A Statistical Analysis and Temporary Cluster based Routing Algorithm for Delay Tolerant Networks

Jixing Xu
Information Engineering College of Qingdao University, Ningxia Road 308, Qingdao, China
E-mail: ytxujixing@gmail.com

*Jianbo Li, Lei You and Chenqu Dai
Information Engineering College of Qingdao University, Ningxia Road 308, Qingdao, China
E-mail: {lijianboqdu, youleiqdu, daichenqu}@gmail.com

***Abstract*—Delay Tolerant Networks are characterized by frequent network topology partition, limited resource, extremely high latency, etc. Consequently successful deliveries of messages in such networks face great challenges. In this paper, by capturing the temporary clusters, we timely use the temporary end-to-end paths to directly and successfully deliver messages. Besides, by collecting a large number of encounter history information as the samples, we use the methods of statistical analysis to objectively and accurately evaluate encountered nodes, thus selecting fewer but better relay nodes to spread messages. Finally based on the above schemes, a Statistical Analysis and Temporary Cluster based routing algorithm (SATC) is proposed to improve the routing performance. Extensive simulations have been conducted and the results show that SATC can achieve a higher delivery ratio and a fewer hop count compared to Epidemic and PRoPHET. Furthermore, its overhead ratio is 70% and 65% less than Epidemic and PRoPHET respectively.**

***Index Terms*—Delay Tolerant Networks; network topology partition; temporary cluster; encounter history; statistical analysis**

I. INTRODUCTION

Delay Tolerant Network [1-3] as a new end to end store and forward network architecture, which originates from the Interplanetary Internet [4] and features frequent network topology partition [5], node mobility, limited device capability and extremely high latency, has become a hot research interest and a great challenge in the field of wireless network. In 2003, Kevin Fall first proposed the concept of DTN. Soon afterwards, the Internet Research Task Force (IRTF) specially set up the DTN Research Group (DTNRG) to study it. Finally in 2007, DTNRG put forward the DTN network architecture [6], which introduced a bundle layer between the application layer and the transport layer (as shown in Fig. 1) to communicate across multiple regions that have different types of networks and different protocols. Now DTN as a emerging concept has been widely studied and applied.

Initially, DTN is mainly used in the military battlefield networks and some special application scenarios (e.g. disaster relief, etc). Now, DTN is widely applied to various civilian areas, such as wildlife tracking networks [7], pocket switched networks [8], habitat monitoring networks [9], underwater sensor networks [10], vehicular ad hoc networks [11-13], etc. These special networks deployed in challenging environment may never have a complete end-to-end path between the sender and the receiver [14]. So the traditional routing protocols based on TCP/IP are difficult to get efficient achievements because that they need to first find a complete path to the destination node before starting the transmission. In order to cope with this problem and finish end-to-end communication, DTN routing adopts store-carry and forward strategy to relay messages hop by hop. But in most DTNs, it may be very difficult to capture the global network topology knowledge and the route information about the final receiver, so how to select optimal next hop relay nodes to move closer to the final destination node is the key issue.

A simple solution is to add the number of message copies by using flooding strategy and relay these message copies to different encountered neighbors, which can greatly increase the opportunity to encounter the final destination node. Reference [15] has proven that source node can enhance network capacity by replicating more message copies to different relay nodes. In case of sufficient network resources and enough network load capacity, it may be the best way to improve message delivery ratio. But in most DTNs, the network resource and device capability are limited. The more message copies mean more consumption of resources and more serious message redundancy. In this case, uncontrolled flooding strategy is still difficult to get good routing performance. Consequently, the key issue of efficient

Manuscript received November 14, 2013; revised February 7, 2014; accepted March 5, 2014.

*Corresponding Author

doi:10.4304/jsw.9.9.2291-2301

routing strategy is to select fewer but better neighbors as next hop relay nodes to spread message copies, thus implementing controlled infection and getting a good balance between higher message delivery ratio and less resource consumption. For this purpose, we need to accurately estimate whether a neighbor should be selected as the next hop relay node. Many scholars have tried to capture the global network topology and additional information (e.g. geographical location information, etc), but the reliability of those information is greatly reduced due to node's mobility and the extremely high end-to-end latency. On the contrary, the basic history information which can be captured in the meeting time of node pairs has a more practical value.

Based on the above considerations, we propose two approaches to estimate whether a neighbor is a good next hop relay node. Firstly, if there is a temporary path to the final destination node among a current temporary cluster, all of these nodes which exist in the path are the optimal relay nodes. This is because that these nodes can quickly and successfully finish this transmission. Secondly, when there is no temporary path to the destination node, by collecting a large number of encounter history information as the samples, we use the methods of statistical analysis to compute several statistics which can be used to objectively and accurately evaluate every encountered node. Then we only select the neighbors which can help to improve routing performance as the next hop relay nodes. Finally, a Statistical Analysis and Temporary Cluster based routing algorithm (SATC) is proposed to improve routing performance.

The rest of this paper is organized as follows. In section 2, we discuss some related works. Section 3 gives a detailed description of SATC scheme. The performance evaluations and comparisons among SATC, Epidemic and PRoPHET are presented in Section 4. Finally, section 5 summarizes this paper and gives future research directions.

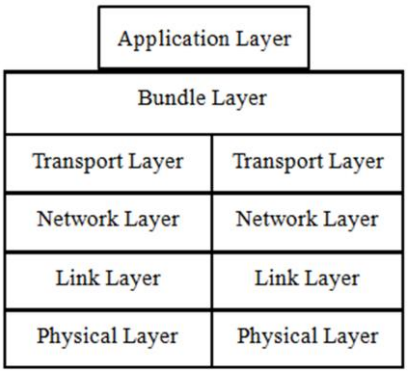


Figure 1. DTN network architecture

II. Related Works

Combining with the research achievements of DTNRG, scholars have proposed a lot of typical DTN routing schemes. A simple taxonomy of DTN unicast routing techniques is illustrated in Fig. 2.

By introducing position fixed throw boxes [16] [17], scholars have proposed some fixed infrastructures assistant routing schemes. Analogously, with the help of mobility pattern fixed ferry nodes, references [18-20] propose some mobile infrastructures assistant routing algorithms. Those DTN routing techniques based on infrastructures can greatly improve message delivery ratio, but they also inevitably increase the complexity of entire network.

