, Spray and Wait. This phenomenon indicates that delivery ratio is sensitive to buffer size, even in a congested network environment. Compared with other buffer management strategies, SDSRP still achieves the best performance. According to the Fig. 8-(e), the change of average hopcounts as a function of buffer size is shown. As can be seen from the graph, over the buffer size from 2MB to 5MB, the average hopcounts of the four buffer management strategies remain level. Moreover, SDSRP still achieves fewer average hopcounts compared with Spray and Wait. Fig. 8-(f) provides some important data of overhead ratio as a function of buffer size. It is worth noticing that there is a potential relationship between Fig. 8-(c) and Fig. 8-(f) for the reason that a larger buffer size indicates that more message copies can be held. Therefore, the curve shapes of Fig. 8-(c) and Fig. 8-(f) are almost inverse. The overhead ratio of SDSRP still remains level, and also achieves the best performance.

Next, in the third group of simulations, we set the initial number of copies to 32, and the buffer size to 2.5MB. The change trends of delivery ratio, average hopcounts, and overhead ratio are plotted as a function of message generation from Fig. 8-(g) to Fig. 8-(i).

Fig. 8-(g) depicts how the delivery ratio varies with the decrease in message generation rate. The notation 10-15 for the message generation rate means that a new message is generated



(a) (b) (c)

(d) (e) (f)

(g) Delivery ratio (h) Average hopcounts (i) Overhead ratio

Fig. 9. Delivery ratio, Average hopcounts, and Overhead ratio as a function of initial number of copies, buffer size, and message generation rate under the real-world trace EPFL.

every 10 to 15 seconds. Thus, the message generation rate decreases with the increasing horizontal axis, resulting in a decrease in congestion. Therefore, there is not a great deal of difference regarding curve shape between Fig. 8-(g) and Fig. 8-(d). The results show that SDSRP outperforms the other buffer management strategies with respect to the delivery ratio regarding different message generation rates. Fig. 8-(h) exhibits the performance of average hopcounts; it reveals the relationship between average hopcounts and message generation rate. As can be seen, message generation rate does not have much influence on average hopcounts. However, SDSRP appears to have a significant improvement along with the decrease of message generation rate; the above phenomenon indicates that reasonable buffer management effectively utilizes the buffer space. At last, Fig. 8-(i) illustrates the changing trend of overhead ratio as a function of message generation rate. The curve shape is similar with that of Fig. 8-(f); it is natural and reasonable for the reason that a lower message generation rate is equivalent to a larger buffer size. SDSRP still outperforms the other buffer management strategies with respect to the overhead ratio. To conclude, compared with the other routing protocols, SDSRP improves the delivery ratio, reduces the average hopcounts and overhead ratio under a random-waypoint mobility pattern.

2) Performance evaluation under real-world trace EPFL: EPFL contains GPS data from the San Francisco taxis acquired over 30 days, we use the data of the first 200 taxis in this paper. We plugged the real-world trace of EPFL into ONE to simulate taxi mobility over the first 18000s.

For the first part of the simulations, we set the buffer size to 2.5MB, and the generation rate to one message per 25-35 seconds. The variation tendencies of delivery ratio, average hopcounts, and overhead ratio as a function of initial copies number are shown from Fig. 9-(a) through Fig. 9-(c). In contrast to the random-waypoint mobility pattern, the movement of the taxis in the real trace lacks regularity and the nodes cannot contact each other as frequently as done in the random-waypoint mobility pattern. However, SDSRP still retains a high delivery ratio while the initial number of copies increases. Thus, it leads us to the conclusion that SDSRP still gets the best delivery performance, even in the EPFL environment. In summary, SDSRP does an excellent job in delivery ratio, average hopcounts, and overhead ratio performances as a function of initial copies number, respectively. The second and third groups of simulations are displayed in Fig. 9-(d) through Fig. 9-(i), which shows the change trends of delivery ratio, average hopcounts, and overhead ratio along with the change of buffer size and message generation rate, separately. It is worth noticing that the curve of Spray and Wait-C in Fig. 9-(i) is different from the one in Fig. 8-(i). In the random-waypoint mobility pattern, the nodes have equal encounter opportunities. Therefore, Spray and Wait-C is equivalent to random selection when the number of copies is small. So the message generation rate has little effect on overhead ratio. However, there is an obvious aggregation phenomenon in the EPFL environment; with the decrease of message generation rate, the useless forwardings also decrease. In conclusion, the scheduling and drop strategy SDSRP effectively solves the congestion problem of Spray and Wait routing in DTNs. In conclusion, either in the random-waypoint mobility pattern or real-world trace EPFL, SDSRP obtains the highest delivery ratio, similar average hopcounts, and lowest overhead ratio regarding different initial numbers of copies, buffer sizes, and message generation rates, compared with Spray and Wait, Spray and Wait-O, and Spray and Wait-C.

