

# A Delay and Cost Balancing Protocol for Message Routing in Mobile Delay Tolerant Networks

Jingwei Miao<sup>a</sup>, Omar Hasan<sup>a</sup>, Sonia Ben Mokhtar<sup>a</sup>, Lionel Brunie<sup>a</sup>,  
Gabriele Gianini<sup>b</sup>

<sup>a</sup>*University of Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France*

<sup>b</sup>*Dipartimento di Informatica, Università degli Studi di Milano, via Bramante 65, Crema – 26013, Italy*

---

## Abstract

The increasing pervasiveness of mobile devices with networking capabilities has led to the emergence of Mobile Delay Tolerant Networks (MDTNs). The characteristics of MDTNs, which include frequent and long-term partitions, make message routing a major challenge in these networks. Most of the existing routing protocols either allocate an unlimited number of message copies or use a fixed number of message copies to route a message towards its destination. While the first approach unnecessarily floods the network, the rigidity of the second approach makes it inefficient from the viewpoint of message replication. Hence, the question that we address in this paper is: “how to dynamically allocate message copies in order to strike a balance between the delay and cost of message delivery?”. We present a novel adaptive multi-step routing protocol for MDTNs. In each routing step, our protocol reasons on the remaining time-to-live of the message in order to allocate the minimum number of copies necessary to achieve a given delivery probability. Experiment results demonstrate that our protocol has a higher delivery ratio and a lower delivery cost compared to the state-of-the-art Spray-and-Wait and Bubble protocols.

*Keywords:* mobile, delay tolerant, routing, adaptive, community

---

## 1. Introduction

Mobile Delay Tolerant Networks (MDTNs) are composed of a set of mobile devices, such as cell phones or sensor units, which can communicate with each other via short range wireless protocols (e.g. Bluetooth). A number of networking scenarios have been categorized as MDTNs, such as Vehicular Ad-hoc

---

*Email addresses:* [jingwei.miao@insa-lyon.fr](mailto:jingwei.miao@insa-lyon.fr) (Jingwei Miao),  
[omar.hasan@insa-lyon.fr](mailto:omar.hasan@insa-lyon.fr) (Omar Hasan), [sonia.ben-mokhtar@liris.cnrs.fr](mailto:sonia.ben-mokhtar@liris.cnrs.fr) (Sonia Ben Mokhtar), [lionel.brunie@insa-lyon.fr](mailto:lionel.brunie@insa-lyon.fr) (Lionel Brunie), [gabriele.gianini@unimi.it](mailto:gabriele.gianini@unimi.it) (Gabriele Gianini)

NETworks (VANETs) [1] and Pocket Switched Networks (PSNs) [2]. The characteristics of MDTNs include frequent and long-term network partitions which makes message routing one of the major challenges in these networks. In order to deal with the intermittent connectivity between nodes (i.e., mobile devices), message routing in MDTNs is often performed in a “store-carry-and-forward” manner [3], in which a node may store and carry a message for some time before forwarding it to another node [4].

In the literature, a number of routing protocols have been proposed for MDTNs. Nevertheless, most of them are inflexible from the viewpoint of message replication. For instance, flooding-based routing protocols [5], which refers to those protocols that always rely on an unlimited number of message copies to route a message, generate a very large number of redundant message copies. This approach can put undue stress on the limited resources of mobile devices. On the other hand, quota-based routing protocols [6, 7] allocate a fixed number of message copies for routing a message. The rigidity of this latter approach makes it inefficient, as a fixed number of message copies cannot suit all routing situations.

To the best of our knowledge, the adaptability of the number of message copies has been addressed only by a few authors. Bulut et al. in [8]. Routing in [8] is divided into multiple periods. In each period, the source node of a message sprays a certain number of message copies into the network. The source node dynamically chooses the allocation of message copies at the beginning of each period, until the message is delivered to the destination node or the Time-To-Live (TTL) expires. The protocol presented by Bulut et al. assumes that all nodes move under the same mobility pattern in a given network space. However, the reality is different. Recent studies [9, 10, 11] on the spatial characteristics of human mobility from real world traces demonstrate that humans in real-life tend to roam in smaller sub-regions rather than the whole network space. In addition, the allocation of message copies in [8] relies on the assumption that the source node is aware of the successful delivery of a message at any given instant. However, such an assumption rarely holds in reality, due to the frequent and long-term network partitions faced by MDTNs.

Moreover, Lo and Lu in [12] proposed a routing protocol, named dynamic congestion control based routing. In [12], message replication is controlled by the congestion level (i.e., light load, moderate load and heavy load). In the case of a light network load, the protocol increases the copies of a message by adding a fixed number of replicas. In the case of a moderate load, the protocol decreases the copies of a message. In the case of a heavy load, the protocol does not replicate a message and simply forwards the message to an encountering node.

Thompson et al., in [13], proposed a protocol to deal with congestion in DTNs. The protocol adjusts the replication rates at individual nodes according to the number of message copies in the network and the buffer capability of individual nodes.

In this paper, we present a novel routing protocol in MDTNs, called the Community-based Adaptive Spray (CAS) routing protocol. The goal of this

protocol is to allocate the minimum number of message copies for a message while still achieving a high delivery ratio. Our protocol exploits the community structure of human networks. A community is defined as a set of nodes which frequently co-exist and encounter (see Section 3.1). It has been demonstrated in the literature that nodes in a MDTN often form such structures [14].

Our protocol is composed of two major parts. First, a sub-protocol responsible for gathering mobility information about nodes upon encountering each other. This sub-protocol aims at learning/synchronizing the topologies of communities in the network. Second, a sub-protocol responsible for the routing process. Routing is organized around the notion of gateways between communities. Specifically, a gateway towards a community  $C$  is the node in a given community that has the highest probability to encounter any node in  $C$ . To route a message towards a given destination node, the source of a message uses the community topology to pre-compute the multi-hop path that traverses the minimal number of communities through their gateway nodes and that has the highest delivery probability. Furthermore, once the routing process is engaged, our routing protocol allocates a given number of message copies at each hop depending on the remaining TTL of the message. The urgency of delivering a message rises as the TTL of the message approaches expiration. The CAS protocol raises the number of message copies in the network in proportion to the remaining TTL in order to increase the probability of message delivery before time runs out. This strategy keeps the number of message copies in the network low while achieving a high delivery ratio.

The contributions of this paper are twofold:

- We propose a novel routing protocol that dynamically allocates message copies according to the remaining TTL of each message.
- The analysis of our protocol shows that it is the generalization of many existing protocols including Direct [15], Epidemic [5], Spray-and-wait [7], and some community based routing protocols [16]. By generalization, we mean that our protocol can dynamically decide to behave like one of these algorithms in order to better suit the current situation.

This paper is a considerably extended version of our previous work [17]. The major additions to the current paper include the following. First, we demonstrate that our routing protocol can represent a class of routing protocols. Second, we present new experiments to evaluate the performance of our protocol. Additionally, we have also developed an analytical model to compute the cover time of the topology of communities. The details of the analytical model are presented in an extended technical report [18].

The remainder of the paper is organized as follows. Section 2 discusses related work on routing in MDTNs. In Section 3, we introduce the system model and the information maintained by each node. In Section 4, we describe our proposal in detail. The simulations and results are presented in Section 5. Finally, we conclude the paper in Section 6.

## 2. Related work

In the literature, a variety of routing protocols have been proposed for MDTNs. The protocols can be classified into two broad categories based on the number of message copies utilized in the routing process [15, 19, 20]: single-copy and multi-copy.

In single-copy routing protocols, such as Direct Delivery [15] and First Contact [21], a single copy for each message exists in the network at any instant. Therefore, these routing protocols achieve the minimum transmission overhead in terms of the number of message copies employed during the routing process. However, due to the frequent and long-term network partitions that characterize MDTNs, these protocols often suffer from low delivery ratio and long delivery latency [22].

In order to guarantee a higher delivery ratio and a lower delivery latency, multi-copy protocols distribute multiple copies of each message in the network. Based on whether the number of message copies of a message is limited or not, multi-copy routing protocols can be further divided into flooding-based [5] and quota-based protocols [6, 7].

Epidemic [5] is a well known flooding-based protocol. It achieves the optimal routing performance in terms of delivery ratio and delivery latency at the cost of huge resource consumption, which makes it unpractical for resource constrained devices. In addition, the massively redundant messages can cause network congestion, which in turn can severely degrade the routing performance [23, 24].

