Cloud-Based Multicasting with Feedback in Mobile Social Networks

Yunsheng Wang, Jie Wu, *Fellow, IEEE*, and Wei-Shih Yang

***Abstract*—With the rapid growth of smartphone usage, mobile social networks (MSNs) are becoming increasingly popular. MSN can be considered as a type of delay tolerant network (DTN) which lacks continuous end-to-end connections between nodes, due to the node mobility and limited transmission range. Inspired by the homophily of social networks that friends are usually similar in characteristics, we present a novel concept – *cloud*, where the nodes in frequent contact with the destinations will form destination clouds. Neighbors in the destination cloud have a special status that can forward the message to the destination directly. We propose a cloud-based multicast scheme with feedback in MSNs with two phases: *pre-cloud* and *inside-cloud*. In the pre-cloud process, the message holder will forward the copy of the multicast message to the encountered node, based on a given forwarding metric. The forwarding metric can be iteratively refined from a feedback control mechanism. In the inside-cloud process, the message holder will wait until it meets with the destinations. We analytically formulate the multicast problem into a continuous Markov chain problem, and formally analyze the latency in this model. Extensive trace-driven simulations show that our scheme significantly improves the performance compared to existing schemes.**

***Index Terms*—Cloud, delay tolerant networks, feedback control, Markov chain, mobile social networks, multicast.**

I. Introduction

As stated in Nielson's report: "as of Q4 2011, 46 percent of US mobile consumers had smartphones, and that figure is growing quickly. In fact, 60 percent of those who said they got a new device within the last three months chose a smartphone over a feature phone" [1]. The mobile social applications in smartphones consume a large amount of bandwidth within the cellular networks. However, the cellular networks are currently overloaded. Thus, it is imperative to develop novel architectures and protocols to solve this problem. A mobile social network (MSN) [2], [3] is a special type of delay tolerant network [4], where the mobile devices contact each other occasionally through opportunistic contacts. The content can be shared among the mobile devices instead of directly downloaded from the infrastructure.

There are multiple types of content to be delivered in MSNs, such as newspapers, online social networking updates, and weather information. The service providers may deliver the content to only a small fraction of mobile users. Then, these content holders walk around and contact other mobile users opportunistically, and exchange the content through either WiFi or Bluetooth. As far as delivery time and cost are concerned, designing an efficient routing protocol to deliver the content to the interested users becomes a multicast problem in MSNs.

In this paper, we consider the multicast problem in MSNs. Multicast is a service where a source node sends the message to all of the destinations. In MSNs, a relay node is an encountered node that can store-carry-and-forward the message to the destinations quickly with a small overhead. Therefore, designing a suitable criterion to select good relay nodes becomes a challenging problem in MSN multicast routing.

Due to the limited buffer space of the devices and bandwidth, it is hard for the mobile devices to obtain the global information. Therefore, designing a distributed routing algorithm in MSNs becomes a hard problem. Several DTN routing schemes have been proposed recently [5], [6], [7], [8], [9], [10], [11]. However, no efficient multicast scheme has been proposed for MSNs. Because of the high mobility and intermittent connection characteristics of MSNs, the delivery rate is low and the latency is high, unless high cost routing schemes are implemented. In this paper, we propose a cost effective cloud-based multicast scheme with feedback in MSNs.

The nodes that contact each other more frequently will have a higher probability of exchanging the messages, which we call *friends*. Each node has its frequent contact friends which, together, form a community. Here, we consider the destination community, which has been termed by a new concept – *cloud*. If the nodes contact the destination with high frequency, these nodes have more opportunities to forward the message to the particular destination. We call these kinds of nodes: *destination neighbors*. Destinations and destination neighbors will constitute *destination clouds*. One of the most basic notions governing the structure of social networks is *homophily* [12]. Friends are usually similar in their characteristics, including the places they live, their interests, etc. Therefore, the individuals with more common interests have more of a chance to contact each other. In our proposed multicast scheme, the nodes only obtain their local neighbor information. As shown in Fig. 1, N_1^1, N_2^1, N_3^1, and N_4^1 are destination neighbors of the destination D_1. They form a destination cloud C_1 for D_1. N_1^2, N_2^2, and N_3^2 are destination neighbors of the destination D_2. They form a destination cloud C_2 for D_2. We can see that N_3^1 and N_1^2 represent the same node, which belongs to both C_1 and C_2.

As shown in Fig. 1, our proposed multicast scheme has two steps: *pre-cloud* and *inside-cloud*. In the pre-cloud process,

Manuscript received October 1, 2012; revised April 9 and July 9, 2013; accepted September 20, 2013. The associate editor coordinating the review of this paper and approving it for publication was T. Hou.

Y. Wang is with the Department of Computer Science, Kettering University, Flint, MI 49504, USA (e-mail: ywang@kettering.edu).

J. Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA (e-mail: jiewu@temple.edu).

W.-S. Yang is with the Department of Mathematics, Temple University, Philadelphia, PA 19122, USA (e-mail: yang@temple.edu).

Digital Object Identifier 10.1109/TWC.2013.102313.121508

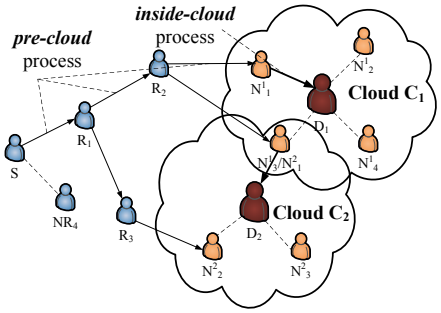


Fig. 1. Cloud-based Multicast in MSNs.

initially, the source S carries a certain amount of copies. When it meets an encountered node, the message forwarding decision is based on the forwarding metric F, which measures the quality of reaching the destinations. The number of forwarding copies is proportional to the forwarding metrics of these two encountered nodes. The node with the new copies becomes a relay node. In Fig. 1, R_1 is a good relay node for S ($F_{R_1} > F_S$). Thus, S will forward portions of the copies of the message to R_1. S and NR_4 come in contact with each other. However, NR_4 is not a good relay node for S ($F_{NR_4} < F_S$). Therefore, S will not forward any copy of the message to NR_4. When the message holder comes into contact with one of the destination neighbors, the number of copies being forwarded to this destination neighbor depends on the number of destination clouds it belongs to. N_1^1 belongs to cloud C_1. Therefore, it will receive one copy from R_2. N_3^1/N_1^2 belongs to two destination clouds, C_1 and C_2. Therefore, R_2 forwards two copies to it. If the encountering node is one of the destinations, the message holder will forward one copy to it. In the inside-cloud process, the destination neighbors with the message only forward the message to the destinations directly.

Recently, many forwarding metrics have been introduced for routing guidance, such as contact frequency or available buffer space. However, there is no optimal metric. Thus, in this paper, we present a novel feedback control mechanism. Initially, the forwarding metric is the total number of contacts with all of the destinations in a fixed time interval. After a successful multicast process, we can capture the shortest multicast time for each node to forward the message to the destinations. In the future multicast process, we will use the reciprocal of the shortest captured multicast time in the previous round as the new forwarding metric for the future routing guidance. In this feedback control mechanism, the shortest forwarding time from the node to the destinations can be iteratively refined and used for the future multicast process. The simulation results will show that the routing performance can be improved round by round until an optimal result is reached.

The major contributions of our work are as follows:

- We introduce a novel concept: *cloud*. The nodes that have high contact frequency with the destination can quickly forward the message to the destination directly. Hence, destination neighbors and the destination constitute the destination cloud.
- We present a cost effective cloud-based multicast scheme with feedback that has two stages: pre-cloud and inside-cloud.
- We propose a feedback control mechanism that uses the previous multicast latency to guide the current multicast.
- We formally analyze the latency of multicast by formulating a continuous Markov chain model. The feedback control mechanism is motivated by the analytical results.
- We evaluate the proposed scheme with both synthetic and real traces. The simulation results show the competitive performance of our proposed scheme in MSNs.