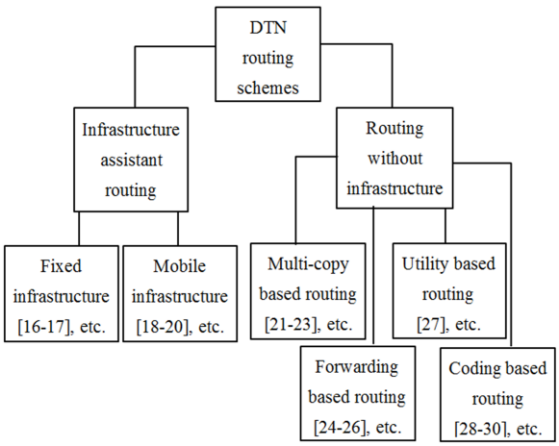


Figure 2. A taxonomy of DTN unicast routing schemes

Epidemic [21] is a typical multi-copy based routing algorithm, which makes attempts to replicate all carried messages to all encountered neighbors by using flooding strategy. The biggest advantage is that Epidemic does not need to capture global network topology knowledge, so its routing complexity is low. But in order to get a high message delivery ratio, it does need a large number of resources and a strong network load capacity. In other words, it is difficult to get a good routing performance in case of insufficient resource. Spray and Wait [22] and Spray and Focus [23] are both based on multi-copy scheme, but they greatly limit the number of message copies. Only in Spray period can they replicate L message copies to L different relay nodes. So they can effectively control the consumption of network resource. However, because that they do not take any measures to evaluate encountered neighbors when selecting next hop relay nodes, their improvements on message delivery ratio are not satisfying.

Relying on knowledge oracles, reference [24] proposes six different forwarding based routing schemes: FC, MED, ED, EDLQ, EDAQ and LP. According to the amount of available knowledge—zero knowledge, partial knowledge and complete knowledge, these algorithms can get different routing performance. Besides, respectively based on ED and MED, AED [25] and MEED [26] make some improvements, thus achieving better routing performance.

PRoPHET [27] is a typical routing scheme based on history utility, which uses the encounter history information and transitivity to predict the delivery possibility between two nodes. Then PRoPHET only selects the neighbors with bigger delivery possibilities than current node as the next hop relay nodes. So

compared with the Epidemic, PRoPHET can greatly improve the routing performance. But when two nodes encounter, PEoPHET only subjectively updates the delivery possibility according to some pre-set parameter values. It can't predict the actual delivery possibility, so there may be some errors relative to the actual delivery possibility.

Some coding based routing algorithms are presented in [28-30], which effectively reduce the consumption of network resource and improve the routing performance to some extent, but also inevitably increase the complexity of routing protocol.

III. SATC ROUTING FRAMEWORK

In DTN routing protocols [31-33], multi-copy strategy can greatly improve the message delivery ratio. But due to limited network resource and restricted device capacity, blind flooding may cause the opposite results. So in this section, we propose two routing schemes to accurately evaluate every encountered node, thus selecting fewer but better relay nodes to spread messages and greatly controlling message redundancy. Finally, we propose a Statistical Analysis and Temporary Cluster based routing algorithm (SATC) to efficiently improve the routing performance. The SATC is based on the following assumptions:

- DTN nodes are initiative to collect encounter history information in every meeting chance.
- The size of the broadcast packet is very small compared to the data message.

A. Routing Scheme based on Temporary Clusters

In most DTN application scenarios, because of spare node density and node mobility, it is very difficult to keep a complete end-to-end path between the sender and the receiver. But the case that some moving nodes form a temporary cluster (as shown in Fig. 3) may often occur. In temporary cluster C1, if relay node *a* is carrying a message *Mi* for destination node *d*, theoretically node *a* can quickly and successfully transfers *Mi* to destination node *d* by using the temporary path *a*-*b*-*d*. But node *a* can only know its one-hop neighbor *b*. So when the unfortunate case that node *a* does not think node *b* is a good next hop relay node occurs, node *a* will eventually miss this opportunity to successfully deliver *Mi*.

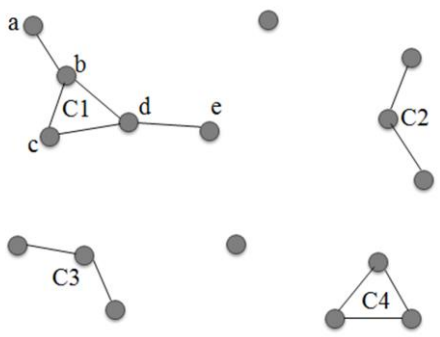


Figure 3. Temporary clusters: C1, C2, C3 and C4

From the above instance we can see that the establishment of the temporary cluster can greatly increase the searching range to find the final destination node, instead of only searching in current node's one-hop neighbors. It is out of this consideration, we try to make full use of these temporary clusters. In order to find all cluster members, current node *s* needs to broadcast a Local Route Discovery Packet (LRDP) to the temporary cluster. A LRDP contains the following information:

- source _id
- broadcast_id
- flag
- hop_count
- path
- time-to-live

The broadcast_id is incremented when source node creates a new LRDP, so the pair <source_id, broadcast_id> can uniquely identify a LRDP. The flag only has two possible values: 0 and 1. The 0 represents that the packet is a route discovery packet, and the 1 indicates that the packet is a route reply packet. So when a node needs to reply a LRDP (flag=0), it just needs to modify the flag (flag=1) and then unicasts the LRDP to the source node, not having to generate a new reply packet. The path stores all intermediate nodes from the source node to a temporary cluster member, and the order that the node is added into the path is the node's forwarding order along the route. So the pair <source_id, broadcast_id, path> can uniquely identify a temporary path to a cluster member, and we can efficiently avoid route loop by detecting whether there are duplicate nodes in the path. The hop_count is the length of the path. The time-to-live is the total time that a LRDP can survive, which is set to an appropriate value to get all LRDP reply packets and is gradually decreased over time.