Quota-based routing protocols [6, 7] achieve more reasonable delivery cost by restricting the number of message copies that they allocate for each message. Specifically, at any point in time, a fixed number of copies of each message exists in the whole network. However, the one-size-fits-all approach for allocating message copies is insufficient. It is obvious that a message with a shorter remaining TTL would need more copies than a message with a longer TTL to ensure successful delivery. The existing protocols do not take this factor into account. Additionally, these protocols assume that all nodes have the same capability of visiting every region covered by the network. However, reality is different. Recent studies [9, 10, 11] on the spatial characteristics of human mobility from real world traces demonstrate that humans usually roam in some relatively small sub-regions rather than the whole network space.

Aside from the aforementioned spatial characteristics of human mobility, human movement is also shown to be driven by social relationships [9, 25]. Hence, the nodes which frequently coexist in a common location are considered to consist of a community. The utilization of community structure has been proved to improve routing performance [14]. Therefore, the community structure has been widely utilized in the design of routing protocols. Such routing protocols are known as community-based routing protocols, which can belong to the class of either single-copy or multi-copy protocols.

Hui and Crowcroft in [26] proposed a community-based routing protocol for MDTNs, called LABEL. In LABEL, it is assumed that each node possesses a label indicating its community. The label of nodes (i.e., the community structure)

is then utilized to guide the forwarding of messages. Specifically, each message is forwarded to the nodes in the same community as the destination node.

Hui et al. in [14] presented a community-based routing protocol, named Bubble. Bubble combines centrality, which indicates the relative importance of a node in the network, and community structure to identify relay nodes. In Bubble, a message is forwarded from a node to the nodes that have a larger global centrality value (i.e, the total number of encountered nodes) than the current node until it reaches a node in the same community as the destination node. The message is then forwarded from a node to the nodes that have a larger local centrality value (i.e. the number of encountered community member nodes). Similar to [14], Li et al. in [27] proposed a community-based routing protocol, called IFR. The difference between IFR and Bubble is that IFR floods a message inside the community of the destination node.

Nevertheless, the centrality of nodes cannot accurately steer a message to the destination node [28]. To address this issue, Dang and Wu in [16] presented a routing protocol, named Clustering routing. Clustering routing utilizes gateway nodes to deliver the messages whose source node and destination node belong to different communities. Gateway nodes are the nodes which have the highest probability of encountering the nodes in other communities. However, messages in [16] are routed under Direct Delivery protocol, which is generally considered to be inefficient when the TTL of a message is limited [19].

Our work differs from the aforementioned works in two ways. First, the CAS routing protocol can dynamically allocate the minimum number of copies for each message while achieving a predefined delivery ratio, according to the TTL of a message. Second, our routing protocol is more generic than most routing protocols, which includes Direct, Epidemic, Spray-and-wait, and Clustering routing [16], as it can behave as one of these algorithms in specific routing situations.

### 3. System model

In this section, we first present the network model. We then introduce the information maintained by each node. Finally, we explain how the information maintained by nodes is updated.

#### 3.1. A delay tolerant network model

**Node.** We define a node as a mobile device. We assume that each node is equipped with a radio interface (e.g., Bluetooth) for short-range communication, and that the transmission range of all nodes is the same. Two nodes are considered to meet, if they are in the transmission range of each other. We assume that a node can only communicate with one other node at the same time. Two meeting nodes can exchange messages with each other. We also assume that each node has a unique identifier.

**Community.** We define a community  $C$  as a set of nodes which frequently co-exist and meet. Recent studies [9, 25] have indeed shown that human mobility

is actually driven by social relationships. Thus, the meeting frequency of nodes in the same community is considered to be much higher than that of nodes in different communities. We also assume that each community has a unique identifier. The set of nodes in a community  $C$  will be indicated hereafter by  $C = \{u_1, u_2, \dots, u_n\}$ , where  $n = |C|$ .

In practice, a node can belong to more than one community. For instance, a student can belong to the community of his fellow students or he can belong to the community of his family members. However, for simplicity, we assume in this work that a node can only belong to one community at a given time instance. The assumption prevents a message from being delivered to multiple communities that the destination node might belong to, which would increase the message delivery cost in terms of message copies. However, we note that this simplification does not have an impact on the accuracy of our routing protocol, i.e., delivery ratio and delivery cost. The same assumption is also made in state-of-the-art literature, such as Hui et al. [14], and Dang and Wu [16]. In future work, we would like to consider the scenario where a single node can belong to multiple communities.

**Inter-contact time between two nodes.** The inter-contact time (also known as inter-meeting time) between two nodes is the time interval between two successive encounters between them.

**Mobility Model.** Mobility models are generally characterized by the inter-contact time between two nodes [8, 19]. Karagiannis et al. in [10] demonstrated that under a large class of mobility scenarios in real life, the inter-contact time follows a power-law in a finite range, and then exhibits an exponential decay. It is consistent with the suggestion made by Gonzalez et al. in [29] that a power law with an exponential decay is a very good approximation of human mobility patterns. Additionally, Chaintreau et al. in [30] pointed out that the exponential decay eliminates the issue of infinite message forwarding delay. Building on these previous studies, the inter-contact time of nodes is assumed to be exponentially distributed or have an exponential tail characterized by a contact rate (the inverse of the expected inter-contact time of any pair of nodes). This is a widely accepted assumption in MDTNs [8, 19, 31, 32].

**Inter-contact time between a node and a community.** The inter-contact time between a node  $u$  and a community  $C$  is the time interval between any two successive contacts between node  $u$  and any node member of the community  $C$ .

**Gateway.** A node  $w$  in a community  $C_i$  is defined as the gateway connecting to another community  $C_j$  if node  $w$ 's average inter-contact time with the nodes of the community  $C_j$  is the shortest among the other nodes of its community  $C_i$ .

**Network.** Let the set of all nodes in the environment be given as the set  $V$ . Let the set of all communities in the environment be  $\mathbb{M} = \{C_i | 1 \leq i \leq k, V = \bigcup_{i=1}^k C_i\}$ . Edges  $e_{i,j}$  and  $e_{j,i}$  exist between two communities  $C_i$  and  $C_j$  (where  $1 \leq i \leq k, 1 \leq j \leq k, i \neq j$ ), if some nodes from the two communities have encountered at least once. The weight of the edge  $e_{i,j}$ , denoted as  $\varpi_{i,j}^w$ ,

is the average inter-contact time of the gateway  $w$  of community  $C_i$  with the nodes of the community  $C_j$ . Notice that the gateway from community  $C_i$  to the community  $C_j$  is not the same node as the gateway from  $C_j$  to  $C_i$ , so the community graph is a directed graph. Let  $E = \{e_{i,j} | 1 \leq i \leq k, 1 \leq j \leq k, i \neq j\}$ . We will represent the MDTN formed by all the nodes of the network by their community graph  $G(\mathbb{M}, E)$ .

**Message.** A message is represented as a tuple  $\langle S, D, I, C, L, T \rangle$ , where  $S$  is the source node,  $D$  is the destination node,  $I$  is the intermediate target node, which is a gateway node connecting the current community to another community (or the destination node if the message is yet in the community of the destination node).  $C$  is the identifier of the community connected by the gateway node (or null if the intermediate target node is the destination node).  $L$  is the number of copies allocated to route the message to the intermediate target node in the current community,  $T$  is the message TTL.

### 3.2. Information maintained by a node

Each node in the network, e.g., node  $u$ , maintains six types of information: its node ID  $u$ , its community ID  $C^u$  (the exponent here indicates the fact that  $u$  belongs to this community), a community table, a gateway table, a contact table, and a community graph.

**Community table.** The community table of node  $u$  holds the node ID and the community ID of all nodes who have encountered the nodes of  $u$ 's community. Node  $u$  also maintains a timestamp which indicates the time when the table was last updated.

**Gateway table.** The gateway table of node  $u$  contains the following fields for each known community  $C_i$  and for each gateway  $w$  linking  $C_i$  to another community  $C_j$ : the community ID of  $C_i$ , the node ID of the gateway  $w$ , the community ID of  $C_j$ , and the average inter-contact time between the gateway  $w$  and the community  $C_j$  given as  $\varpi_{i,j}^w$ . Node  $u$  also maintains a timestamp which records the last updated time of this table.