The remainder of this paper is organized as follows: Section II reviews the related work. Section III shows the preliminary work. Section IV describes the details of the cloud-based multicast scheme with feedback. Section V formally analyzes the MSN multicast, and provides the motivation for the feedback control mechanism. Section VI focuses on the simulation and evaluation. We conclude our work in Section VII.

II. Related Work

In recent years, there have been many papers focused on designing an efficient routing scheme for DTNs. Initially, the simple flooding approaches are introduced, such as epidemic routing [6], spray-and-wait [7], RAPID [13], and so on. However, these protocols all have a high cost in order to achieve a high delivery rate and small latency. Then, some history and encounter-based approaches are proposed to improve the routing performance, such as MaxProp [5], PROPHET [14], and delegation forwarding [15]. In these papers, some forwarding metrics are used to control the number of copies. The forwarding metric can be the number of contacts with other nodes or with the destinations, the recent time contact with the destination, and so on.

There have been recent works which consider multicasting in DTNs. The existing research focuses on three models: *single node* [16], [17], *multiple copies* [9], [18], [19], and *single copy* [20], [8], [21] models. In the single node model, one single node holds all destinations, and delivers them to each destination as they contact one another, through movement. In [16], Zhao et al. proposed the basic single node model together with new semantics for DTN multicasting, which explicitly specify temporal constraints on group membership and message delivery. Yang and Chuah [17] presented a two-stage single node model, where routes to destinations are first identified through a ferry, followed by the message delivery along the discovered routes. In the multiple copies model, the destination set is replicated at a contact once a certain condition related to the quality of the encountered node is satisfied. In [18], Lo and Luo extended their work [22] from unicast to multicast in DTNs, which takes advantage of the contact behavior in DTN environments, as to predict good message relay nodes that help to forward messages to their destinations. The approach dynamically adjusts the number of message copies generated, depending on different network situations. In [19], the authors implemented the multicast routing protocol in vehicular delay tolerant networks, which was a combination of geographic-based routing [23], PROPHET [14], and spray-and-wait [7] protocols. A differentiation between dense and sparse scenarios was used, according to an estimation of the number of nodes met. In the single copy model, there is only one copy for each destination, and destinations can be

scattered at different nodes. In [20], Gao et al. developed a single copy model where the forwarding metric is based on the social network perspective, which is a social-based approach, based on a notion of "ego-centric betweenness" in order to optimize multicast performance in DTNs. In [8], Wu and Wang proposed and applied the destination set splitting in DTN multicasting, which is based on the tree structure in order to improve the performance. In [21], the authors studied the DTN multicast problem from the graph indexing point of view. They solved the problem of minimizing the remote communication cost for multicast in DTNs. The authors analyzed this problem in the case of scheduled trajectories and known traffic demands, and proposed a solution based on a novel graph indexing system.

Mobile social networks have caught increased amounts of attention in recent years. Miluzzo et al. investigated a CenceMe application in mobile devices, using sensor-enabled mobile phones to share information through social network applications [24]. Bulut and Szymanski introduced a friendship-based routing scheme for MSNs. They use a metric to accurately detect the quality of friendship, and to improve the routing performance [25]. Han et al. proposed a new MSN application scenario, where service providers deliver information to only a small fraction of target-users; then, these target-users opportunistically propagate the information to other mobile users in MSNs [26]. Wu and Wang used individuals' social feature information to form a hypercube, and proposed a hypercube routing scheme for MSNs [11]. In [3], Hui et al. used the social network concept of centrality and community to design a social-based forwarding for DTNs. However, these methods are hard to implement in a distributed way, and it is costly to obtain the relevant information. In this paper, we introduce a novel way to use the information, differing from the previous routing process, to improve the current multicast forwarding performance.

In [27], Jones et al. proposed a minimal estimated expected delay (MEED) routing scheme for DTNs, which is an extension work of minimal expected delay (MED) by Jain et al. [28]. MEED computed the expected delay by using the observed contact history, in which a node records the connection and disconnection time of each contact over a sliding history window [27]. MEED and MED consider the expected delay of each contact, not the delay of the whole routing process. In our proposed feedback control mechanism, we use the previous routing process delay to guide the current progress.

III. Preliminaries

A. System Model

Assume that there are n nodes in the whole network. The source node is denoted as S. The destination set of a multicast is represented as $\{D_1, D_2, ..., D_m\}$. Therefore, there are $n-m-1$ relay nodes $\{R_1, R_2, ..., R_{n-m-1}\}$. Each message has a timestamp, which records each time point at which it travels from the source to the destinations. Each node records the number of times in which it makes contact with other nodes. Each node has a forwarding metric table recording the forwarding metrics for itself and other nodes.

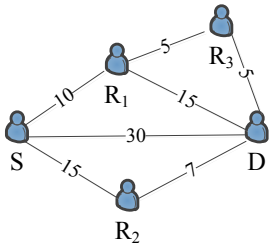


Fig. 2. Motivation of the feedback mechanism: the values between the mobile nodes are the inter contact time.

B. Motivation of the Feedback Mechanism

Most of the previous DTN routing protocols are based on the deterministic metrics, such as inter contact time, or number of contacts. As shown in Fig. 2, the source S has a message for the destination D. When S meets with nodes R_1 and R_2, if the forwarding metric is based on the inter contact time between each node to the destination, R_2 is a better relay node (path $S \rightarrow R_2 \rightarrow D$ with a delay of 22, which is shorter than path $S \rightarrow R_1 \rightarrow D$ with delay of 25). However, R_1 has another path to D through R_3, which has a shorter delay (path $S \rightarrow R_1 \rightarrow R_3 \rightarrow D$ with a delay of 20). Since we considered the delivery delay as a measurement metric in this example, the path through $S \rightarrow R_1 \rightarrow R_3 \rightarrow D$ has the shortest inter-contact time. Therefore, we chose this path as the forwarding path. If we can design a feedback mechanism which updates the forwarding metric according to the delay of the previous round, the routing performance can be improved. We assume that each pair of nodes has a similar contact frequency in the current time period as in the previous time period.

C. Main Idea

The destination neighbors in the destination cloud are the nodes that have high contact frequency with a particular destination. The multicast process has two stages: pre-cloud and inside-cloud. In the pre-cloud stage, the message forwarding is a multi-copy partition process. The message partition rule is based on the forwarding metrics. When the message has been received by one of the destination neighbors, then we transform to the inside-cloud stage: the message will only be forwarded to the destinations directly.

In the feedback control mechanism, when the destination receives the message, it will share the feedback information about the relay nodes from the previous round to the nodes it encounters in the future. Then, these relay nodes update their forwarding metrics based on the feedback information. The forwarding metrics are then refined through iterative feedback.

IV. Cloud-based Multicast

A. Cloud Formation

First, we define a new concept – *cloud*. The nodes that contact each other more frequently will have a higher probability of exchanging the messages, which we call 'friends.' Each node has its frequent contact friends which, together, form a community. Here, we consider the destination community, which has been termed by a new concept - cloud. If the nodes contact the destination with high frequency, these nodes have more opportunities to forward the message to the particular

destination. We call these kinds of nodes: *destination neighbors*. We use a contact frequency threshold (θ) to control the size of the cloud. We assume that a destination neighbor with a copy of a message waits until meeting with the destination, and gives the message to the destination directly. Therefore, if the contact threshold is very small, it will form a large cloud. Although it can reduce the number of forwardings, it will also increase the latency. If the contact threshold is very large, it will form a small, tight cloud. Therefore, it will dilute the role of the cloud. Thus, our first problem is to find an effective threshold for the formation of a cloud. We will compare different values for the contact threshold in the simulation.

B. Forwarding Metric

The forwarding metric is used to measure the capability of a node to forward the message to the destinations. Intuitively, to improve the performance of MSN, if a node i (with a copy of the message) encounters a relay node j (without a copy of the message), i should give a copy of the message to j only if the forwarding metric of j is larger than the forwarding metric of i. The question is, what is the definition of a good forwarding metric? Is there an optimal forwarding metric? Thus, our second problem is to find an effective definition of forwarding metric, or ultimately, an optimal forwarding metric.