When a node needs to deliver some messages, but there are no routes to the destination nodes, a route discovery process is initiated. The node creates a new LRDP and broadcasts the LRDP to its all one-hop neighbors. When a neighbor receives the LRDP, it first needs to rebroadcast the LRDP to its own one-hop neighbors after increasing the hop_count and adding itself into the path, and then it also needs to reply the LRDP by unicasting the LRDP to the source node along the reverse path after modifying the flag (flag=1). A cluster node may receive different copies of the same LRDP from various neighbors. In this case, it still rebroadcast the LRDP to its one-hop neighbors. The purpose is for the consideration that the currently existing path may be interrupted in the process of establishing the route due to node mobility. So it may eventually fail to get the final route if we only keep track of one single path, ignoring other possible paths. In DTNs, due to sparse node density, the size of a temporary cluster won't be too big. So the overhead caused by LRDP broadcast can be controlled. The detailed process of a LRDP on a cluster member Ni is shown in Fig. 4. Lines 1-15 are the process of receiving a route discovery packet. Lines 2-3 and lines 9-10 can efficiently avoid the broadcast loop. Line 17 is the process of receiving a route reply packet.

Now Ni needs to go on unicasting the LRDP to the source node along the reverse path.

As a LRDP travels from source node to a cluster member, it automatically sets up the route. Eventually when receiving all LRDP reply packets from cluster members, source node can construct a temporary route table. A route entry contains the following information:

- destination_node
- path
- hop_count
- path_capacity
- time_out

Algorithm 1: LRDP broadcast process on cluster member Ni

Input:

one-hop neighbors of Ni: *neigh_list*

current broadcast packet: *LRDP*

output:

```
1.  If  LRDP.flag==0
2.      If LRDP.path.contains(Ni)
3.          drop LRDP;
4.      Else
5.          LRDP.hop_count++;
6.          LRDP.path.add(Ni);
7.          initialize bro_list to null;
8.          For every node in neigh_list
9.              If node.id != LRDP.source_id  &&
                ! LRDP.path.contains(node)
10.                 add node into bro_list;
11.         End for
12.         broadcast LRDP to bro_list;
13.         LRDP.flag=1;
14.         unicast the LRDP to its source node along
            the reverse path of LRDP.path;
15.     End if
16. Else
17.     go on unicasting LRDP to its source node
        along the reverse path of LRDP.path;
18. End if
```

Figure 4. LRDP broadcast process on cluster member

There may be multiple paths between source node and a cluster member. So in this case, we choose the shortest path to construct the route. Now there is a new issue: how to define the shortest path? Here, we use the two metrics: path capacity and hop count. The path capacity is defined in (1). The i(TTL) represents the initial time-to-live assigned to a new LRDP and the r(TTL) gives the remaining time-to-live of the LRDP when source node receives the reply packet. Then the half of the round trip time (i.e. $\frac{i(TTL)-r(TTL)}{2}$) can objectively reflect the current path capacity, and the less time represents a better path capacity. In (1), we use the reciprocal of $\frac{i(TTL)-r(TTL)}{2}$ to define the final C_{path}, so that the less time can get a bigger C_{path} value. Finally, we define the shortest path as follows. If the hop counts of the paths are all bigger than $hop_{threshold}$, we select the path with the best path capacity as the shortest path. Else we select the path with the shortest hop count as the shortest path. The detailed process of constructing route table is shown in Fig. 5.

$$C_{path} = \frac{1}{\frac{i(TTL)-r(TTL)}{2}} \quad (1)$$

Algorithm 2: Construct route entry on source node S

Input:

route table of S: *routeTable*

current route reply packet: *LRDP*

the threshold of hop count: $hop_{threshold}$

output:

```
1.  destination = the last node in LRDP.path;
2.  compute path_capacity of LRDP.path ;
3.  If  routeTable has a route R to destination
4.      If  R.hop_count >hop_threshold  &&
            LRDP.hop_count >  hop_threshold
5.          If  path_capacity> R.path_capacity
6.              reconstruct route R using LRDP;
7.          Else if path_capacity== R.path_capacity
8.              If LRDP.hop_count<R.hop_count
9.                  reconstruct route R using LRDP;
10.             End if
11.         End if
12.     Else
13.         If  LRDP.hop_count<R.hop_count
14.             reconstruct route R using LRDP;
15.         Else if LRDP.hop_count==R.hop_count
16.             if path_capacity> R.path_capacity
17.                 reconstruct route R using LRDP;
18.             End if
19.         End if
20.     End if
21. Else
22.     create a new route using LRDP;
23.     add the route into routeTable;
24. End if
```

Figure 5. The process of constructing route table

When creating and updating a route, the time_out is

set to an appropriate time. And the route will be deleted when the time is exhausted. Eventually, source node S can get the routes to all cluster members. If there is a message for a discovered cluster member, then node S encodes the relevant route entry into the message and forwards the message to the relevant next hop node along the path stored in the route entry.

B. Routing Scheme based on Statistical Analysis

Most DTN nodes usually do not move around completely randomly, and their mobility tends to be regular in some ways. On the one hand, if two nodes have encountered frequently in the past, it is likely that they will meet again in the future. On the other hand, DTN node may do periodic activity, and it may pass the same place in the same time of different periods. An example where may demonstrate such periodic mobility is, for example, a social networks, where most people go to work and come home in a regular time on every workday and most students regularly move among campus. So the history information can greatly help to estimate whether a neighbor is a good next hop relay node which can improve message delivery ratio. Based on the observations, we can define the node activity cycle according to specific application scenarios and divide the activity cycle into different time units. For example, in social networks, people's activity cycle is usually one day and couple hours can be treated as a time unit. Then by collecting the encounter count information between two nodes in the time units of recent several cycles, we use the methods of statistical analysis to compute several statistics, which can be used to objectively evaluate every encountered node.

For this purpose, a DTN node needs to maintain an N × M dynamic matrix (2) to record the encounter history information with other N nodes. M is equal to L×P, where P represents the number of the cycles and every cycle is divided into L different time units. The $C_i(\mathrm{j},\mathrm{k})$ records the encounter count between node i and node j in the k^{th} time unit. Then for the last M time units, all encounter history information are stored in the DM_i.

$$DM_i = \begin{bmatrix} C_i(1,1) & \cdots & C_i(1,\mathrm{k}) & \cdots & C_i(1,\mathrm{M}) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ C_i(\mathrm{j},1) & \cdots & C_i(\mathrm{j},\mathrm{k}) & \cdots & C_i(\mathrm{j},\mathrm{M}) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ C_i(\mathrm{N},1) & \cdots & C_i(\mathrm{N},\mathrm{k}) & \cdots & C_i(\mathrm{N},\mathrm{M}) \end{bmatrix} \quad (2)$$

With the DM_i, we define the several statistics as the follows.