**Contact table.** The contact table of node  $u$  maintains the following fields for each encountered node  $v$ : the node ID of  $v$ , the sum of inter-contact times between the nodes  $u$  and  $v$  denoted as  $\tau_{u,v}$ , the number of encounters  $\sigma_{u,v}$  (the latter two quantities are used to compute the average inter-contact time), the end time of the last encounter  $t_{u,v}^e$ , and the start time of the ongoing encounter  $t_{u,v}^b$  (if there is one).

**Community graph.** In the community graph of node  $u$ , each vertex denotes a known community e.g., community  $C_i$ . Each vertex is labeled by a tuple  $\langle |C_i|, \varpi_{C_i} \rangle$ , where  $|C_i|$  is the number of nodes in community  $C_i$ ,  $\varpi_{C_i}$  is the average inter-contact time between the member nodes of the community  $C_i$ . The weight of the directed edge  $e_{ij}$  denoted by  $\varpi_{i,j}$  is the average inter-contact time between the gateway in community  $C_i$  and the community  $C_j$ . The community graph of node  $u$  locally reflects the topology of the communities of the network as it is known by  $u$ , which is illustrated in Figure 1.

The accuracy of the community graph maintained locally by nodes depends on the dynamic nature of the structure of communities. If the communities

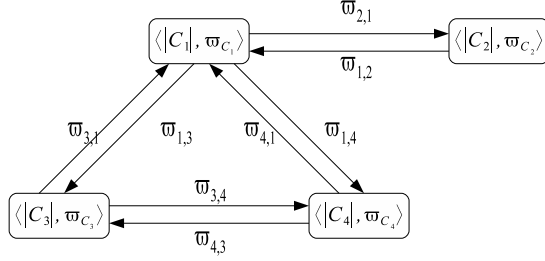


Figure 1: Community Graph

are stable, the community graph of a node remains mostly accurate and up to date. However, if the communities are highly dynamic, the community graph maintained by a node may need to be update frequently. Frequent updates may become a challenge in a large scale dynamic community graph because of the resources required for propagation of the changes. However, as pointed out by Hui et al. in [14], the community structure of human nodes is fairly stable.

### 3.3. Maintenance of information

The maintenance of the above information is driven by events. There are three kinds of events in the protocol: 1) *Connect Event*, 2) *Disconnect Event*, and 3) *Update Event*. We assume that nodes in the network honestly synchronize the maintained information.

**Connect Event.** It happens at the moment when two nodes enter the transmission range of each other. When a connect event takes place, two meeting nodes honestly exchange and update their corresponding values in each field of their community table, community graph, and gateway table, according to the value of their timestamps. They also set the timestamps.

**Disconnect Event.** It happens at the moment when two nodes  $u$  and  $v$  go out of the transmission range of each other. Let  $t_{u,v}^e$  be the end time that node  $u$  encounters node  $v$ . Let  $t_{u,v}^b$  be the begin time that node  $u$  encounters node  $v$  of the ongoing encounter. Thus, the last inter-contact time between node  $u$  and  $v$  is set to  $\Delta\tau_{u,v} = t_{u,v}^b - t_{u,v}^e$ . The sum of inter-contact time between nodes  $u$  and  $v$  is  $\tau_{u,v} = \tau'_{u,v} + \Delta\tau_{u,v}$ , where  $\tau'_{u,v}$  is the sum of inter-contact time between node  $u$  and  $v$  before last encounter. Then, the number of encounters  $\sigma_{u,v}$  increases by 1.

Consequently, the average inter-contact time between two nodes  $u$  and  $v$  is expressed as follows:

$$\varpi_{u,v} = \begin{cases} +\infty, & \sigma_{u,v} = 1 \\ \frac{\tau_{u,v}}{\sigma_{u,v} - 1}, & \sigma_{u,v} > 1 \end{cases} \quad (1)$$

The average inter-contact time between two nodes is the major parameter used by methods of community detection. Specifically, a node joins (or leaves) a community if it is (or is not) qualified to be in the community. Since the



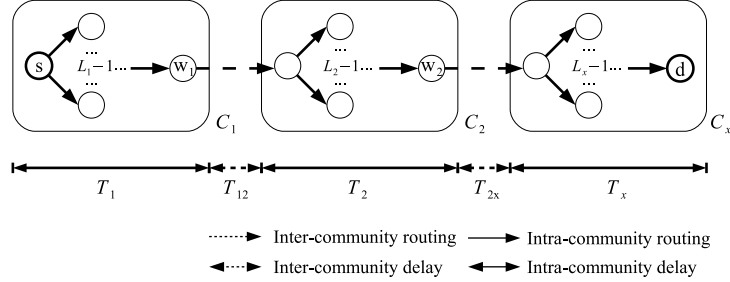


Figure 2: Routing Protocol Overview

community detection is out of this paper’s scope, the reader is suggested to refer to [33] for further information.

Moreover, as in [31, 34], we assume that the nodes in the same community have the same average inter-contact time. As a consequence  $\varpi_{C_i} = \varpi_{u,v}$ , for any nodes  $u$  and  $v$  are in the same community  $C_i$ .

**Update Event.** The update event is periodically invoked synchronously by all nodes every  $\gamma$  time units. Let  $C'_j$  be the set of nodes which belong to a community  $C_j$  and have met node  $u$  by more than once. Then the average inter-contact time between node  $u$  and community  $C_j$  is computed as follows.

$$\varpi_{i,j}^u = \frac{\sum_{v \in C'_j} \tau_{u,v}}{\sum_{v \in C'_j} (\sigma_{u,v} - 1)} \quad (2)$$

If node  $u$  achieves an average inter-contact time with community  $C_j$  that is shorter than the one of the current gateway  $w$  (that is  $\varpi_{i,j}^u \leq \varpi_{i,j}^w - \epsilon$ , where  $\epsilon$  is a threshold), then the relevant fields are updated i.e., the node ID of the gateway from  $C_i$  to  $C_j$ , the inter-contact time and timestamp are updated to  $u$ ,  $\varpi_{i,j}^u$  and the current time respectively.

## 4. Protocol design

In this section, we first briefly describe the mechanism of CAS. We then present the design of CAS in detail. We proceed to demonstrate that our protocol is the generalization of several existing routing protocols. Finally, we briefly discuss and provide a reference to our analysis of cover time estimation.

### 4.1. Overview of CAS

In this section, we give an overview of our routing protocol. A routing example is depicted in Figure 2. This figure shows a number of nodes belonging to three communities  $C_1$ ,  $C_2$ , and  $C_x$ . A source node  $s$  that belongs to the community  $C_1$  wants to send a message to a node  $d$  that belongs to the community  $C_x$ .

In CAS, the routing process is composed of multiple sub-processes. In each sub-process, a message is routed to an intermediate target node, which is the gateway node bridging two communities or the destination node if the latter belongs to the same community as the source node. At the beginning of each sub-process, the shortest path from the current community to the community of the destination node is computed according to the community graph. For instance, in Figure 2, when the source node  $s$  generates a message, it computes the shortest path from its community  $C_1$  to the destination node's community  $C_x$ , that is,  $C_1C_2C_x$ . After that, node  $s$  looks up its gateway table to find the gateway node  $w_1$ , which connects its community with the community  $C_2$ . The node  $w_1$  is then considered as the first intermediate target node to reach. At the same time, the minimum number of message copies needed to route the message to the intermediate target node is computed, based on the message TTL and a predefined corresponding delivery ratio. These message copies are sprayed inside the current community in a binary spraying manner in which a message carrier hands over half of the copies it holds to each node it meets. When the intermediate target node encounters a node in its connecting community (e.g.,  $C_2$  in the case of  $w_1$ ), it forwards the message to it. The above process is repeated until the message reaches the destination node or expires.

#### 4.2. Design of CAS

As explained above, the CAS routing process is divided into multiple sub-processes. Each sub-process consists of the following two phases: 1) optimization of the number of message copies to be distributed and 2) message routing.