We will use the feedback mechanism to approximate an optimal forwarding metric. We will update the definition of forwarding metric after each multicast process is nearly completed, and use the updated forwarding metric for the next round. We then iterate this updating procedure many times. The iterative updating procedure will eventually give us an approximation to the optimal forwarding metric. To this end, let us denote $F_i(l)$ to be the forwarding metric of node i in the l-th round, $l = 1, 2, ...$ The initial forwarding metric $F_i(1)$ is defined as follows. We consider in a fixed time interval, a matrix representing quantities proportional to the contact probabilities between pairwise nodes:

$$\begin{pmatrix} a_{11} & . & . & . & a_{1n} \\ . & . & & & . \\ . & & . & & . \\ . & & & . & . \\ a_{n1} & . & . & . & a_{nn} \end{pmatrix}.$$

Here, for $i \neq j$, a_{ij} can be the contact frequency, or inter contact time, etc., between nodes i and j. We also let $a_{ii} = -\sum_{j \neq i} a_{ij}$. For a concrete example, we may as well let a_{ij} be the contact frequency between i and j for the rest of the paper.

Initially, we let the forwarding metric $F_i(1)$ be the sum of a_{ij} over all destination nodes j. In the next two subsections, we will describe the mutlicast process and how to update the forwarding metric in the feedback mechanism.

C. Multicast Process

There are two steps to complete the multicast process: *pre-cloud* and *inside-cloud*.

Algorithm 1 Pre-cloud Stage

/* A non-destination neighbor i with $Z(i)$ copies of a message, encounters node j without that message. */
/* Their forwarding metrics are F_i and F_j, respectively. */
if j is one of the destinations **then**
  Forwarding one copy to j.
else
  if j belongs to one or more than one of the destination clouds **then**
    Forwarding $min(Z(i), e(j))$ number of copies to j, where $e(j)$ is the number of destination clouds j belongs to.
else
  if $F_j > F_i$ **then**
    Forwarding $\left\lceil \frac{F_j}{F_i+F_j} \cdot Z(i) \right\rceil$ number of copies to j.

In the multicast process, we assume that each node i keeps its metadata, which contains its forwarding metric and $e(i)$, where $e(i)$ is the number of destinations to which i belongs. Let $Z(i)$ be the number of message copies possessed by node i. In the pre-cloud stage, if a message holder i (with $Z(i) > 0$) is outside of every destination cloud, when it meets node j (with $Z(j) = 0$), they first exchange their metadata. If j is a destination node, then i gives a copy to j. If j belongs to one or more destination clouds, then i gives j $min(Z(i), e(j))$ copies. If j is not in any destination cloud and their forwarding metrics satisfy $F_j > F_i$, then i gives j $\left\lceil \frac{F_j}{F_i+F_j} \cdot Z(i) \right\rceil$ copies. If j is not in any destination cloud and satisfies $F_j \leq F_i$, then i does not give j any copies. Algorithm 1 shows the whole process of the pre-cloud stage.

We note that initially, the forwarding metric of node i, $F_i(1)$ is the total contact frequency of node i with all destinations in the fixed time interval. When the multicast process is nearly completed, the forwarding metric will be updated to $F_i(2)$ for the next round. We record the number of destinations which have already received the message. If the number reaches a predefined threshold, we suppose this round of multicast process is nearly completed. In this paper, we predefine the threshold to be half of the number of destinations. Repeating this process, we will have a sequence of the forwarding metric $F_i(l)$, for the $l+1$-th round, $l = 1, 2, ...$ The details will be described in the feedback mechanism in the next subsection.

In the inside-cloud stage, if a destination neighbor has a copy of a message, it only forwards one copy to the destination directly. This process can dramatically reduce the number of forwardings, which is considered to be the cost of the multicast process. Here, the destination node can also act as a relay node.

When the storage of the mobile node is full, the message discard policy will follow FIFO, which means that the first coming message will be discarded.

D. Feedback Control Mechanism

In this part, we describe in detail our proposed feedback control mechanism to improve the accuracy of the forwarding metric prediction.

In each round, there is a new multicast message generated. Therefore, each round is considered as an independent multicasting process. For each message, we assign a timestamp, which records the exact time at which the message has been forwarded from one node to another. When the message

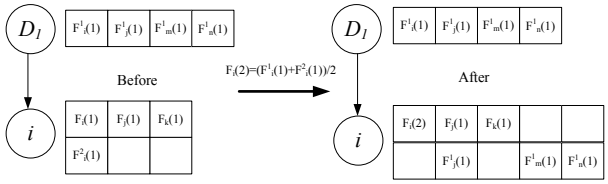


Fig. 3. Destination feedback process.

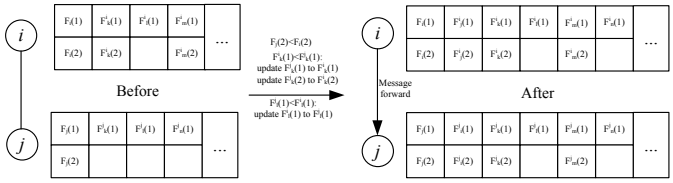


Fig. 4. Metadata update process.

arrives at one of the destinations, it calculates the latency for all nodes in the routing path from the source to the destination, and records the results in the destination's metadata. After that, the destination node gives the metadata to other nodes when they contact each other. Thus, each encountered node has the metadata that records its multicast latency of the previous round in an additional space. The metadata also contains the same information for other nodes. Therefore, the feedback information is quickly spread out through the whole network. After a node has obtained the metadata (of the previous round) from more than a given threshold number of destinations, it updates its forwarding metric by using the average latency to the destinations. Once the forwarding metric is updated, it is then used in the pre-cloud algorithm for the next round. In our simulation, we set the threshold to be half the number of the destinations.

In Fig. 3, for example, destination D_1 calculates the latency for relay nodes i, j, m, and n; all of these nodes have participated in the previous multicast process. After that, D_1 has a contact with i. In this example, we suppose that the threshold to update the forwarding metric is 2. Suppose that before D_1 and i make contact, i has already received a feedback from destination D_2. Therefore, i will update its forwarding metric to $F_i(2)$, which is equal to the average of two feedbacks, $F_i(2) = \frac{F_i^1(1)+F_i^2(1)}{2}$, where $F_i^j(l)$ is the latency of the delivery of a message sent from i to destination j using forwarding metric $F_i(l)$ in the l-th round. Note that i also has information similar to that of some other nodes.

Fig. 4 illustrates the metadata exchange process between two non-destination nodes. After exchanging metadata, these two encountered nodes will record the updated information not only for themselves, but also for all other nodes previously encountered by these two nodes. For example, before this contact, i has the information about k of the first round, $F_k(1)$, and j has $F_k(2)$. After exchanging the metadata, i will update its forwarding metric table by recording the information of k in the second round in its additional buffer space. Hence, the metadata only consumes a small amount of bandwidth.

V. Analysis

A. Multicast without cloud

As discussed in Section III-A, if we do not consider the destination cloud concept, the n-node MSN can be represented as $\{N_1, N_2, ..., N_n\} = \{S, R_1, R_2, ..., R_{n-m-1}, D_1, D_2, ..., D_m\}$. For simplicity of writing formulas, we will relabel $\{S, R_1, R_2, ..., R_{n-m-1}, D_1, D_2, ..., D_m\}$ as $\{1, 2, ..., n\}$, so that $i = 1$ is representing the source node S, $i = 2, ..., n-m$ are representing the relay nodes $R_1, R_2, ..., R_{n-m-1}$, and $i = n-m+1, ..., n$ represent the destination nodes $D_1, D_2, ..., D_m$, respectively. Then we can model the multicast routing process as a continuous time Markov chain, as follows.