(1) Average (AVG)

The $P_i(j,k)$ shown in (3) represents the probability that $C_i(j,k)$ may occur, and in this case it is a fixed value. Then the AVG is defined in (4).

$$P_i(j,k) = \frac{1}{M} \quad k = 1,2,3 \cdots M \quad (3)$$

$$\mathrm{AVG(i,j)} = \sum_{k=1}^{M} C_i(\mathrm{j},\mathrm{k}) \times P_i(j,k) = \frac{\sum_{k=1}^{M} C_i(\mathrm{j},\mathrm{k})}{M} \quad (4)$$

The AVG is the average level of all samples which can objectively reflect the central tendency. So $\mathrm{AVG}(i,d)$ can accurately describe the meeting frequency level between node i and destination node d. In this case, a neighbor with a bigger AVG than current node is generally considered more likely to meet the destination node again in the near time unit.

(2) Cycle average (CAVG)

In most DTN scenarios, for example, in the social networks, it is possible that DTN nodes have different mobility patterns in different time units of the same activity cycle and have similar mobility patterns in the same time unit of different activity cycles. So in order to describe the similar mobility pattern in the same time unit of different activity cycles, we define the cycle average to compute the average encounter count between two nodes in the same time unit of different activity cycles. For node i and node j, equation (5) can compute the summation of the encounter counts in every l^{th} time unit of P different activity cycles. Then the CAVG is defined in (6). A neighbor with a bigger CAVG than current node is more likely to meet the destination node again in the l^{th} time unit of the current cycle.

$$SUM_{i,j}(l) = \sum_{\substack{1 \le k \le \boldsymbol{M} \\ k \bmod \boldsymbol{L} = l}} C_i(j,k) \qquad l = 1,2,\cdots \boldsymbol{L} \quad (5)$$

$$\mathrm{CAVG}(i,j,l) = \frac{SUM_{i,j}(l)}{\boldsymbol{P}} \quad (6)$$

(3) Mode (MODE)

The mode is the data which appears most frequently among all samples, so it can reflect the most possible level. In this case, we also use the mode as the metric to evaluate encountered nodes. The $MODE(i,d)$ denotes the mode of the all encounter counts between node i and node d. A neighbor with a bigger MODE than current node is more likely to meet the destination node again.

(4) Variance (VAR)

The VAR defined in (7) can comprehensively reflect the degree of dispersion of all samples, which is often used to describe the stability of data. A smaller VAR indicates that the samples can more accurately reflect the general level.

$$\mathrm{VAR(i,j)} = \sum_{k=1}^{M} \left(C_i(\mathrm{j},\mathrm{k}) - \mathrm{AVG(i,j)}\right)^2 \times P_i(j,k) = \frac{\sum_{k=1}^{M} (C_i(\mathrm{j},\mathrm{k}) - \mathrm{AVG(i,j)})^2}{M} \quad (7)$$

Finally with the above statistics, we can objectively estimate whether a neighbor should be selected as the next hop relay node. For the current node s, the neighbor node j and the destination node d, we firstly take into account AVG. By using (8), we can compute the utility value U_{avg}. When $U_{avg}(s,j,d) \ge threshold_{avg}$, which indicates that the neighbor j has an obvious advantage on AVG compared to the current node s, then the neighbor j should be selected as the next hop relay node. Secondly, we take into account CAVG and MODE. Assuming that the current time unit is the l^{th} time unit, we use (9) to compute another utility U_2. When $U_2(j,d,l) > U_2(s,d,l)$, which indicates that node j is more likely to meet the destination node d in the current time unit, so the neighbor j should be selected as the next hop relay node, too. The detailed algorithm is show in Fig. 6. In addition, the communication time between two nodes is

restricted due to node mobility, so it is very possible that current node can't copy all messages to all selected neighbors. As a result, the delivery order of messages may also affect the routing performance. So as shown in lines 14-15 of algorithm 3, we give the tuple that has a smaller VAR a higher priority and start transferring tuples according to the priority.

$$U_{avg}(s,j,d) = \frac{\mathrm{AVG(j,d)}}{\mathrm{AVG(s,d)}} \quad (8)$$

$$U_2(i,d,l) = \mathrm{CAVG}(i,d,l) \times \beta + MODE(i,d) \times (1-\beta) \quad (9)$$

Algorithm 3: Statistical analysis based routing on Ni

Input:

the list of messages: msg_list

one-hop neighbors: neigh_list

the value of $threshold_{avg}$: $threshold_{avg}$

the current time unit of the current cycle: tu

output:

1. initialize *tuple_list* to null;
2. For each *message* in *msg_list*
3. *destination=message.destination*;
4. For each *neighbor* in *neigh_list*
5. If $U_{avg}(Ni, neighbor, destination)$ >= $threshold_{avg}$
6. Add *tuple<message,neighbor>* into *tuple_list*;
7. Else
8. If $U_2(neighbor, destination, tu)$>= $U_2(Ni, destination, tu)$
9. Add *tuple<message,neighbor>* into *tuple_list*;
10. End if
11. End if
12. End for
13. End for
14. Sort *tuple_list* in increasing order according to the relevant *VAR* value of every tuple;
15. Try every *tuple* according to the order;

Figure 6. Statistical analysis based routing scheme

C. Buffer Management

In DTNs, due to the limited buffer resource, the forwarding order of messages and the deletion strategy may also affect the routing performance. So we also take into account buffer management strategy. According to the above different routing schemes, buffer is divided into two parts as illustrated in Fig. 7.

The left part has a higher priority and messages are first transmitted from here. The two types of messages: the broadcast packets and the messages with temporary routes to their destinations should be stored in the left part buffer. In addition, these messages are sorted according to the FIFO order.

The right part has a lower priority, and messages are deleted first from here when buffer overflows. The messages without route information to their destinations are stored in the right part buffer, and messages are also sorted according to the FIFO order.