##### 4.2.1. Optimization of the number of message copies

According to the analysis of the routing performance presented in [7], if  $L$  copies of a message are distributed in the network and the TTL of the message is  $T$ , the expected delivery probability of the message can then be calculated as  $p_d = 1 - e^{-\lambda LT}$ , where  $\lambda$  is the contact rate, that is, the inverse of the average inter-contact time between nodes [8]. Hence, if the expected delivery probability  $p_d$  is assigned, the relationship between the number of message copies and the TTL can be expressed as Equation 3. Note that the delivery probability is the delivery ratio from the viewpoint of all generated messages.

$$L \times T = -\frac{\ln(1 - p_d)}{\lambda} \quad (3)$$

Recall that an MDTN is modeled as a directed graph of communities  $G(\mathbb{M}, E)$  (Section 3.1). Let  $h$  a path in  $G(\mathbb{M}, E)$ . Let  $n$  be the number of communities in path  $h$ . The weight (denoted as  $wgt(h)$ ) of path  $h$  is the sum of the weights of the edges that form the path, which is expressed in Equation 4. It represents the sum of the average inter-contact time between the communities in path  $h$ . Note: we assume that in the community graph that represents a MDTN every community is reachable (i.e., a non-reachable community constitutes a distinct MDTN).

$$wgt(h) = \begin{cases} \sum_{i=1}^{n-1} \varpi_{i,i+1}^w, & \text{if } n > 1 \\ 0, & \text{if } n = 1 \end{cases} \quad (4)$$

Further, let us consider two communities  $C_1$  and  $C_2$ . Let  $h(C_1, C_2)$  be any path which originates within community  $C_1$  and ends within community  $C_2$ . In a network, it is possible that more than one such path exists. Let  $H(C_1, C_2)$  be the set of all possible paths  $h(C_1, C_2)$ . Let  $h^*(C_1, C_2)$  be the shortest path from community  $C_1$  to community  $C_2$ .

Assume a node  $u$  carries a message denoted by  $\langle S, D, I, C, L, T \rangle$ . Let  $h^*$  be the shortest path from the community of node  $u$  to the community of the destination node. Let  $n$  be the number of communities in the path  $h^*$ . Let  $L_i$  denote the number of message copies distributed in a community  $C_i$ . Let  $w$  be the gateway node connecting community  $C_i$  to community  $C_{i+1}$ . Let  $p_d$  be the expected delivery probability of the message. Let  $\lambda_i$  be the contact rate of nodes in community  $C_i$ , which is the inverse of the average inter-contact time  $\varpi_{C_i}$  of nodes in community  $C_i$ . Then the optimization of delivery cost can be expressed as in Equation 5-7.

$$\min \sum_{i=1}^n L_i \quad (5)$$

$$s.t. \sum_{i=1}^{n-1} \varpi_{i,i+1}^w + \sum_{i=1}^n \frac{-\ln(1-p_d)}{\lambda_i L_i} \leq T \quad (6)$$

$$L_i \leq |C_i|, \text{ where } 1 \leq i \leq n \quad (7)$$

The purpose of the objective function Equation 5 is to minimize the number of message copies utilized to route a message. Equation 6 expresses the fact that the sum of the time spent to route the message in communities and the sum of the average inter-contact time between communities should be less than the TTL of the message. The constraint Equation 7 expresses the fact that the number of message copies allowed in a community should be less than the number of nodes in the community.

All  $L_i$  are positive integers and finite; thus Equation 5-7 defines a classical integer optimization problem for which a number of heuristics can be applied. However, in practice, the number of communities in path  $h^*$ ,  $n$ , and the acceptable values for  $L_i$  are small, so an exhaustive search, i.e. the enumeration of all the possible values for  $L_i$ , is clearly tractable. For instance, there are about 8 communities in the MIT reality mining dataset [14], and most of communities contain less than 10 nodes. The pseudo-code of the solution is illustrated in Figure 3.

Without loss of generality, let's consider that the algorithm is invoked by a node  $u$ . Node  $u$  has a message  $m$ . The shortest path from the community of node  $u$  to the community of the destination node of message  $m$  is  $h^*$ . The algorithm initially gets the community (denoted by  $C^*$ ) next to the community

of nodes  $u$  on the path  $h^*$  (Line 1); and sets the number of the nodes in node  $u$ ' community as the default value for  $L_u$ , which indicates the number of message copies of message  $m$  for node  $u$  (Line 2). The algorithm then enumerates all the possible values for the number of copies in each community on the path  $h^*$  (Lines 19 to 31). More specifically, the algorithm uses the elements in a vector  $\pi$  to express the number of copies of message  $m$  allocated for each community on the path  $h^*$ . The algorithm initially sets all the elements in  $\pi$  to be 1. It then starts with the last element in  $\pi$  (Line 19), and selects all possible values for the last element (Lines 21 to 23). The algorithm then uses the values of the elements in  $\pi$  to verify whether (1) the allocated number of message copies can satisfy Equation 6; (2) the sum of these values is smaller than the previous minimal sum, which is initially set as the number of nodes on the path  $h^*$  (Line 15). If so, the algorithm updates the minimal sum (Line 16) and  $L_u$  (Line 17). The algorithm then resets the value of the last element to be 1 (Line 25), and goes to the element before the last element (Line 26). The above process is repeated until all possible values are enumerated. Finally, the algorithm returns  $C^*$  and  $L_u$ .

There are two heuristic methods to reduce the computation complexity of algorithm `OptimalCost`: (1) using a smaller sample to test the values for  $L_2, \dots, L_n$ . (2) terminate the computation of the algorithm when the first combination of values in  $\pi$  is found.

#### 4.2.2. Message Routing

In CAS, we distinguish the routing within a community from the routing among communities. In the former case, the allocated message copies are sprayed inside the current community in a binary manner, in order to minimize the time to spray message copies. Specifically, a node hands over half of the number of a message copies to an encountered node, which is in the same community and does not have the message. When the node has just one copy of the message, the message can only be forwarded to the intermediate target node. As we stated in Section 3, the meeting frequency of nodes in the same community is much higher than that of nodes in different communities. Thus, messages can be delivered to the intermediate target node with a higher probability.

As for the routing among communities, a gateway node forwards a message to an encountered node if it belongs to the next community along the shortest path from the gateway node's community to the community of the destination node. After it gets the message, the receiving node re-computes the shortest path from its community to the community of the destination node, and sets the intermediate target node as the gateway node bridging the next community, or the destination node if it is in the same community as the destination node. After that, it calls the Algorithm `OptimizeCost` to compute the needed number of message copies utilized to route the message to the intermediate target node. The complete mechanism of the routing protocol in CAS is summarized in Figure 4.

---

**Algorithm: OptimizeCost****Participants:** Node  $u$ .**Input:** (1)  $D_m$ , the destination node of message  $m$ . (2)  $T$ , the TTL of message  $m$ . (3)  $p_d$ , a predefined delivery probability of message  $m$ . (4)  $h^*$ , the shortest path from node  $u$ 's community to node  $D_m$ 's community.**Output:** (1)  $C_u^*$ , the community next to node  $u$ 's community on the path  $h^*$ . (2)  $L_u$ , the number of copies of message  $m$  carried by node  $u$ .**Setup:** (1)  $n$ , the number of communities in path  $h^*$ . (2)  $\varpi_{sum}$ , the weight of the shortest path  $h^*$ . (3)  $L_{min}$ , the minimum number of message copies utilized to route the message. It is initially set to the sum of the number of nodes in each communities in the path  $h^*$ . (4),  $\pi$ ,  $1 \times n$  vector, whose elements are 1 (5)  $\lambda_i$ , the contact rate of nodes in community  $C_i$ , where  $1 \leq i \leq n$ .

```
1:  $C_u^* \leftarrow h^*[2]$  {The index of the community  $C_u^*$  is 2.}
2:  $L_u \leftarrow |h^*[1]|$ 
3:  $j \leftarrow n$  {The index for the vector  $\pi$ .}
4: while  $j \geq 1$  do
5:    $T_{sum} \leftarrow 0$ 
6:    $L_{sum} \leftarrow 0$ 
   {Compute the needed number of message copies for a combination of values in
    $\pi$ .}
7:   for  $i \leftarrow 1$  to  $n$  do
8:     if  $\pi[i] \leq |h^*[i]|$  then
9:        $L'_i \leftarrow \pi[i]$ 
10:    else
11:       $L'_i \leftarrow |h^*[i]|$ 
12:    end if
13:     $T_{sum} \leftarrow T_{sum} + \frac{-\ln(1-p_d)}{\lambda_i L'_i}$ 
14:     $L_{sum} \leftarrow L_{sum} + L'_i$ 
15:  end for
   {Find the minimal number of message copies.}
16:  if  $T_{sum} + \varpi_{sum} \leq T$  and  $L_{sum} < L_{min}$  then
17:     $L_{min} \leftarrow L_{sum}$ 
18:     $L_u \leftarrow \pi[1]$ 
19:  end if
   {List all possible combination of values in  $\pi$ }
20:   $j \leftarrow n$ 
21:  while  $j \geq 1$  do
22:    if  $\pi[j] < |h^*[j]|$  then
23:       $\pi[j] \leftarrow \pi[j] + 1$ 
24:      break
25:    else
26:       $\pi[j] \leftarrow 1$ 
27:       $j \leftarrow j - 1$ 
28:    end if
29:  end while
30: end while
31: return  $(C_u^*, L_u)$ 
```

Figure 3: Algorithm:OptimizeCost

---

**Protocol: RouteMessage**

**Participants:** Two encountering nodes  $u$  and  $v$ .

**Input:** (1)  $m$ , a message carried by node  $u$ . (2)  $D_m$ , the destination node of message  $m$ . (3)  $I_m$ , the intermediate target node of message  $m$ . (4)  $L_u$ , the number of copies of message  $m$  for node  $u$ . (5)  $L_v$ . (6)  $T$ , the TTL of message  $m$ . (7)  $p_d$ , a predefined message delivery probability. (8)  $C_u^*$ , the community next to node  $u$ 's community on the shortest path from node  $u$ 's community to node  $D_m$ 's community. (9)  $C_v^*$ .