We consider a fixed time interval, and for $i \neq j$, let a_{ij} be the contact frequency of node i to node j in the fixed time interval. We define $a_{ii} = -\sum_{j \neq i} a_{ij}$. Then the contact table A is an $n \times n$ matrix,

$$A(i,j) = \begin{pmatrix} a_{11} & . & . & . & a_{1n} \\ . & . & & & . \\ . & & . & & . \\ . & & & . & . \\ a_{n1} & . & . & . & a_{nn} \end{pmatrix},$$

that satisfies $a_{ij} \geq 0$, if $i \neq j$ and $a_{ii} \leq 0$, and $\sum_j a_{ij} = 0, \forall i$.

It is well-known that the matrix A can be considered as a generator of a continuous time Markov chain [29] where each node i waits for a random time ξ_i, then contacts a node j. Here, ξ_i follows an exponential distribution $\xi_i \sim exp(-a_{ii})$ [30], [31], [32], i.e., $P(\xi_i > t) = e^{a_{ii}t}$, and ξ_i, $i = 1, 2, ..., n$, are independent. It is well-known that if, at a certain time i is contacting, then the conditional probability that node j is contacted by i is:

$$\begin{aligned} P(j|i) &= \frac{a_{ij}}{-a_{ii}}, \quad i \neq j, \\ P(i|i) &= 0. \end{aligned} \tag{1}$$

When there is no restriction on which node is contacting which nodes, the first contact time of the whole system after $t = 0$ is $\eta_1 = min(\xi_k; k = 1, ..., n)$. Let $\eta_0 = 0$ and η_{t+1} be the first contact time after η_t, $t = 0, 1, 2....$ Then the successive contact times η_t, $t = 0, 1, 2...$, form an increasing sequence of random times. Therefore, if we only observe what happens at contact times, then we obtain an embedded discrete-time Markov chain that describes the transitions of the number of copies of the messages in the multicast routing process. To this end, we let $Z_t(i)$ be the number of copies that node i has at time t, $t = 0, 1, 2, ...$ and put $Z_t = (Z_t(1), Z_t(2), ..., Z_t(n))$. Initially, we suppose that there are L copies in the source node, so $Z_0 = (L, 0, 0, ..., 0)$. Since the total number of copies is unchanged, $Z_t \in \Omega$, for all t, where $\Omega = \{Z; \sum_{i=1}^n Z(i) = L\}$ is the state space of the discrete-time Markov chain.

Let F_i be the forwarding metric of node i. Intuitively, in the routing process, i is more efficient than j if $F_i > F_j$. Therefore, to obtain an efficient routing algorithm, i should give more copies of the message to j only if $F_i < F_j$ when i and j are in contact.

Let $S_t = \{i = 1, 2, ..., n-m; Z_t(i) \geq 1\}$ be the *message holder set* at time t. For each $i \in S_t$, let $R_t(i) = \{j = 2, .., n-m, F_j > F_i, Z_t(j) = 0, Z_t(i) \geq 2\} \bigcup \{n-m+1, ..., n\}$ be the *receiver set of* i at time t. Note that if $i \in S_t$ and $Z_t(i) = 1$, then $R_t(i) = \{n-m+1, ..., n\}$. Also if $i \notin S_t$, then $R_t(i) = \emptyset$.

Suppose at time t, nodes i and j come in contact with each other. If i is in the message holder set and j is in the receiver set of i, then i splits the message copies into two parts,

according to the ratio of the forwarding metrics, and gives one part to j. Suppose that $Z_t = Z$ before i and j have a contact, then after the contact, the state becomes $Z_{t+1} = Z^{ij}$, where:

$$Z^{ij}(k) = \begin{cases} Z(k), & \text{if } k \neq i, j, \\ \left\lfloor \frac{F_i}{F_i+F_j} \times Z(i) \right\rfloor, & \text{if } k = i, \\ \lceil \frac{F_j}{F_i+F_j} \times Z(i) \rceil, & \text{if } k = j, \end{cases} \quad (2)$$

for $i \in S_t$, $j \in R_t(i)$. Thus Z^{ij} is the new state obtained by state Z after i and j have exchanged messages. In Eq. 2, if $k = j$, which means k is in the receiver set of i, k will receive $\left\lceil \frac{F_j}{F_i+F_j} \right\rceil$ in the third equation. At the same time, the message holder i will take the remainder.

Suppose that at time t with state Z_t, $i \in S_t$ is a message holder, and i is the first node in contacting with other nodes; then i has to wait until it meets with a node $j \in R_t(i)$ to proceed. If j is not in $R_t(i)$, then i does not give any copy to j. In terms of Markov chain, this mechanism is equivalent to imposing a broken-link condition for the Markov chain on the links that go from i to any node in $(R_t(i))^c$, the complement of the receiver set of i, by setting $A(i, j) = 0$, for all $j \in (R_t(i))^c$. Then the Markov generator with such broken-link condition is given by

$$A_{Z_t}(i,j) = \begin{cases} A(i,j), & \text{if } i \in S_t, j \in R_t(i), \\ 0, & \text{if } i \in S_t, j \notin R_t(i), i \neq j, \\ \sum_{k \notin R_t(i)} A(i,k), & \text{if } i = j, i \in S_t, \\ A(i,j), & \text{if } i \notin S_t. \end{cases} \quad (3)$$

Here, as we can see from the second equation of Eq. 3, $A_{Z_t}(i, j)$ is set to zero if i is a message holder and j is not in the message receiver set of i. Otherwise, if $i \neq j$, then $A_{Z_t}(i, j)$ is unchanged, as seen from the first and the fourth equation of Eq. 3. Since the sum over a row of a Markov generators is always zero, the third equation of Eq. 3 is just for re-normalization. Algorithm 2 shows the whole process.

By the Markov property and Eq. 1, the probability of routing process that goes through a path of states $(Z_0, Z_1, ..., Z_n)$ is $P(Z_0, Z_1, ..., Z_n) = P(Z_0, Z_1)P(Z_1, Z_2) \cdots P(Z_{n-1}, Z_n)$, where:

$$P(Z_t, Z_{t+1}) = \begin{cases} \frac{A_{Z_t}(i,j)}{-\sum_{k \in S_t} A_{Z_t}(k,k)}, & \text{if } Z_{t+1} = Z_t^{i,j}, \\ 0, & \text{if } Z_{t+1} \neq Z_t^{i,j}. \end{cases} \quad (4)$$

Given time t with state Z_t, the first contact time for the routing is $\eta_{Z_t} = min(\xi_k; k \in S_t)$, where ξ_k's are independent exponential distributed random variables, $\xi_k \sim exp(-A_{Z_t}(k, k))$. The expected waiting time for the first contact after time t is $T_{Z_t} = E(\eta_{Z_t}) = \frac{1}{-\sum_{k \in S_t} A_{Z_t}(k,k)}$. Then, the total multicast process time is $T = T_{Z_0} + T_{Z_1} + ... + T_{Z_{\tau-1}}$, where $\tau = min\{t; t \geq 0, Z_t(j) > 0, \forall j = n - m + 1, ..., n\}$ is the completion time of the multicast routing process. The expected latency is then given by:

$$E(T) = \sum_{Z_0 ... Z_{\tau-1}} (T_{Z_0} + T_{Z_1} + ... + T_{Z_{\tau-1}}) P(Z_0 ... Z_{\tau-1}). \quad (5)$$

B. Multicast with cloud

When we include the cloud concept, there is a new type of node: destination neighbor. There are many ways to define what the cloud of a destination is. Here we consider a threshold model. Let θ be a preset (fixed) threshold value. We say that a node j, $j = 2, ..., n - m$, is in the cloud of a destination node d if $a_{jd} > \theta$. Let $C_d = \{j = 2, ..., n-m; a_{jd} > \theta\}$ denote the cloud of the destination node d, $d = n - m + 1, ..., n$. Note that C_d's may be overlapped. Let p be the cardinality of $\cup_{d=n-m+1}^{n} C_d$. Then the set of the relay nodes can be rewritten as $\{R_1, R_2, ...R_{n-m-1}\} = \cup_{d=n-m+1}^{n} C_d \cup \{\mathbb{R}_1, ...\mathbb{R}_{n-m-p-1}\}$; here we have used a new notation (in open-faced R) and relabeled the non-destination neighbor relay nodes. Let $\mathbb{C} = \cup_{d=n-m+1}^{n} C_d$ be the destination neighbor set and $\mathbb{R} = \{\mathbb{R}_1, ...\mathbb{R}_{n-m-p-1}\}$ be the set of non-destination neighbor relay nodes.