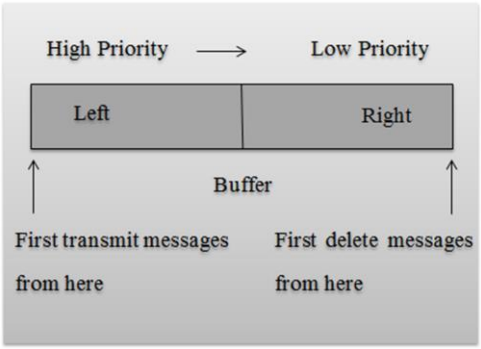


Figure 7. Buffer management

D. SATC

Finally, combining with the above schemes, we finally implement the Statistical Analysis and Temporary Cluster based routing algorithm (SATC). The detailed algorithm is shown in Fig. 8.

Lines 1-13 are the processing procedure of the messages stored in the left part buffer. Lines 2-3 use algorithm 1 to go on broadcasting a LRDP. In lines 5-7, according to the relevant route entry stored in the message, Ni goes on forwarding the message to the relevant next hop node. In lines 9-10, when the relevant next hop node is not within the scope of Ni's one-hop neighbors, which indicates that the current route is broken due to node mobility, then Ni deletes the current route entry from the message and moves the message to the right part buffer. Lines 14-22 are the processing procedure of messages stored in the right part buffer. Lines 14-15 use algorithm 2 to reconstruct a new route table. Lines 16-21 find the messages which can be transferred to the destination nodes by using the current route table. And then Ni encodes the relevant route entry into the corresponding message and forwards the message to the next hop node along the route. Finally, line 22 uses algorithm 3 to select fewer but better neighbors to spread the remaining messages.

IV. SIMULATION

In this section, we use the ONE simulator [34] to implement our SATC algorithm. The ONE is an Opportunistic Network Environment simulator which provides a powerful tool for generating mobility traces, running DTN messaging simulations with different routing protocols, and visualizing both simulations interactively in real-time and results after their completion. At its core, ONE is an agent based discrete

event simulation engine, which is mainly made up of five modules: movement models, event generators, routing, visualization and results. A detailed description of the modules and their interactions is shown in Fig.9. At each simulation step the ONE engine updates a number of modules that implement the main simulation functions. The ONE platform has a good scalability and facilitates the further development of the users.

Algorithm 4: SATC routing scheme on node Ni

Input:

messages in left part buffer: msg_left

messages in right part buffer: msg_right

one-hop neighbors: neigh_list

output:

1. For *message* in msg_left
2. If *message* is a LRDP broadcast packet
3. execute Algorithm 1;
4. Else
5. Get the relevant *next_hop_node* of *message* according to the relevant route entry stored in the *message*;
6. If *next_hop_node* is in *neigh_list*
7. forward *message* to *next_hop_node;*
8. Else
9. delete the relevant route from *message*;
10. move *message* to *msg_right*;
11. End if
12. End if
13. End for
14. broadcast a new LRDP to *neigh_list*;
15. create temporary route table *RT* using Algorithm 2;
16. For *message* in *msg_right*
17. If *RT* has a route *R* to *message.destination*
18. encode *R* into *message*;
19. forward *message* to its next hop node;
20. End if
21. End for
22. execute algorithm 3 on the remaining messages of *msg_right*;

Figure 8. The process of SATC on node Ni

In order to carry out simulations, we use 126 DTN nodes in a Helsinki City Model based mobility scenario which consists of $4500 \times 3400\ m^2$ area. These nodes are divided into 6 groups. Group 1 and Group 3 are pedestrians groups and they consist of 40 nodes separately. Group 2 is the car group which also consists of 40 nodes. Group 4, Group 5 and Group 6 are tram groups and they consist of two nodes separately. Pedestrians move with speeds of 0.5–1.5 m/s, cars move with speeds of 2.7 – 13.9 m/h, and trams move with speeds of 7 – 10 m/h. Two types of devices are introduced in the simulation: one is Bluetooth Device with transmission speed of 250 kBps and transmission range of 20m; other is High Speed Device with transmission speed of 10 MBps and transmission range of 1000m. Group 1, Group 2 and Group 3 have a 5MB buffer respectively and they are all based on the Shortest Path Map Based movement model. Group 4, Group 5 and Group 6 have a 50MB buffer respectively and they are all based on the Route Map Based movement model. Group 1,Group 2, Group 3, Group 5 and Group 6 use Bluetooth Device, Group 4 uses both Bluetooth device and High Speed Device. Besides, the other simulation settings of 126 nodes are shown in Table1. And the special parameters of SATC are: time-to-live = 1000 msec (in LRDP), time_out = 3000 msec (in route entry), $hop_{threshold}$ = 2 (in algorithm 2), M = 24×7, P = 7, L = 24, $\beta = 0.5$, $threshold_{avg}$ = 2.0 (in algorithm 3). In the beginning of the simulations, we allow a warm up period of 7 days to move nodes for the purpose of collecting enough encounter history information.

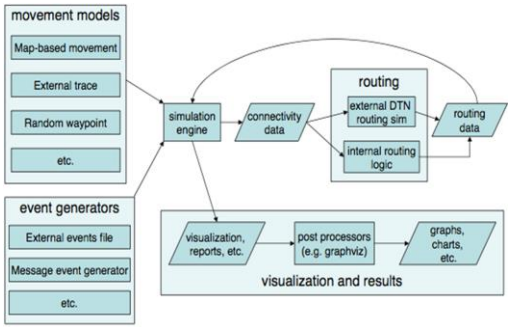


Figure 9. Overview of the ONE simulator[34]

TABLE I.
THE SAME SIMULATION SETTINGS OF 126 NODES

Parameter	Default value
Initial topology	Uniform
Message size	500K
Message interval	40s
Time-To-Live(TTL)	4 hours
Simulation time	12 hours

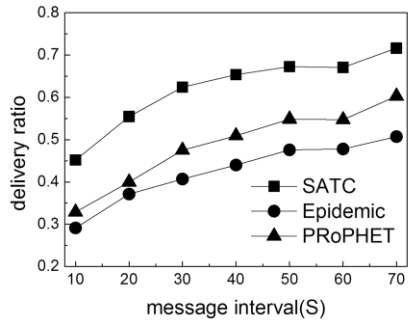
Finally, extensive simulations have been conducted to compare routing performance of SATC, Epidemic and PRoPHET in terms of message delivery ratio, network overhead ratio and average hop count. We mostly focus on their different routing performance in different message interval, message time-to-live and message size.