**Output:** Message  $m$  is forwarded to node  $v$ , if one of the following conditions is met: (1) node  $v$  is the destination; (2) node  $v$  is the intermediate target node; (3) nodes  $u$  and  $v$  are in the same community, and node  $u$  has more than one copy of message  $m$ ; (4) node  $u$  is the gateway node bridging community  $C_u^*$ , and node  $v$  is in community  $C_u^*$ , and node  $u$  has not yet forwarded message  $m$  to another node member of the community  $C_u^*$ . Furthermore, the number of copies of message  $m$  for node  $v$ ,  $L_v$ , is computed and the intermediate node on the message route is updated.

**Events and Associated Actions:****node  $u$  initiates the protocol**

```
1: if  $C^u = C^v$  then
2:   if  $v = I_m$  then
3:     node  $u$  forwards message  $m$  to node  $v$ .
4:   else if  $L_u > 1$  then
5:     node  $u$  forwards half number of copies of message  $m$  to node  $v$ , and keeps the
       remaining copies.
6:   end if
7: else
8:   if  $v = D_m$  then
9:     node  $u$  forwards message  $m$  to node  $v$ 
10:  else if  $u$  is the gateway node connecting  $C_u^*$  and  $C_u^* = C^v$  and node  $u$  has not
       yet forwarded message  $m$  to another node in the community  $C_u^*$  then
11:    node  $u$  forwards message  $m$  to node  $v$ 
12:  end if
13: end if
```

**upon node  $v$  receives a message  $m$  from node  $u$** 

```
1: if  $C^u \neq C^v$  and  $v \neq D_m$  then
2:    $\langle C_v^*, L_v \rangle \leftarrow \text{OptimizeCost}(D_m, T, p_d)$ 
3:   if  $C^{D_m} = C^v$  then
4:      $I_m \leftarrow D_m$ 
5:   else
6:     node  $v$  looks up its gateway table to find the gateway node  $w$  to community
        $C_v^*$ 
7:      $I_m \leftarrow w$ 
8:   end if
9: end if
```

Figure 4: Protocol:RouteMessage

---

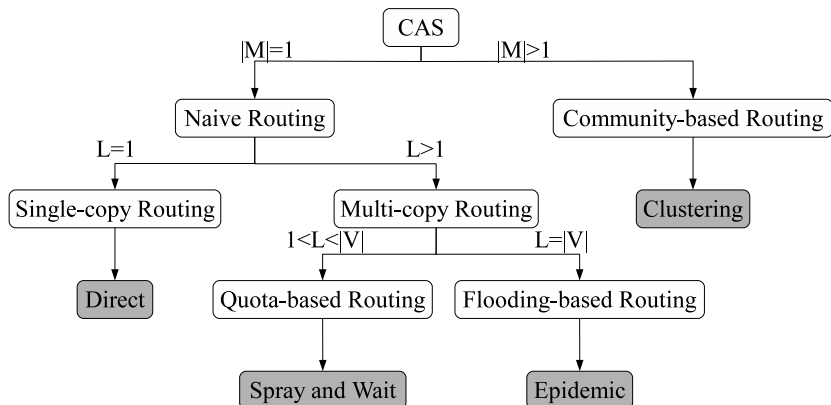


Figure 5: The transformation from CAS to several kinds of routing protocols, where  $L$  is the number of allocated copies for a message,  $V$  is the set of all nodes in a network,  $M$  is the set of all communities in a network.

#### 4.3. CAS generalizes classes of routing protocols

Another relevant point about CAS is that it represents the generalization of many existing routing protocols, including Direct [15], Spray-and-Wait [7], Epidemic [5], and Clustering routing [16]. Depending on the number of communities in the network and the number of message copies, CAS can dynamically transform to these routing protocols. Let  $V$  be the set of all nodes in a network. Let  $\mathbb{M}$  be the set of all communities in the network. Let  $L$  be the number of allocated copies for a message. The transformation from CAS to these routing protocols is then illustrated in Figure 5.

- Direct ( $L = 1, |\mathbb{M}| = 1$ ): The source node of a message can only forward the message to the destination node. Direct is a well known single-copy routing protocol.
- Spray-and-Wait ( $1 < L < |V|, |\mathbb{M}| = 1$ ): Each message is associated with some forwarding tokens, which indicates message copies. Each message has two phases: the spray phase and the wait phase. If there is more than one forwarding token left, the message is in the spray phase. During the spray phase, the forwarding tokens of a message can be sprayed to an encountered node without the message. If there is only one forwarding token left, it is in the wait phase. During the wait phase, a message can only be forwarded to the designation node. Spray-and-Wait is a representative routing protocol of quota-based routing.
- Epidemic ( $L = |V|, |\mathbb{M}| = 1$ ): Each node stores its messages in its buffer. When two nodes meet, each of them complements the missing messages according to the messages in the other's buffer. It is therefore flooding-based in nature.

- Clustering routing ( $|\mathbb{M}| > 1$ ): The encounter information of nodes is synchronized and processed to identify the community structure and the gateway nodes. A gateway node connecting its community to another community is the node that has the highest probability of meeting any node in the latter community. In this protocol, Direct routing is employed to route a message to a gateway node or the destination node if the message is in the destination node’s community. Clustering routing is a community-based routing protocol.

It is worth pointing out that Epidemic and Direct achieve the upper and lower bounds of routing performance in terms of delivery ratio and delivery cost [19, 15] respectively. CAS realizes a trade-off by minimizing the delivery cost while maintaining a predefined delivery ratio.

#### 4.4. Analysis of cover time estimation

Since our protocol depends on the local community graph to allocate message copies, it is necessary to estimate the cover time of the community graph when the role of a node is changed. For instance, when a node becomes the new gateway node, the information about this change is propagated to all the relevant nodes of the community. The cover time is defined as the difference from the time of the change at a node to the time when all the relevant nodes are aware of the change. The cover time of maintained information is analyzed from the following two aspects: 1) intra-community message exchange and 2) inter-community message exchange. In the extended version of this paper, published as a technical report [18], we develop an analytical model for estimating the cover time of the community graph. We also evaluate the accuracy of our analytical model by comparing the theoretical results with the simulation results by using a standard simulator. The reader may refer to the technical report [18] for the detailed analysis.

## 5. Performance evaluation

In this section, we present the performance evaluation of CAS. We first introduce our simulation settings and the mobility model in Sections 5.1 and 5.2 respectively. We then present the routing protocols against which we compare the performance of CAS in Section 5.3 followed by the description of the performance evaluation metrics in Section 5.4. Finally, we present the simulation results in Section 5.5.

### 5.1. Simulation settings

The simulations have been conducted by the ONE simulator [35, 36]. The simulation scenario considers a rectangular area of 2000 m  $\times$  1500 m. This area is equally divided into 12 regions each measuring 500 m  $\times$  500 m. Initially, twenty nodes are deployed in each region. Each node considers the region in which it has been deployed as its *local region*. According to the mobility model



Table 1: Parameter settings

Parameter Name	Value
Simulation area	2000 m $\times$ 1500 m
Transmission range	10 m
Simulation duration	13 hours + TTL
Warm-up period	1 hour
Message generation rate	1 message per minute
Number of communities	12
Number of nodes in a community	20
Node speed	1.34 m/s
$p_l$ (low mobility nodes)	1
$p_r$ (low mobility nodes)	0
$p_l$ (high mobility nodes)	0.5
$p_r$ (high mobility nodes)	0.5
$\gamma$	1 hour
$p_d$	0.8

(see Section 5.2), a node is more likely to visit its local region than other regions. This leads to the encounter frequency of nodes deployed in the same region being much higher than that of nodes deployed in different regions. Consequently, nodes associated to one region constitute a community.