The differences between using cloud and not using cloud lie in the message partition, the receiver set $R_t(i)$, and where to set the broken-link conditions for the Markov generator at time t. In what follows, we continue to use the notation $e(i)$ for the number of destination clouds to which node i belongs.

For the cloud model, we first need to define the receiver set of i at time t, $R_t(i)$. Suppose at time t the state is $Z_t = Z$. The message holder set $S_t = \{i = 1, 2, ..., n - m; Z_t(i) \geq 1\}$ is the same as in the non-cloud model. Let $i \in S_t$. If $i \in C_d$ for some d, then $R_t(i) = \{d = n-m+1, ..., n; i \in C_d\}$ is the set of all destinations, such that i belongs to their clouds. If $i \in S_t$, but not in any cloud, then $R_t(i) = \{j = 2, .., n - m, F_j > F_i, Z_t(j) = 0, Z_t(i) \geq 2\} \bigcup \{n - m + 1, ..., n\}$ is the same as in the non-cloud model.

Next, we set the rules for the message partition. Suppose i contacts j at time t, then after the contact, the state is changed to a new state Z^{ij} according to the following rules.

Let $i \in S_t$ and $j \in R_t(i)$ $(i \neq j)$. Then

1) If $k \neq i, j$, then the value of $Z^{ij}(k) = Z(k)$, which means that k is not the contact node, so the copy in k's buffer will not change.
2) If $k = j$, then there are 2 situations:
   a) If i is in the cloud of d for some d, then j must be a destination node and $Z^{ij}(j) = 1$ and $Z^{ij}(i) = Z(i) - 1$;
   b) If i is not in any cloud, then

Algorithm 2 Analysis of Multicast Routing Without Cloud

```
/* When node i ∈ S_t with Z(i) copies of a message encounters node
j ∈ R_t(i) at time t. */
if k = i then
    Z_{t+1}(k) = ⌊ F_i/(F_i+F_j) × Z_t(i) ⌋.
else
    if k = j then
        Z_{t+1}(k) = ⌈ F_j/(F_i+F_j) × Z_t(i) ⌉.
    else
        Z_{t+1}(k) = Z_t(k)
/* Update S_{t+1}, R_{t+1}(i) and Markov generator A */
if i ∈ S_{t+1}, j ∈ R_{t+1}(i) then
    A_{Z_{t+1}}(i, j) = A_{Z_t}(i, j).
else
    if i ∈ S_{t+1}, j ∉ R_{t+1}(i), i ≠ j then
        A_{Z_{t+1}}(i, j) = 0.
else
    if i = j, i ∈ S_{t+1} then
        A_{Z_{t+1}}(i, j) = Σ_{k∉R_{t+1}(i)} A(i, k).
    else
        A_{Z_{t+1}}(i, j) = A_{Z_t}(i, j)
```

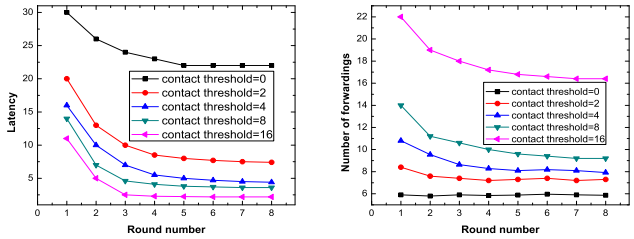


Fig. 5. Comparing the performance in the 4 destinations situation: (L): latency and (R): number of forwardings.

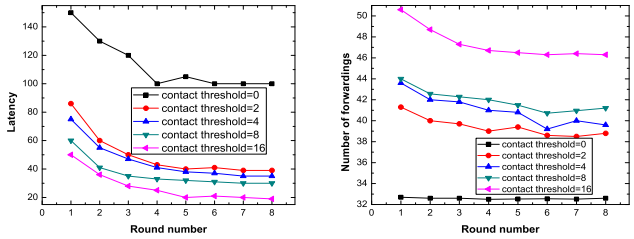


Fig. 6. Comparing the performance in the 32 destinations situation: (L): latency and (R): number of forwardings.

i) if j belongs to one or more destination clouds, then i forwards j $min(Z(i), e(j))$ copies;
ii) if j is a destination node ($j \in D$), then i forwards one copy to j;
iii) if j is not in any destination cloud, then i gives j $\left\lceil \frac{F_j}{F_i+F_j} \cdot Z(i) \right\rceil$ copies.

3) If k is the sender ($k = i$), then the number of copies remaining in i's buffer is the remaining part of the initial $Z(i)$ subtracted by the amount forwarded to receiver j, as discussed above.

In summary, if at time t the state is $Z_t = Z$ and i, j are the first pair in contact after time t, then after contact the new state Z^{ij} becomes

1) For $i \in \cup_d C_d$, $j \in R_t(i)$,

$$Z^{ij}(k) = \begin{cases} Z(k), & \text{if } k \neq i, j, \\ 1, & \text{if } k = j, \\ Z(i) - 1, & \text{if } k = i; \end{cases} \tag{6}$$

2) For $i \notin \cup_d C_d$, $j \in R_t(i)$,

$$Z^{ij}(k) = \begin{cases} Z(k), & \text{if } k \neq i, j, \\ min(Z(i), e(j)), & \text{if } k = j, j \in \mathbb{C}, \\ 1, & \text{if } k = j, j \in D, \\ \lceil \frac{F_j}{F_i+F_j} \times Z(i) \rceil, & \text{if } k = j, j \notin \mathbb{C} \cup D, \\ Z(i) - min(Z(i), e(j)), & \text{if } k = i, j \in \mathbb{C}, \\ Z(i) - 1, & \text{if } k = i, j \in D, \\ Z(i) - \lceil \frac{F_j}{F_i+F_j} \times Z(i) \rceil, & \text{if } k = i, j \notin \mathbb{C} \cup D. \end{cases} \tag{7}$$

Since a node i has to wait until it contacts a node j in $R_t(i)$ to proceed, any other contacts before that time do not contribute to our multicast process. Therefore, the Markov generator with broken-link conditions is given by

$$A_{Z_t}(i,j) = \begin{cases} A(i,j), & \text{if } i \in S_t, j \in R_t(i), \\ 0, & \text{if } i \in S_t, j \notin R_t(i), i \neq j, \\ \sum_{k \notin R_t(i)} A(i,k), & \text{if } i = j, i \in S_t, \\ A(i,j), & \text{if } i \notin S_t. \end{cases} \tag{8}$$

Note that the Markov generator with broken-link conditions has the same form as that of the non-cloud model, Eq. 3. However, since the receiver sets are different, the Markov generators are actually different between cloud and non-cloud models.

The formula for the expected latency for the cloud model is the same as the one in the non-cloud model, Eq. 5.

By using the feedback control mechanism, the reciprocal of the calculated expected latencies of all nodes are used as the next round forwarding metrics.

We suppose there are 100 nodes in the network. We create a Markov generator A in a 100×100 matrix. The value of A_{ij} is a randomly generated real number in the interval $[1, 15]$, which represents the number of contacts between nodes i and j. Then we can numerically analyze the expected latency iteratively.

In Figs. 5 and 6, we show the latency and the number of forwardings in situations with 4 and 32 destinations, as well as in different cloud sizes. When the contact threshold is 0, the nodes are in the same cloud. The multicast scheme becomes a 2-hop routing transmission, where the source node will forward all copies to an encounter node that has a higher forwarding metric; then, this node will wait to forward the message to the destinations. Although, in this situation, the number of forwardings can be reduced, and the latency increases dramatically. Accompanied by the decreasing cloud size, the latency decreases and the number of forwardings increases. When the contact threshold reaches 16, which exceeds the maximum number of contacts between each node, no destination cloud exists. The message will be forwarded to an encounter node with a forwarding metric higher than the message holder until the destination is reached.