A. *Vary Message Interval*

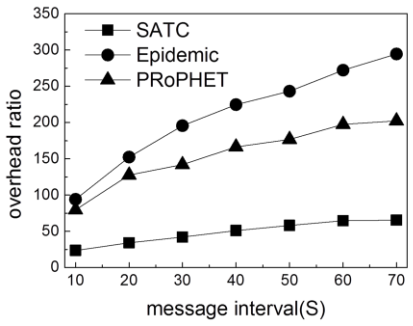
Fig. 10 describes the different simulation results of varying message interval. SATC can get the best routing performance in terms of message delivery ratio, overhead ratio and average hop count, which can validate that the

routing strategy of SATC is more efficient and can greatly improve the transmission performance of the entire network.

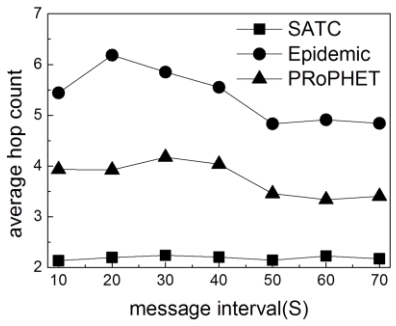
Epidemic does not take any strategy to control message redundancy, thus inevitably creating a large number of redundant messages in whole network. When buffer resource is insufficient, Epidemic will discard a lot of messages, reducing the utilization efficiency of buffer resource and greatly increasing the network overhead ratio. As shown in Fig. 10, the routing performance of Epidemic is the worst among the three routing schemes. In addition, Epidemic does not estimate whether a neighbor is a good relay node, but just flooding messages to the entire network. So it can't guarantee the accuracy of routing strategy. As shown in Fig. 10(c), the average hop count of Epidemic is the most, increasing the cost of message transmission.



(a) delivery ratio



(b) overhead ratio



(c) average hop count

Figure 10. The delivery ratio, overhead ratio, average hop count VS message interval

Different from Epidemic, PRoPHET uses the encounter history information and transitivity to predict the delivery possibility between two nodes and only selects the neighbor with a bigger delivery possibility as next hop relay node. So compared to Epidemic, PRoPHET improves the message delivery ratio, lowers overhead ratio and decreases the average hop count. But when two nodes encounter, PEoPHET only subjectively updates the delivery possibility according to some pre-set parameters, which can't accurately predict the actual delivery possibility. So its routing performance is still worse than SATC.

Our SATC not only uses the temporary cluster to directly and quickly deliver message to the final destination node, but also takes the methods of statistical analysis to objectively and accurately estimate whether a neighbor should be selected as next hop relay node. As a result, SATC can greatly improve message delivery ratio in the case of controlling message redundancy, lowering the network overhead ratio and reducing the consumption of resource. As shown in Fig. 10, SATC can get the best routing performance. The message delivery ratio of SATC is 40% and 20% more than Epidemic and PRoPHET respectively, but the overhead ratio of SATC is 75% and 65% less than Epidemic and PRoPHET respectively. Besides, SATC can get the fewest average hop count, which indicates that the routing strategy of SATC is more efficient and more accurate, thus greatly decreasing the cost of message transmission.

B. Vary Message's time-to-live (TTL)

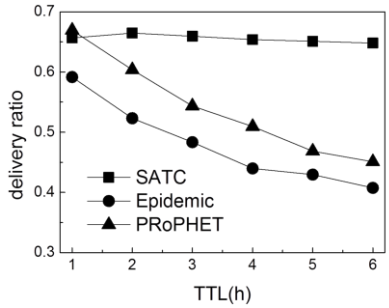
Fig. 11 shows the different simulation results of varying message TTL. SATC can still outperform Epidemic and PRoPHET in terms of message delivery ratio, overhead ratio and average hop count. And Epidemic is still unacceptable in this case.

When message's TTL increases constantly, there will be more messages in the entire network, thus greatly affecting routing performance due to the limited buffer resource. As shown in Fig.11, the message delivery ratios of Epidemic and PRoPHET are both decreasing, and the overhead ratios of Epidemic and PRoPHET keep increasing. The key reason is that they can't efficiently control message redundancy, creating a large number of redundant messages. So they can't cope with the increase of message's TTL. On the contrary, by using the methods of statistical analysis to select fewer but better neighbors as next relay nodes, SATC can greatly control message redundancy. As a result, the message delivery ratio of SATC can be stabilized at a high level, but the overhead ratio is still kept in a low level. At the same time, SATC also gets the fewest average hop count, which indicates that SATC can still keep the accuracy of routing strategy in this case and can greatly reduce the cost of message transmission.

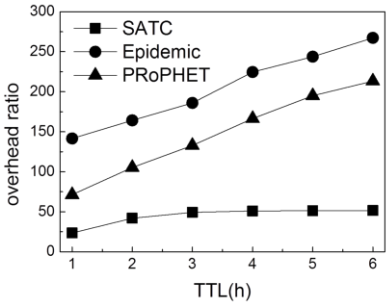
Finally from the whole figure, we can see that SATC can efficiently cope with the constant increase of message TTL, thus providing better routing performance compared to Epidemic and PRoPHET. In addition, the overhead ratio of SATC is 70% and 60% less than Epidemic and PRoPHET respectively.

C. Vary Message Size

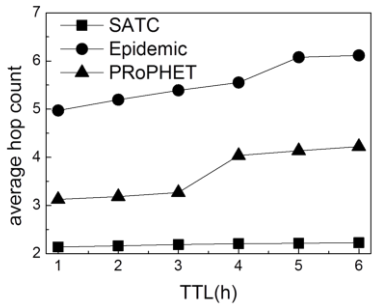
Fig. 12 compares the different simulation results of varying message size. SATC can still get some advantages in terms of message delivery ratio, overhead ratio and average hop count. And in this case, Epidemic is still unacceptable due to the lowest message delivery ratio and the highest overhead ratio.