Moreover, each node is equipped with a radio interface for short-range communication. The transmission range and rate are set as 10 m and 2 Mb/s respectively. This is consistent with contemporary protocols, such as Bluetooth [35]. Additionally, the speed of nodes is set to 1.34 m/s, which is an average human walking speed [35]. Since we find that the routing performance of CAS and the chosen routing protocols converges after about six hours, we set thirteen hours for each simulation in order to achieve statistical confidence. Since CAS depends on the knowledge of the network topology, we specify the first hours as a warm-up period to allow the nodes to setup the community graph. During the warm up period, no message is generated. After this period, every minute, a random node sends a message to a random destination node. The detailed settings are listed in Table 1.

### 5.2. Mobility model

In our evaluation, we adopt the community-based mobility model proposed in [37], which has been widely utilized for the evaluation of community-based routing protocols [38, 16]. In this mobility model, each community is associated with a geographical area. The movement of a node, which belongs to a community, consists of a sequence of *local* and *roaming* epochs. A local epoch is a random movement restricted inside the area associated with the community. A roaming epoch is a random movement inside the entire network. If the node's previous epoch was a local one, the next epoch is a local one with probability  $p_l$ , or a roaming epoch with probability  $1 - p_l$ . Similarly, if the node's previous

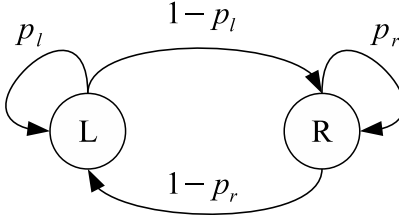


Figure 6: Community-based mobility model

epoch was a roaming one, the next epoch is a roaming one with probability  $p_r$ , or a local one with probability  $1 - p_r$ . The state transition between local and roaming epochs is shown in Figure 6.

In addition, we assign different mobility characteristics to the nodes in a region in order to produce a scenario closer to reality, where there is heterogeneity in mobility among nodes. To achieve this, the nodes in a region are divided into two kinds: *low mobility* nodes and *high mobility* nodes. The low mobility nodes can only roam inside their local regions, while the high mobility nodes may roam among their local region as well as the entire space. Specifically, 95% nodes in each region are chosen as the low mobility nodes with  $p_l=1$  and  $p_r=0$ , while other nodes are chosen as the high mobility nodes with  $p_l=0.5$  and  $p_r=0.5$ .

A number of mobility traces have been collected from human movements in real-life scenarios. These mobility traces include the Reality Mining trace [14], which was collected in a campus environment, and Infocom05 [14], which was collected in a conference environment. As future work, we would like to perform the evaluation of our routing protocol by using real mobility traces, such as the ones mentioned above as well as others which are available publicly from sources such as the crawdad database (crawdad.cs.dartmouth.edu). This would allow us to evaluate the performance of our protocol in real-life scenarios.

### 5.3. Routing protocols

Based on the above settings, we have compared the performance of CAS against the following protocols:

**Epidemic:** Each node forwards a copy of each unexpired message in its buffer to the encountered node that does not have a copy of the message.

**Direct:** The source node of a message can only forward the message to the destination node.

**Binary Spray-and-Wait (BSW):** Each message has  $L$  copies. A message carrier forwards half of the message copies to an encountered node, if its  $L > 1$  and the latter does not have the message. The message, which has only one message copy left, can only be forwarded to the destination.

**Bubble:** It utilizes social information about nodes, such as their centrality and the community to which they belong. In this protocol, a message is forwarded based on the global rankings of two encountering nodes, until it reaches a node in the community of the destination node. After that, the message is

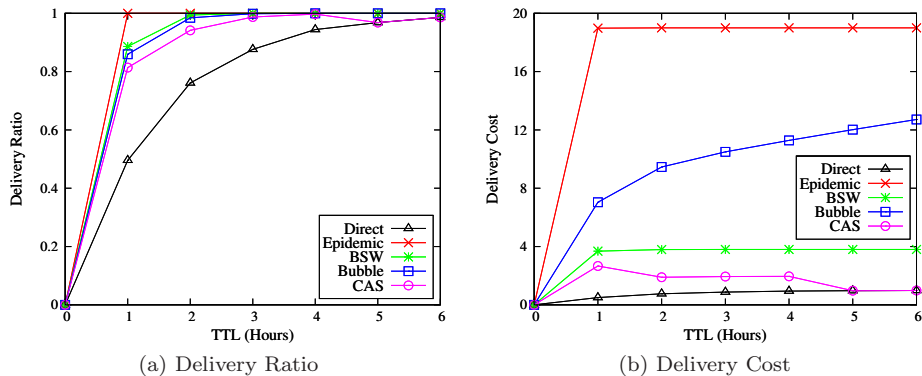


Figure 7: Comparison of the routing performance of several algorithms in the single-community case.

forwarded based on the local rankings of two encountering nodes, until it either reaches the destination node or expires. In the experiment, the length of time window is set to 1 hour.

#### 5.4. Metrics

The goal of CAS is to minimize the number of message copies and while maintaining a predefined delivery ratio. Therefore, we have measured the following metrics for the simulations that we have conducted in this work:

**Delivery ratio:** The proportion of messages that have been delivered out of the total unique messages created.

**Delivery cost:** The total number of message transmissions in the simulation. To normalize this, we divide it by the total number of unique messages created.

#### 5.5. Simulation results

As mentioned in Section 4.3, CAS includes some basic routing protocols as special cases according to the number of communities in the network and the allocated number of message copies. Therefore, in this section, we compare the routing performance of CAS with the chosen routing protocols in the following two cases: single-community case and multiple-community case.

##### 5.5.1. Single-community case

We arbitrarily select one of the twelve regions for this study. The nodes deployed in this region constitute a community. For the Binary Spray-and-Wait protocol, each message is initially associated with 4 copies. Thus, at most 20% nodes can participate in message routing in Binary Spray-and-Wait.

Figure 7(a) shows the delivery ratio of the compared routing protocols. We can observe that Epidemic and Direct always achieve the best and worst delivery

ratio respectively, for all values of TTL. We also observe that the delivery ratio of Bubble is very close to that of Binary Spray-and-Wait. Binary Spray-and-Wait achieves a better delivery ratio than CAS. The maximum and minimum difference between CAS and Binary Spray-and-Wait is 7.22% and 0.28%. The delivery ratio of CAS is always higher than the predefined delivery ratio (i.e., 0.8). It is worth noting that CAS achieves the same delivery ratio as Direct when the TTL is greater or equal to 5 hours. This is because CAS allocates the same number of copies for each message as Direct. This indicates that CAS can dynamically transform to Direct.

Figure 7(b) shows the delivery cost of the compared routing protocols as a function of the TTL of the generated messages. We observe that Epidemic and Direct have the highest and lowest delivery cost respectively, whatever the value of TTL. The delivery cost of Bubble keeps increasing as the TTL increases. The delivery cost of Binary Spray-and-Wait is lower than that of Bubble and remains stable as the TTL increases. However, it is very close to the allocated number (i.e., 4) of copies for each message in Binary Spray-and-Wait. It is worth noting that unlike other routing protocols, the delivery cost of CAS decreases as the TTL increases. In particular, CAS achieves the same delivery cost as Direct when the TTL is greater or equal to 5 hours. This is because CAS can dynamically allocate copies for a message according to its TTL.

### 5.5.2. Multiple-community case

In the multiple-community case, we conduct two experiments to investigate the impact of the density of high mobility nodes and the mobility model settings on the routing performance of our protocol.

#### A. Impact of the density of high mobility nodes

In this section, we investigate the impact of the density of high mobility nodes on the routing performance of CAS and the chosen routing protocols. The investigation is conducted by two experiments, in which the densities of high mobility nodes are set as 5% and 50% respectively. In order to avoid the impact of the settings for roaming nodes on the routing performance, the values of  $p_l$  and  $p_r$  for high mobility nodes are set to 0.5. Moreover, for Binary Spray-and-Wait, each message is initially associated with 24 copies. Since there are twenty nodes in each of the twelve regions, at most 10% nodes can participate in message routing in Binary Spray-and-Wait.