In both sparse (Fig. 5) and dense (Fig. 6) destination situations, the performance is initially imperfect due to high latency and a large number of forwardings. After the first round of multicast, the destinations will give the feedback information, with the feedback information as the forwarding metric, and the latency and the number of forwardings can be decreased dramatically. The latency decreases by about 33%, and the number of forwardings decreases by about 15% from round 1 to round 2. The latency decreases by about 21%, and the number of forwardings decreases by about 6% from round 2 to round 3. After about four rounds, the network performance converts to a stable situation. Therefore, we believe that our cloud-based multicast with feedback can improve the network performance.

VI. Simulation

We compare the performance of the proposed cloud-based multicast scheme with feedback to several state-of-the-art ones, including the delegation forwarding multicast scheme [9], spray-and-focus-based multicast scheme [33], and the epidemic-based multicast scheme [6], in Matlab using two real traces in the following categories:

1) *Delivery rate comparison*: we compare the delivery rate in real traces.

2) *Various cloud sizes comparison*: when the contact threshold (θ) is large, the destination cloud will disappear while the contact threshold becomes smaller, and the cloud size becomes larger. We compare the performance in different contact thresholds.

3) *Various proportions of destinations*: we compare the performances when the number of destinations varies.

4) *Speed of feedback control information spread*: we show how quickly the feedback control information can be spread

TABLE I
DELIVERY RATE IN *Intel* AND *Infocom 2006* TRACES.

Scheme	CM threshold 0	CM threshold 2	CM threshold 8	CM infinity threshold	DM	SM	EM
Intel	73.5%	74.3%	75.5%	74.1%	74.7%	74.3%	73.7%
Infocom	69.7%	71.8%	71.3%	70.7%	71.7%	70.9%	69.3%

to the nodes in the whole network, and the impact of feedback control information for multicast performance.

5) *Storage overhead comparison*: we compare the storage overhead of our proposed cloud-based multicast with feedback with other state-of-the-art DTN multicast schemes.

In the simulation, we consider two performance metrics:

1) *Latency*: the average delivery time for all of the delivered destinations to receive the message. The end-to-end delay is an important concern in cloud-based multicast with feedback design.

2) *Number of forwardings*: the average number of forwardings for all destinations to receive the message. This value represents the overhead in the network, in terms of how many times a message must be forwarded in order to reach all the destinations. The number of forwardings in the simulation measures the number of multicast messages exchanged, which does not include the metadata exchange.

A. Simulation Methods

We evaluate the performance of our proposed *cloud-based multicast scheme with feedback* (**CM**), which is compared with the following multicast schemes:

1) *Delegation forwarding multicast scheme* (**DM**) [9] : a message holder only replicates a copy of the message to a node with a higher forwarding metric for a destination, unless it is one of the destinations. Then the message holder also raises its own forwarding metric to the higher one it has successfully contacted.

2) *Spray-and-focus-based multicast scheme* (**SM**): it is an extension of spray-and-focus [33] in the multicast version. There are two phases in SM: *spray* and *focus*. In the spray phase, the message holder forwards half of the copies of the message to any node it meets without this message. After the spray phase, if any of the destinations have not received the message, it continues to the focus phase. In the focus phase, the message holder only forwards half of the copies of the message to a node with a higher forwarding metric for a destination, unless it is one of the destinations.

3) *Epidemic-based multicast scheme* (**EM**) [6]: the message holder will replicate a copy of the message to any node it meets without this message. It is a pure flooding-based multicast.

B. Simulation Settings

In the simulation, we compare the performances of the routing schemes in two real traces: Infocom 2006 [34] and Intel traces [35].

Intel trace includes Bluetooth sightings by groups of users carrying small devices (iMotes) for six days in the Intel Research Cambridge Computer Laboratory. There are $2,766$ contacts between 9 nodes over a period of $273,930$ time slots in seconds. In our simulation, we randomly set one of these 9 nodes as the source, and we choose other nodes as the destinations. The number of destinations is from 2 to 8.

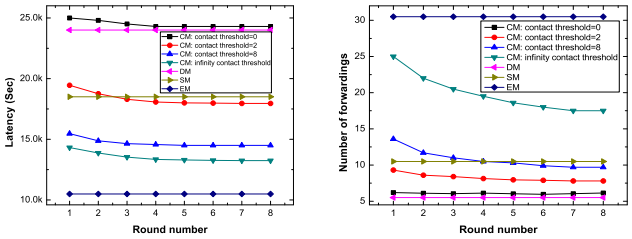


Fig. 7. Comparing the performance in the Intel trace in a variety of cloud sizes: (L): latency and (R): number of forwardings.

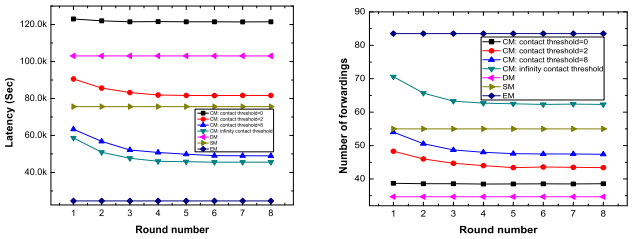


Fig. 8. Comparing the performance in the Infocom 2006 trace in a variety of cloud sizes: (L): latency and (R): number of forwardings.

Infocom 2006 trace includes the contacts between the iMote devices carried by participants for four days. 78 students and researchers carried the mobile iMotes to communicate in this trace. There are 20 stationary iMotes and several external devices. There are $227,657$ contacts between these participants over a period of $337,418$ time slots in seconds. In our simulation, we randomly select one participant with a mobile iMote as a source, and we choose other participants as the destinations. The number of destinations are set as $2, 4, 8, 16, 32$, and 64.

In the simulation, we generate 100 multicast messages in each round as to compare the performance in the simulation.

C. Simulation Results

1) Delivery rate comparison: in this part, we compare our proposed CM scheme with different contact thresholds. In Table I, we find that the delivery rate performs similarly in all compared schemes. Therefore, we only compare the latency and number of forwardings in the rest of the simulation.

2) Various cloud sizes comparison: in this part, we set the number of destinations to 4 in the Intel trace, and 32 in the Infocom 2006 trace. From Figs. 7 and 8, we find that our proposed cloud-based multicast with feedback routing scheme performs best among all of the schemes. Using the feedback control mechanism to update the forwarding metric in each round can improve the multicasting performance. After about five rounds, the two performance parameters stabilize to firm values.

In the Intel trace, by comparing the latency in Fig. 7, when we set the contact threshold to 0, the cloud-based multicast scheme with feedback has a longer latency than the delegation-based multicast scheme. When the contact threshold is 0, all nodes in the whole network are considered to be neighbors in one cloud. Therefore, the cloud-based multicast scheme with feedback is operating in an inefficient way. When we increase the contact threshold, the destination clouds become

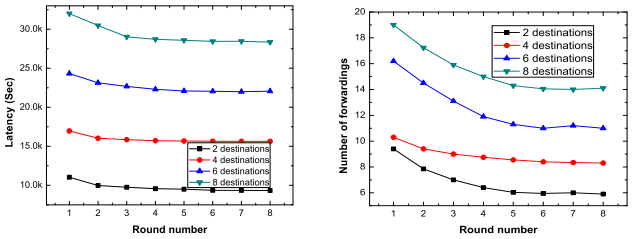


Fig. 9. Comparing the performance in the Intel trace in different numbers of destinations in each round: (L): latency and (R): number of forwardings.

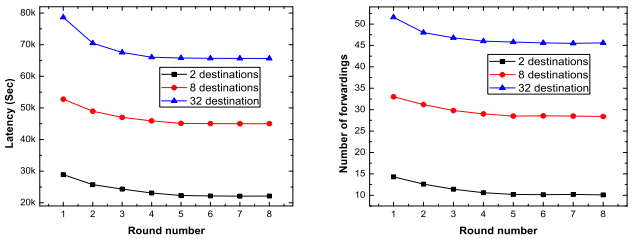


Fig. 10. Comparing the performance in the Infocom 2006 trace in different numbers of destinations in each round: (L): latency and (R): number of forwardings.