V. CONCLUSION

In DTNs, the probabilistic nodal mobility and interruptible wireless links lead to nondeterministic and intermittent connectivity. The store-carry-and-forward paradigm is used by most routing protocols to efficiently deliver messages. However, due to limited storage space, excessive copies of messages easily lead to buffer overflowing. Therefore, how to reasonably allocate network resources becomes significant. In this paper, aiming to improve the delivery ratio, we present a non-heuristic message scheduling and drop strategy on the Spray and Wait routing protocol (SDSRP), which calculates the priority of each message by evaluating the impact of both replicating and dropping a message copy on delivery ratio. Simultaneously, it schedules messages and makes drop decisions according to the priority. We conduct simulations in ONE under the synthetic random-waypoint mobility pattern and the real-world trace EPFL. The simulation results show that, compared with Spray and Wait, Spray and Wait-O, and Spray and Wait-C, SDSRP achieves higher delivery ratio, similar average hopcounts, and a lower overhead ratio.

ACKNOWLEDGEMENT

This work is supported by the National Natural Science Foundation of China under Grant Nos. 61272412, Specialized Research Fund for the Doctoral Program of Higher Education under Grant Nos. 20120061110044, and Chinese Scholarship Council (No. [2014]3026). This work is supported in part by NSF grants CNS 149860, CNS 1461932, CNS 1460971, CNS 1439672, CNS 1301774, ECCS 1231461, ECCS 1128209, and CNS 1138963.

REFERENCES

[1] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. of ACM SIGCOMM 2003*, pp. 27–34.

[2] I. Akyildiz, B. Akan, and C. Chen, "InterPlaNetary internet: state-of-the-art and research challenges," *Computer Networks*, vol. 43, no. 2, pp. 75–112, 2003.

[3] M. Y. S. Uddin, H. Ahmadi, T. Abdelzaher, and R. Kravets, "Intercontact Routing for Energy Constrained Disaster Response Networks," *IEEE Transactions on Mobile Computing*, vol. 12, no. 10, pp. 1986–1998, 2013.

[4] A. Pentland, R. Fletcher, and A. Hasson, "Daknet: rethinking connectivity in developing nations," *IEEE Computer*, vol. 37, no. 1, pp. 78–83, 2004.

[5] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet," in *Proc. of ASPLOS 2002*, pp. 96–107.

[6] M. Xiao, J. Wu, and L. Huang, "Community-Aware Opportunistic Routing in Mobile Social Networks," *IEEE Transactions on Computers*, vol. 63, no. 7, pp. 1682–1695, 2014.

[7] A. Vahdat and D. Becker, "Epidemic Routing for Partially-Connected Ad Hoc Networks," Tech. Rep., Apirl 2000.

[8] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks," in *Proc. of ACM WDTN 2005*.

[9] A. Lindgren and K. S. Phanse, "Evaluation of queuing policies and forwarding strategies for routing in intermittently connected networks," in *Proc. of IEEE COMSWARE 2006*.

[10] D. Kim, H. Park, and I. Yeom, "Minimizing the impact of buffer overflow in dtn," in *Proc. of International Conference on Future Internet Technologies (CFI) 2008*.

[11] E. Wang, Y. Yang, and J. Wu, "A Knapsack-based Message Scheduling and Drop Strategy for Delay-tolerant Networks," in *Proc. of EWSN 2015*, pp. 120–134.

[12] L. Yong and J. Q. Meng, "Adaptive optimal buffer management policies for realistic dtns," in *Proc. of IEEE GLOBECOM 2009*.

[13] X. Zhang, G. Neglia, and J. Kurose, "Performance modeling of epidemic routing," *Computer Networks*, vol. 51, no. 10, pp. 2867–2891, 2007.

[14] A. Elwhishi, P. Ho, K. Naik, and B. Shihada, "A Novel Message Scheduling Framework for Delay Tolerant Networks Routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 5, pp. 871–880, 2013.

[15] A. Krifa and C. Barakat, "Optimal buffer management policies for delay tolerant networks," in *Proc. of IEEE SECON 2008*, pp. 260–268.

[16] ——, "An optimal joint scheduling and drop policy for delay tolerant networks," in *Proc. of IEEE WoWMoM 2008*, pp. 1–6.

[17] ——, "Message Drop and Scheduling in DTNs: Theory and Practice," *IEEE Transactions on Mobile Computing*, vol. 11, no. 9, pp. 1470–1483, 2012.

[18] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility," in *Proc. of IEEE PerCom Pervasive Computing and Communications Workshop 2007*.

[19] Y. P. Kim, J. I. Koo, and E. Jung, "Composite methods for improving spray and wait routing protocol in delay tolerant networks," in *Proc. of IEEE ISCIT 2010*, pp. 1229–1234.

[20] M. Shahid and L. Asif, "Multishceme spray and wait routing in delay tolerant networks by exploiting nodes delivery predictability," in *Proc. of IEEE ICCIT 2012*, pp. 255–260.

[21] W. Gao, Q. Li, and G. Cao, "Forwarding redundancy in opportunistic mobile networks: Investigation and elimination," in *Proc. of IEEE INFOCOM*, 2014.

[22] G. Robin, N. Philippe, and K. Ger, "Message delay in manet," in *Proc. of ACM SIGMETRICS 2005*, pp. 412–413.

[23] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, "CRAWDAD data set epfl/mobility (v. 2009-02-24)," Downloaded from http://crawdad.cs.dartmouth.edu/epfl/mobility.

[24] A. Keränen, J. Ott, and T. Kärkkäinen, "The one simulator for dtn protocol evaluation," in *Proc. of ICST 2009*.