Figs. 8(a) and (b) show the delivery ratio of the compared routing protocols with different densities of high mobility nodes. As the same as in the single-community case, we can observe that Epidemic and Direct always achieve the best and worst delivery ratio respectively. We also observe that CAS always achieves a much better delivery ratio than Bubble for all values of TTL. Indeed, the maximum difference between the delivery ratio of CAS and Bubble is about 69.17% and 49.17% in Figs. 8(a) and (b) respectively. This is because CAS takes advantage of gateway nodes to steer the forwarding of the messages whose source and destination nodes belong to different communities in the right

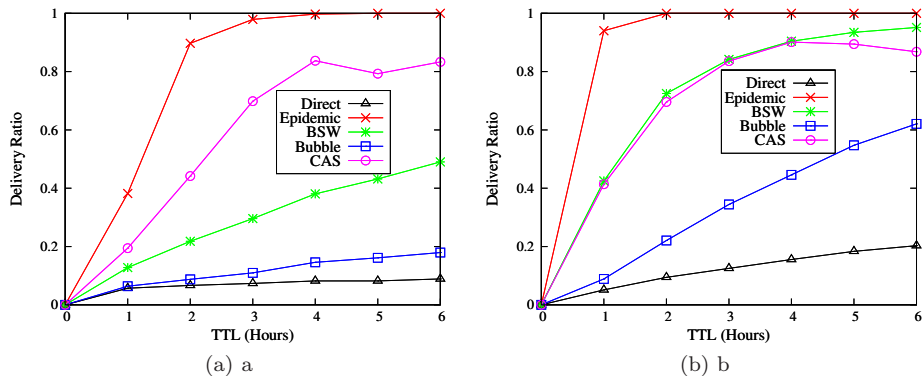


Figure 8: Comparison of delivery ratio of several algorithms with different densities of high mobility nodes. 5% and 50% nodes are chosen as the high mobility nodes in (a) and (b) respectively.

direction. Figure 8(a) shows that CAS achieves a much better delivery ratio than Binary Spray-and-Wait; while Figure 8(b) shows that Binary Spray-and-Wait even achieves a little better delivery ratio than CAS. This results from that the increasing number of roaming nodes raises the opportunity of message exchange between different communities. This also reflects the shortcoming of Binary Spray-and-Wait, that is, assuming that nodes have the same capability of visiting the entire network. Moreover, It is worth noting that delivery ratio of CAS is a little lower when TTL is 5 hours than that when TTL is 4 hours. This is because CAS allocates less copies for the messages with longer TTLs.

The performance of delivery cost of the compared routing protocols is illustrated in Figs. 9(a) and (b). Since the delivery cost of Epidemic increases much quickly than that of other protocols, we omit the delivery cost of Epidemic to illustrate the delivery cost of other protocols in detail. We can observe that Direct always has the lowest delivery cost. Bubble achieves a close delivery cost as Direct in Figure 9(a) and a little higher delivery cost than Direct in Figure 9(b). Additionally, Binary Spray-and-Wait always consumes the assigned number of message copies. Figure 9(a) shows that the delivery cost of CAS raises as the TTL increases, When the TTL is less than 3 hours. Since the TTL increase, the gateway nodes can reach more communities, which invokes the allocation of message copies. When the TTL is greater than 3 hours, the delivery cost of CAS decreases as the TTL increases. This is because CAS allocates less copies for the messages with longer TTLs. Figure 9(b) shows that the delivery cost of CAS decreases as the TTL increases. The increasing number of high mobility nodes can enhance the successful delivery probability for the messages whose source and destination nodes belong to different communities, which can terminate the allocation of message.

## B. Impact of the settings of mobility model

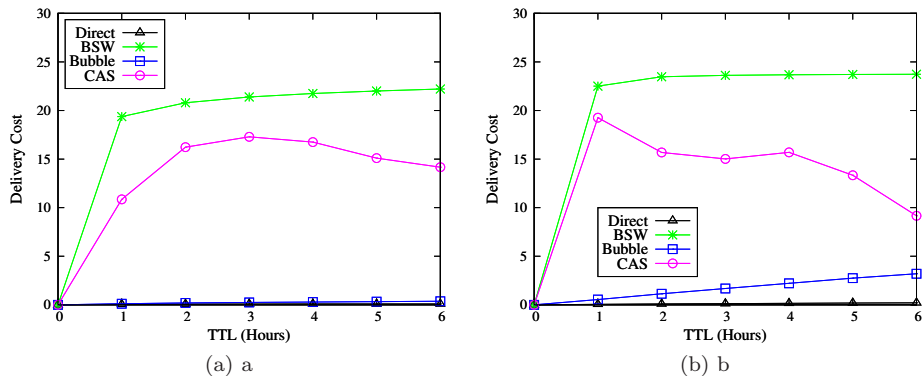


Figure 9: Comparison of delivery cost of several algorithms with different densities of high mobility nodes. 5% and 50% nodes are chosen as the high mobility nodes in (a) and (b), respectively.

In this section, we investigate the impact of the settings of the chosen mobility model on the routing performance of CAS. As we stated in Section 5.1, a high mobility node is more likely to visit its local region than other regions. Hence, for high mobility nodes, we vary the value of  $p_l$  from 0.5 to 0.9 with step by 0.1 and set the value of  $p_r$  as  $1 - p_l$  in the simulations. Moreover, 5% nodes are chosen as high mobility nodes in order to avoid the impact of the density of high mobility nodes on the routing performance of CAS.

First we look at the delivery ratio. From the results illustrated in Figure 10(a), we can observe that CAS achieves similar results under different settings with respect to  $p_l$  and  $p_r$ . The settings with  $p_l = 0.5, p_r = 0.5$  and  $p_l = 0.9, p_r = 0.1$  always achieve the best and worst delivery ratio respectively, when TTL is less than 4 hours. This is because that a higher probability  $p_r$  can provide more opportunities to route messages whose source and destination nodes are in different communities. When TTL is greater than 4 hours, the setting with  $p_l = 0.8, p_r = 0.2$  achieves the best delivery ratio while the setting with  $p_l = 0.6, p_r = 0.4$  achieves the worst delivery ratio. Since as the TTL increases, the higher probability  $p_l$  can make the gateway nodes collect more messages whose source and destination nodes are in different communities.

Next we compare the delivery cost of CAS with different settings. As shown in Figure 10(b), we can observe that CAS achieves similar results under different settings with respect to  $p_l$  and  $p_r$ . When TTL is greater than 3 hours, the setting with  $p_l = 0.5, p_r = 0.5$  has the lowest delivery cost. Since, with a higher probability  $p_r$ , the gateway nodes collect fewer messages from their community. Additionally, the gateway nodes can forward the messages whose source and destination nodes belong to different communities earlier to the next communities, which in turn results in fewer message copies are allocated in the next communities.

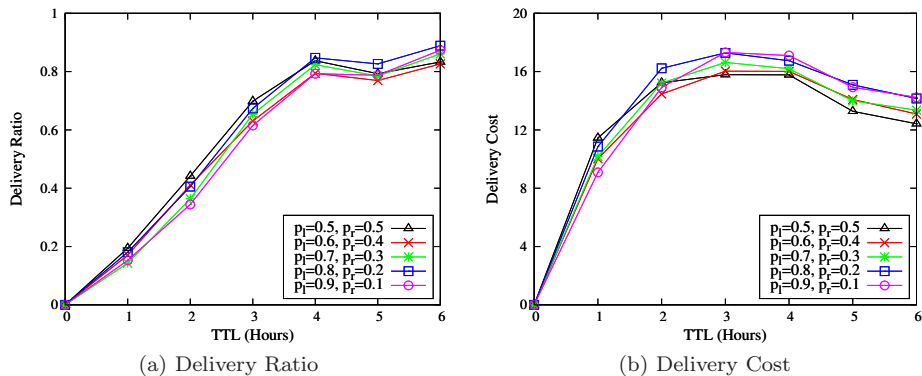


Figure 10: The impact of the settings of the mobility model on the routing performance of CAS

## 6. Conclusion

In this paper, we presented CAS, the self-regulating protocol for efficient routing in mobile delay tolerant networks. CAS controls the message replication according to the TTLs of messages, and takes advantage of the gateway nodes to steer the forwarding of the messages whose source nodes and destination nodes belong to different communities. We presented an algorithm that can dynamically compute the minimum number of message copies for each message while maintaining a predefined delivery ratio in a practical manner. We demonstrated that CAS includes some basic routing protocols as special cases by theoretical analysis and practical simulations. Our simulations on a widely utilized community-based mobility model, demonstrate that CAS can improve the routing performance compared to quota-based Binary Spray-and-Wait protocol and community-based Bubble protocol.