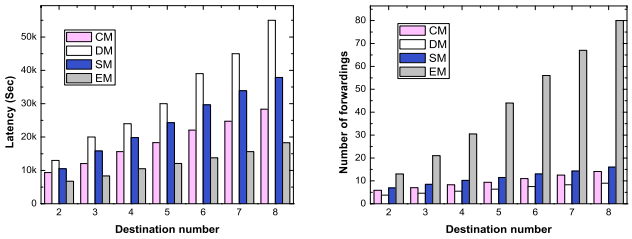


Fig. 11. Comparing the performance in the Intel trace in different numbers of destinations: (L): latency and (R): number of forwardings.

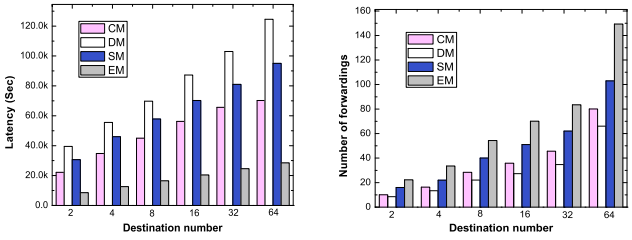


Fig. 12. Comparing the performance in the Infocom 2006 trace in different numbers of destinations: (L): latency and (R): number of forwardings.

smaller, and the end-to-end latency decreases dramatically. The latency in the 2-contact threshold case reduces by about 26% compared with the latency in the 0-contact threshold case. The 2-contact threshold case has a similar performance as the spray-and-focus-based multicast scheme. It further reduces by about 19.3% from the 2-contact threshold case to the 8-contact threshold case. When we set the contact threshold to infinity, which means that there is no destination cloud, the cloud-based multicast scheme with feedback will always work in the pre-cloud phase. By comparing the latency in this situation with the epidemic-based multicast scheme, we find that our proposed scheme only increases by about 26% in end-to-end latency. However, in Fig. 7, the epidemic-based multicast scheme has a much higher number of forwardings than the cloud-based multicast scheme with feedback in the infinity contact threshold case, by about 74%. When the contact threshold reduces, the average number of forwardings also reduces. The 8-contact threshold case has a similar number of forwardings as the spray-and-focus multicast scheme. When the contact threshold reduces to 0, the cloud-based multicast scheme with feedback has a similar number of forwardings as the delegation-based multicast scheme. The results from the Infocom trace show the same trend as in the Intel trace in Fig. 8.

In both the Intel and Infocom traces, we find that the overall latency decreases by about 5.6%, and the number of forwarding reduces by about 2.8% from round 1 to round 2. In round 3, the latency decreases by about 11.3%, and the number of forwardings reduces by about 5.7%, compared with round 2.

3) Various proportions of destinations: in this part, we set the contact threshold to 4 in both the Intel and Infocom 2006 traces. *In each round, there is a new multicast message generated. Therefore, each round is considered as an independent multicasting process.* From both the Intel and Infocom 2006 traces, we find that the multicast routing performance improves by applying the feedback control mechanism in cases with varying numbers of destinations. As shown in Fig. 9, in the Intel trace, round 1 can reduce the latency by about 5.2%, and reduce the number of forwardings by about 10.4% overall. After about five rounds, the performance metrics convert to stable values. The Infocom 2006 trace shows the same results as the Intel trace in Fig. 10. These results show the robustness of our proposed scheme.

From Figs. 11 and 12, we find that the cloud-based multicast scheme with feedback can decrease the latency dramatically, compared to the delegation-based multicast scheme; at the same time, the number of forwardings only increases slightly in both the Intel and Infocom traces. Our scheme can reduce the latency and number of forwardings compared with the spray-and-focus-based scheme. Compared with the epidemic-based scheme, our proposed cloud-based multicast scheme with feedback can reduce the number of forwardings dramatically.

In the Intel trace, it shows that the cloud-based multicast scheme with feedback can reduce the latency by about 38.7% and 23.2%, overall, compared with the delegation-based multicast scheme and spray-and-focus-based multicast scheme, respectively. In the number of forwardings comparison, the cloud-based multicast scheme with feedback reduces about 49% and 28.4% of the overhead overall, compared with the epidemic-based multicast scheme and spray-and-focus-based multicast scheme, respectively. The results in the Infocom trace show the same trend as the Intel trace.

4) Feedback information spread speed: in this part, we will show how fast the feedback information can be spread to the nodes in MSNs. Fig. 13 shows the percentage of nodes that received the feedback information in both the Intel and Infocom 2006 traces. In the Intel trace, it needs about 2,000 seconds to spread the information to the majority of nodes (95%). After 4,000 seconds, all of the nodes update their forwarding metrics. In the Infocom 2006 trace, it needs about 8,000 seconds to spread the information to all of the nodes.

5) Storage overhead comparison: in this part, we show the performance of the storage overhead in real traces. Here, we set the contact threshold to 4 in the cloud-based multicast scheme with feedback. As shown in Fig. 14, the x-axis is the

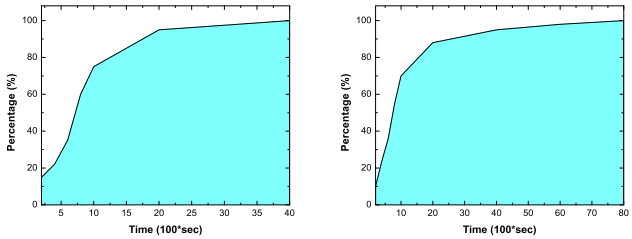


Fig. 13. Percentage of the nodes that received the feedback information: (L): Intel and (R): Infocom.

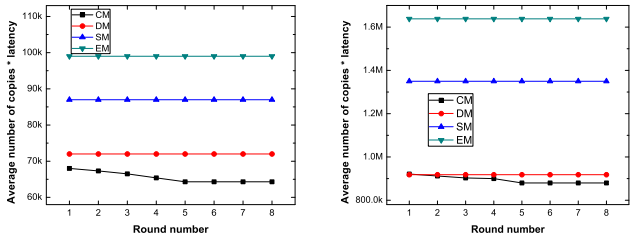


Fig. 14. Comparison of the storage overhead: (L): Intel and (R): Infocom.

round number, and the y-axis is the product of an average number of copies in the storage and delivery latency. We can see that in both Intel and Infocom 2006 traces, our proposed cloud-based multicast with feedback has the best performance in storage overhead comparison.

D. Summary of Simulation

Although the cloud-based multicast scheme with feedback increases the number of forwardings compared with the delegation-based multicast scheme, it significantly reduces the latency, especially when the number of destinations is large. It also reduces the number of forwardings and latency at the same time, compared with the spray-and-focus-based multicast scheme. Compared with the epidemic-based multicast scheme, the cloud-based multicast scheme significantly decreases the number of forwardings, while slightly increasing the latency. When the destination cloud size is large, the cloud-based multicast scheme with feedback performs with a long latency and a small number of forwardings. When we increase the contact threshold, the destination cloud size becomes small. It reduces the latency, while increasing the number of forwardings. The feedback control mechanism refines the forwarding metric and improves the performance step by step. The improvement is significant at the beginning, but then, the improvement reduces. After several rounds, the performance metrics will stay with stable values. Considering the characteristics of MSNs, we do not use broadcasting to spread the feedback information. Our feedback control mechanism can spread the information quickly by using the local information exchanging scheme only. The competitive performances in both the Intel and Infocom 2006 traces verify that the cloud-based multicast scheme with feedback performs effectively under different conditions.