(a) delivery ratio



(b) overhead ratio

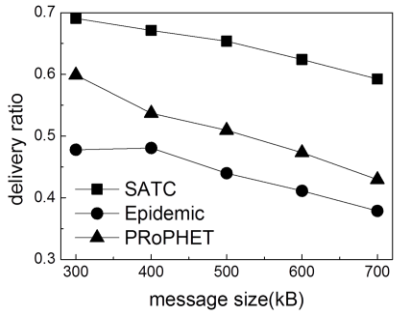


(c) average hop count

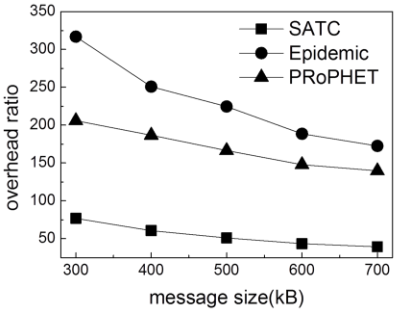
Figure 11. The delivery ratio, overhead ratio, average hop count VS message TTL

In Fig. 12 (a), the message delivery ratios of the three routing schemes keep decreasing when message size constantly increases. But SATC's message delivery ratio is always bigger than Epidemic and PRoPHET, which indicates that the buffer utilization efficiency of SATC is more efficient and SATC is able to cope with the increase of message size. In Fig. 12 (b), the overhead ratio of SATC is 75% and 60% less than Epidemic and PRoPHET respectively. Besides, the average hop count of SATC is also the fewest, which can prove once again that the routing strategy of SATC is more efficient, thus greatly decreasing the cost of message transmission.

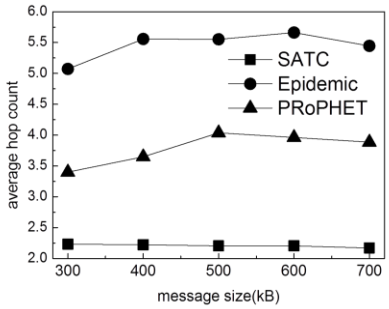
Finally from the Fig. 12, we can draw a conclusion that SATC can efficiently cope with the increase of message size, providing better routing performance compared to Epidemic and PRoPHET.



(a) delivery ratio



(b) overhead ratio



(c) average hop count

Figure 12. The delivery ratio, overhead ratio, average hop count VS message size

V. Conclusion And Future Works

In DTNs, due to node mobility, sparse node density, etc, there may never be a complete path between the sender and the receiver. But the case that some moving nodes form a connected temporary cluster may often occur. Relying on these clusters, a relay node may be able to quickly and successfully deliver message to the final destination node. Furthermore, by collecting a large number of encounter history information as the samples, we can use the methods of statistical analysis to objectively and accurately estimate whether a neighbor should be selected as next hop relay node, thus selecting fewer but better relay nodes to spread messages.

Extensive simulations have been conducted and the simulation results shows that SATC can outperform Epidemic and PRoPHET in terms of message delivery ratio, overhead ratio and average hop count. SATC can replace Epidemic and PRoPHET, thus providing better routing performance.

Our future work will focus on the DTN routing schemes in social networks. Social network is closely related with public life, and the DTN network model is increasingly used in civilian areas. In addition, we will look for opportunities to deploy our routing scheme to specific application scenarios.

ACKNOWLEDGEMENT

This research is supported in part by Foundation research project of Qingdao Science and Technology Plan under Grant No.12-1-4-2-(14)-jch and Natural Science Foundation of Shandong Province under Grant No.ZR2013FQ022.

REFERENCES

[1] Fall K, "A delay-tolerant network architecture for challenged internets," *proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Karlsruhe. Germany: ACM, 2003:27-34.

[2] Fan XM, Shen ZG, Zhang BX, Chen H, "State-of-the-Art of the architecture and techniques for delay-tolerant network," *Acta Electronica Sinica*, 2008,36(1):161-170.

[3] Fall K, Farrell S, "DTN: An architectural retrospective," *IEEE Journal on Selected Areas in Communications*, 2008,26(5):828-836.

[4] Burleigh S, Hooke A, Torgerson L, Fall K, Cerf V, Durst B, et al, "Delay-Tolerant networking: An approach to interplanetary Internet," *IEEE Communications Magazine*, 2003,41(6):128-136.

[5] M. Chuah, L. Cheng, B. Davison, "Enhanced Disruption and Fault Tolerant Network Architecture for Bundle Delivery," *in proceedings of IEEE Globlecom*, 2005.

[6] Cerf V, Burleigh S, Hooke A, et al, "Delay-Tolerant Network Architecture," *IETF RFC4838. Informational*, 2007.

[7] Juang P, Oki H, Wang Y, Martonosi M, Peh LS, Rubenstein AD, "Energy-Efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. *ACM SIGOPS Operating Systems Review*, 2002,36(5):96-107.

[8] Erramilli V, Chaintreau A, Crovella M, Christophe D, "Diversity of forwarding paths in pocket switched networks," *in proc of the 7th ACM SIGCOMM Conf. on Internet Measurement (IMC)*, San Diego: ACM Press, 2007. 161-174.

[9] Alberto Cerpa, Jeremy Elson, Deborah Estrin, Lewis Girod, Michael Hamilton, Jerry Zhao, "Habitat Monitoring: Application Driver for Wireless Communications Technology," *in the proceeding of ACM SIGCOMM Workshop on Data Communications*, Apr 2001.

[10] Z. Guo, B. Wang, J.-H. Cui, "Generic prediction assisted single-copy routing in underwater delay tolerant sensor networks," *Ad Hoc Networks*, pp. 1–14, Jan. 2013.

[11] H. Wu, R. Fujimoto, R. Guensler, M. Hunter, "Mddv: Mobilitycentric data dissemination algorithm for vehicular networks," *in Proc. ACM SIGCOMM Workshop on Vehicular Ad Hoc Networks (VANET)*,2004.

[12] P. R. Pereira, A. Casaca, J. J. P. C. Rodrigues, V. N. G. J. Soares, J. Triay, C Cervello-Pastor, "From Delay-Tolerant Networks to Vehicular Delay-Tolerant Networks," *IEEE Commun. Surv. Tutorials*, vol. 14, no. 4, pp. 1166–1182.

[13] V. N. G. J. Soares, J. J. P. C. Rodrigues, F. Farahmand, "GeoSpray: A geographic routing protocol for vehicular delay-tolerant networks," *INFORMATION FUSION*, pp. 1–12, Nov. 2011.