## Acknowledgements

This work is supported in part by the China Scholarship Council (CSC) UT-INSA Ph.D. program, the MDPS German-French Doctoral College and the French National Research Agency (SocEDA, Grant ANR-10-SEGI-013).

## References

- [1] T. Kosch, C. Adler, S. Eichler, C. Schroth, M. Strassberger, The scalability problem of vehicular ad hoc networks and how to solve it, *IEEE Wireless Communications* 13 (5) (2006) 22–28.
- [2] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, C. Diot, Pocket switched networks and human mobility in conference environments, in: *Proc. of ACM WDTN*, 2005, pp. 244–251.

- [3] K. Fall, A delay-tolerant network architecture for challenged internets, in: Proc. of ACM SIGCOMM, 2003, pp. 27–34.
- [4] T. Spyropoulos, T. Turletti, K. Obraczka, Routing in delay-tolerant networks comprising heterogeneous node populations, *IEEE Transaction on Mobile Computing* 8 (8) (2009) 1132–1147.
- [5] A. Vahdat, D. Becker, Epidemic routing for partially connected ad hoc networks, Tech. rep., Duke University (2000).
- [6] S. C. Nelson, M. Bakht, R. Kravets, A. F. Harris, III, Encounter: based routing in dtns, *Mobile Computing Communication Review* 13 (2009) 56–59.
- [7] T. Spyropoulos, K. Psounis, C. S. Raghavendra, Spray and wait: an efficient routing scheme for intermittently connected mobile networks, in: Proc. of ACM WDTN, 2005, pp. 252–259.
- [8] E. Bulut, Z. Wang, B. K. Szymanski, Cost-effective multiperiod spraying for routing in delay-tolerant networks, *IEEE/ACM Transaction on Networking* 18 (2010) 1530–1543.
- [9] C. Boldrini, A. Passarella, Hcmm: Modelling spatial and temporal properties of human mobility driven by users’ social relationships, *Computer Communications* 33 (9) (2010) 1056–1074.
- [10] T. Karagiannis, J. Le Boudec, M. Vojnović, Power law and exponential decay of intercontact times between mobile devices, *IEEE Transaction on Mobile Computing* 9 (10) (2010) 1377–1390.
- [11] W. Hsu, T. Spyropoulos, K. Psounis, A. Helmy, Modeling spatial and temporal dependencies of user mobility in wireless mobile networks, *IEEE/ACM Transaction on Networking* 17 (5) (2009) 1564–1577.
- [12] S. Lo, C. Lu, A dynamic congestion control based routing for delay-tolerant networks, in: Proc. of IEEE FSKD, 2012, pp. 2047–2051.
- [13] N. Thompson, S. C. Nelson, M. Bakht, T. Abdelzaher, R. Kravets, Retiring replicants: congestion control for intermittently-connected networks, in: Proc. of IEEE INFOCOM, 2010, pp. 1–9.
- [14] P. Hui, J. Crowcroft, E. Yoneki, Bubble rap: Social-based forwarding in delay-tolerant networks, *IEEE Transaction on Mobile Computing* 10 (11) (2011) 1576–1589.
- [15] T. Spyropoulos, K. Psounis, C. S. Raghavendra, Efficient routing in intermittently connected mobile networks: the single-copy case, *IEEE/ACM Transaction on Networking* 16 (2008) 63–76.



- [16] H. Dang, H. Wu, Clustering and cluster-based routing protocol for delay-tolerant mobile networks, *IEEE Transaction on Wireless Communication* 9 (6) (2010) 1874–1881.
- [17] J. Miao, O. Hasan, S. Ben Mokhtar, L. Brunie, A self-regulating protocol for efficient routing in mobile delay tolerant networks, in: *Proc. of IEEE DEST-CEE, 2012*, pp. 1–6.
- [18] J. Miao, O. Hasan, S. Ben Mokhtar, L. Brunie, G. Gianini, A delay and cost balancing protocol for message routing in mobile delay tolerant networks, *Tech. Rep. RR-LIRIS-2014-001, INSA Lyon* (2014).
- [19] T. Spyropoulos, K. Psounis, C. S. Raghavendra, Efficient routing in intermittently connected mobile networks: the multiple-copy case, *IEEE/ACM Transaction on Networking* 16 (2008) 77–90.
- [20] A. Elwhishi, P. Ho, K. Naik, B. Shihada, Self adaptive contention aware routing protocol for intermittently connected mobile networks, *IEEE Transactions on Parallel and Distributed Systems* 24 (7) (2013) 1422–1435.
- [21] S. Jain, K. Fall, R. Patra, Routing in a delay tolerant network, *Computing Communication Review* 34 (4) (2004) 145–158.
- [22] M. Grossglauser, D. Tse, Mobility increases the capacity of ad hoc wireless networks, *IEEE/ACM Transaction on Networking* 10 (4) (2002) 477–486.
- [23] A. Balasubramanian, B. Levine, A. Venkataramani, Replication routing in dtms: A resource allocation approach, *IEEE/ACM Transaction on Networking* 18 (2) (2010) 596–609.
- [24] A. Prodhan, R. Das, H. Kabir, G. Shoja, Ttl based routing in opportunistic networks, *Journal of Network and Computer Applications* 34 (5) (2011) 1660–1670.
- [25] M. Musolesi, C. Mascolo, Designing mobility models based on social network theory, *Mobile Computing Communication Review* 11 (3) (2007) 59–70.
- [26] P. Hui, J. Crowcroft, How small labels create big improvements, in: *Proc. of IEEE PerCom, 2007*, pp. 65–70.
- [27] Y. Li, Y. Cao, S. Li, D. Jin, L. Zeng, Integrating forwarding and replication in dtn routing: A social network perspective, in: *Proc. of IEEE VTC, 2011*, pp. 1–5.
- [28] L. Vu, Q. Do, K. Nahrstedt, 3r: fine-grained encounter-based routing in delay tolerant networks, in: *Proc. of IEEE WoWMoM, 2011*, pp. 1–6.
- [29] M. Gonzalez, C. Hidalgo, A. Barabasi, Understanding individual human mobility patterns, *Nature* 453 (7196) (2008) 779–782.

- [30] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, J. Scott, Impact of human mobility on opportunistic forwarding algorithms, *IEEE Transaction on Mobile Computing* 6 (6) (2007) 606–620.
- [31] Y. Li, G. Su, D. Wu, D. Jin, L. Su, L. Zeng, The impact of node selfishness on multicasting in delay tolerant networks, *IEEE Transaction on Vehicular Technology* 60 (5) (2011) 2224–2238.
- [32] R. Groenevelt, P. Nain, G. Koole, The message delay in mobile ad hoc networks, *Performance Evaluation* 62 (1–4) (2005) 210–228.
- [33] P. Hui, E. Yoneki, S. Y. Chan, J. Crowcroft, Distributed community detection in delay tolerant networks, in: *Proc. of ACM/IEEE international workshop on Mobility in the evolving internet architecture, 2007*, pp. 7:1–7:8.
- [34] Y. Li, G. Su, Z. Wang, Evaluating the effects of node cooperation on dtn routing, *AEU-International Journal of Electronics and Communications* 66 (1) (2012) 62–67.
- [35] A. Keränen, T. Kärkkäinen, J. Ott, Simulating mobility and dtns with the one, *Journal of Communications* 5 (2) (2010) 92–105.
- [36] J. Miao, O. Hasan, S. Ben Mokhtar, L. Brunie, K. Yim, An investigation on the unwillingness of nodes to participate in mobile delay tolerant network routing, *International Journal of Information Management* 33 (2) (2013) 252–262.
- [37] T. Spyropoulos, K. Psounis, C. S. Raghavendra, Performance analysis of mobility-assisted routing, in: *Proc. of ACM MobiHoc, 2006*, pp. 49–60.
- [38] O. Hasan, J. Miao, S. Ben Mokhtar, L. Brunie, A Privacy Preserving Prediction-based Routing Protocol for Mobile Delay Tolerant Networks, in: *Proc. of IEEE AINA, 2012*, pp. 1–8.