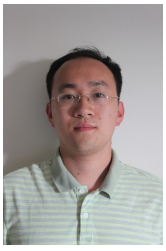
VII. Conclusion

In this paper, we proposed a cloud-based multicast scheme with feedback in mobile social networks. Our scheme has two parts: multicast process and feedback control process. First, we introduce a new concept: destination cloud. If the number of contacts, the nodes encountering the destination, is larger than a contact threshold, this node is called the *destination neighbor* in the destination cloud. In the multicast process, there are two phases: pre-cloud and inside-cloud. In the pre-cloud phase, the message holder will only forward the message to an encountered node with a larger forwarding metric. The forwarding metric updates each round by the feedback control mechanism. In the inside-cloud phase, the message holder will only forward the message to the destination directly. We formally analyzed the delivery latency by modeling our problem into a continuous time Markov chain model. Trace-driven simulation results showed that our proposed cloud-based multicast scheme with feedback performs better than the delegation-based, the spray-and-focus-based, and the epidemic-based multicast schemes. We believe that the destination cloud concept will play an important role in multicast routing in mobile social networks. The feedback control mechanism can improve the performance of MSN multicast. Our future work will focus on implementing real world cloud-based multicast with feedback applications in mobile social networks.

References

[1] Nielson, "More US Consumers Choosing Smartphones as Apple Closes the Gap on Android." http://blog.nielsen.com/nielsenwire/consumer/, 2012. [Online; accessed 28-February-2012].
[2] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in *Proc. 2005 ACM WDTN*, pp. 244–251.
[3] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: social-based forwarding in delay tolerant networks," in *Proc. 2008 ACM MobiHoc*, pp. 241–250.
[4] K. Fall, "A delay-tolerant network architecture for challenged Internets," in *Proc. 2003 ACM SIGCOMM*, pp. 27–34.
[5] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "Maxprop: routing for vehicle-based disruption-tolerant networks," in *Proc. 2006 IEEE INFOCOM*.
[6] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Technical Report, Dept. of Computer Science, Duke University, 2000.
[7] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *Proc. 2005 ACM WDTN*, pp. 252–259.
[8] J. Wu and Y. Wang, "A non-replication multicasting scheme in delay tolerant networks," in *Proc. 2010 IEEE MASS*, pp. 89–98.
[9] Y. Wang, X. Li, and J. Wu, "Multicasting in delay tolerant networks: delegation forwarding," in *Proc. 2010 IEEE GLOBECOM*.
[10] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *IEEE Trans. Mobile Comput.*, vol. 6, pp. 606–620, 2007.
[11] J. Wu and Y. Wang, "Social feature-based multi-path routing in delay tolerant networks," in *Proc. 2012 IEEE Infocom*.
[12] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: homophily in social networks," *Annual Review of Sociology*, vol. 27, pp. 415–444, 2001.
[13] A. Balasubramanian, B. Levine, and A. Venkataramani, "DTN routing as a resource allocation problem," in *Proc. 2007 ACM SIGCOMM*, pp. 373–384.
[14] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 7, 2003.
[15] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot, "Delegation forwarding," in *Proc. 2008 ACM MobiHoc*, pp. 251–260.
[16] W. Zhao, M. Ammar, and E. Zegura, "Multicasting in delay tolerant networks: semantic models and routing algorithms," in *Proc. 2005 ACM WDTN*, pp. 268–275.
[17] P. Yang and M. Chuah, "Efficient interdomain multicast delivery in disruption tolerant networks," in *Proc. 2008 IEEE Mobile Ad-hoc and Sensor Networks*, pp. 81–88.
[18] S.-C. Lo and N.-W. Luo, "Quota-based multicast routing in delay-tolerant networks," in *Proc. 2012 International Symposium on Wireless Personal Multimedia Communications*, pp. 544–548.

[19] A. Palma, P. Pereira, A. Casaca, and null, "Multicast routing protocol for vehicular delay-tolerant networks," in *Proc. 2012 IEEE International Conference onWireless and Mobile Computing, Networking and Communications*, pp. 753–760.
[20] W. Gao, Q. Li, B. Zhao, and G. Cao, "Multicasting in delay tolerant networks: a social network perspective," in *Proc. 2009 ACM MobiHoc*.
[21] M. Mongiovi, A. Singh, X. Yan, B. Zong, and K. Psounis, "Efficient multicasting for delay tolerant networks using graph indexing," in *Proc. 2012 IEEE INFOCOM*, pp. 1386–1394.
[22] S.-C. Lo and W.-R. Liou, "Dynamic quota-based routing in delay-tolerant networks," in *Proc. 2012 IEEE Vehicular Technology Conference*, pp. 1–5.
[23] I. Stojmenovic, "Position-based routing in ad hoc networks," *IEEE Commun. Mag.*, vol. 40, pp. 128–134, July 2002.
[24] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell, "Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application," in *Proc. 2008 ACM SenSys*, pp. 337–350.
[25] E. Bulut and B. Szymanski, "Friendship based routing in delay tolerant mobile social networks," in *Proc. 2010 IEEE GLOBECOM*.
[26] B. Han, P. Hui, V. A. Kumar, M. V. Marathe, G. Pei, and A. Srinivasan, "Cellular traffic offloading through opportunistic communications: a case study," in *Proc. 2010 ACM CHANTS*, pp. 31–38.
[27] E. P. C. Jones, L. Li, and P. A. S. Ward, "Practical routing in delay-tolerant networks," in *Proc. 2005 ACM WDTN*, pp. 237–243.
[28] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *Proc. 2004 ACM SIGCOMM*, pp. 145–158.
[29] S. M. Ross, *Stochastic Processes*, 2nd ed. Wiley, 1995.
[30] V. Conan, J. Leguay, and T. Friedman, "Characterizing pairwise inter-contact patterns in delay tolerant networks," in *Proc. 2007 ICST International Conference on Autonomic Computing and Communication Systems*.
[31] R. Groenevelt, P. Nain, and G. Koole, "Message delay in manet," in *Proc. 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*.
[32] M. Grossglauser and D. Tse, "Mobility increases the capacity of ad hoc wireless networks," *IEEE/ACM Trans. Netw.*, vol. 10, no. 4, pp. 477–486, 2002.
[33] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and focus: efficient mobility-assisted routing for heterogeneous and correlated mobility," in *Proc. 2007 IEEE PERCOMW*.
[34] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD trace cambridge/haggle/imote/infocom2006 (v. 2009-05-29)," May 2009.
[35] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD trace cambridge/haggle/imote/intel (v. 2006-01-31)," Jan. 2006.



Yunsheng Wang received B.Eng. in Electronic and Information Engineering from Dalian University of Technology, Dalian, China, in 2007; the M.Res. in Telecommunication from University College London, London, UK, in 2008; the Ph.D. from Department of Computer and Information Sciences, Temple University, Philadelphia, PA, US, in 2013. He is an Assistant Professor in the Department of Computer Science in Kettering University, Flint, MI, US. His research interests include various topics in the application and protocols of wireless networks. His current research focuses on the efficient communication in delay tolerant networks and mobile opportunistic social networks.

Jie Wu (F'09) is the chair and a Laura H. Carnell Professor in the Department of Computer and Information Sciences at Temple University. Prior to joining Temple University, he was a program director at the National Science Foundation and Distinguished Professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. Dr. Wu regularly published in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON SERVICE COMPUTING, and the *Journal of Parallel and Distributed Computing*. Dr. Wu was general co-chair/chair for IEEE MASS 2006 and IEEE IPDPS 2008 and program co-chair for IEEE INFOCOM 2011. Currently, he is serving as general chair for IEEE ICDCS 2013 and ACM MobiHoc 2014, and program chair for CCF CNCC 2013. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a CCF Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.

Wei-Shih Yang received the B.S. degree in mathematics from National Taiwan University, in 1976, and the M.A. and Ph.D. degree in mathematics from Cornell University, in 1981 and 1984, respectively. He is a professor in the Department of Mathematics at Temple University, Philadelphia, PA. His research interests are in probability theory, mathematical physics and quantum computing. He has worked on phase transitions and critical phenomena in statistical mechanics, Gaussian and non-Gaussian random fields with applications to quantum field theory, percolation, Ising model, and self-avoiding random walks. His most recent works include the following topics: quantum random walks with applications to quantum computing and ruin probability of risk processes.