[14] Z. Zhang, "Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges," *IEEE Commun. Surv. Tutorials*, vol. 8, no. 1, pp. 24–37, 2006.

[15] Sharma G, Mazumdar R, Shroff B, "Delay and capacity tradeoffs in mobile ad hoc networks: A global perspective," *IEEE/ACM Trans on Networking*, 2007,15(5):981-992.

[16] Zhao W, Chen Y, Ammar M, Corner M, Levine B, Zegura E, "Capacity enhancement using Throwboxes in DTNs," *In: Proc. of the IEEE Int'l Conf. on Mobile Ad Hoc and Sensor Systems (MASS 2006)*, Vancouver: IEEE Communications Society, 2006. 31-40.

[17] Banerjee N, "An energy-efficient architecture for DTN Throwboxes," *In: Proc. of the Infocom 2007*, Anchorage: IEEE Communications Society, 2007. 776-784.

[18] W. Zhao, M. Ammar, E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," *in MOBIHOC'04*, pp 187-198, 2004.

[19] B. Tariq, M. Ammar, E. Zegura, "Message ferry route design for sparse ad hoc networks with mobile nodes," *in MOBIHOC'06* , pp 37-48, 2006.

[20] W. Zhao, M. Ammar, E. Zegura, "Controlling the mobility of multiple data transports ferries in a delay-tolerant network," *in INFOCOM'05*, pp 1407-1418, 2005

[21] A. Vahdat and D. Becker, "Epidemic Routing for Partially Connected ad hoc Networks. Duke University; Durham, NC, USA: Apr," 2000.

[22] T. Spyropoulos, K. Psounis, C. S. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pp. 252–259, 2005.

[23] T. Spyropoulos, K. Psounis, C. S. Raghavendra, "Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility," *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops' 07. Fifth Annual IEEE International Conference on*, pp. 79–85, 2007.

[24] Jain S, Fall K, Patra R, "Routing in a delay tolerant network," *In: Yavatkar R, Zegura EW, Rexford J, eds. Proc. of the 2004 ACM SIGCOMM*, Portland: ACM Press, 2004. 145-158.

[25] Zhou XB, Lu HC, Li JS, Hong PL, "AED: Advanced earliest-delivery algorithm used in DTN," *Journal of Electronics & Information Technology*, 2007,29(8):1956-1960.

[26] Jones EPC, Li L, Schmidtke JK, Ward PAS, "Practical routing in delay-tolerant networks," *IEEE Trans. on Mobile Computing*, 2007,6(8):943-959.

[27] Lindgren A, Doria A, Schelén O, "Probabilistic routing in intermittently connected networks," *SIGMOBILE Mobile Computing Communications Review*, 2003, 7(3):19-20

[28] Y. Wang, S. Jain, M. Martonosi, K. Fall, "Erasur-coding based routing for opportunistic networks," *in SIGCOMM' 05*, pp 229-236, 2005.

[29] Liao Y, Tan K, Zhang ZS, Gao LX, "Estimation based erasure-coding routing in delay tolerant networks," *In: Proc. of the 2006 Int'l Wireless Communications and Mobile Computing Conf. Vancouver: IEEE*

Communication Society, 2006. 557-562.
[30] Widmer J, Le Boudec JY, "Network coding for efficient communication in extreme networks," *In: Guerin R, Govindan R, Minshall G, eds. Proc. of the 2005 ACM SIGCOMM*, Philadelphia: ACM Press, 2005. 284-291.
[31] Yang SY, Jiang JT, Chen PZ, "OOPProPHET: A New Routing Method to Integrate the Delivery Predictability of ProPHET-Routing with OOP-Routing in Delay Tolerant Networks," Journal of Computers, vol.8, no.7, 2013.
[32] Zhenguo Yang, Liusheng Huang, Mingjun Xiao, Wang Liu, "Flow-Based Transmission Scheduling in Constrained Delay Tolerant Networks," Journal of Computers, vol.7, no.1, 2012.
[33] Fangbin Liu, Fengyu Liu, Hong Zhang, "An Energy-efficient and Real-time Anonymous Routing Protocol for Ad hoc Networks," Journal of Computers, vol.7, no.4, 2012.
[34] Ari Keranen, Jorg Ott, Teemu Karkkainen, "The ONE simulator for DTN protocol evaluation, "*Proceeding of the Second International Conference on Simulation Tools and Techniques*, Rome, Italy: ACM, 2009: 1-10.

Jixing Xu was born in Yantai, China, 1989. Xu received the BS degree in software engineering from Qingdao University in 2013 and is currently working toward the MS degree in Information Engineering College of Qingdao University. Xu's research interests include delay tolerant networks, with an emphasis on routing protocols, algorithms and social networks.

JianBo Li was born in Weifang city, Shandong Province, China on Apr 6, 1980. Li received the B.Eng and M.A.Sc in information engineering college from Qingdao University, China, in 2002 and 2005 respectively. Li's major study field concentrated on wireless sensor networks.

Between July 2005 and Aug 2006, he worked as an assistant teacher in Information engineering college of Qingdao university. From September 2006 , he joined the School of computer science & technology of University of Science and Technology of China as a PHD candidate and received the PhD degree in June 2009. Since July 2009, he has been worked at the Information Engineering College, Qingdao University, China, as an Associate Professor and Assistant Dean. He has a paper published on Journal of Networks as follows: J. Li, S.Jiang, "A scalable Clustering Algorithm in Dense Mobile Sensor Networks," Journal of Networks, vol.6(3), Academy Publisher , pp 505-512, Mar 2011. His research interests include routing protocol, data gathering, and topology control in wireless sensor networks, ad hoc networks and delay tolerant networks.

Dr Li now is a senior member of CCF (China Computer Federation).

Lei You was born in Yantai, China, 1989. You received the BS degree in software engineering from Qingdao University in 2012, and is currently working toward the MS degree in information Engineering College of Qingdao University. You's research interests include delay tolerant networks, with an emphasis on routing protocols, algorithms and social networks.

Chenqu Dai was born in Qingdao, China, 1987. Dai received the BS degree in computer science and technology from Qingdao University in 2011, and is currently working toward the MS degree in information Engineering College of Qingdao University. Dai's research interests include delay tolerant networks, with an emphasis on routing protocols, algorithms and social networks.