PAPER

CROP: Community-Relevance-Based Opportunistic Routing in Delay Tolerant Networks

Je-Wei CHANG[†] *and* **Chien CHEN**[††a)]**,** *Nonmembers*

SUMMARY Researchers have developed several social-based routing protocols for delay tolerant networks (DTNs) over the past few years. Two main routing metrics to support social-based routing in DTNs are centrality and similarity metrics. These two metrics help packets decide how to travel through the network to achieve short delay or low drop rate. This study presents a new routing scheme called Community-Relevance based Opportunistic routing (CROP). CROP uses a different message forwarding approach in DTNs by combining community structure with a new centrality metric called *community relevance*. One fundamental change in this approach is that community relevance values do not represent the importance of communities themselves. Instead, they are computed for each community-community relationship individually, which means that the level of importance of one community depends on the packet's destination community. The study also compares CROP with other routing algorithms such as BubbleRap and SimBet. Simulation results show that CROP achieves an average delivery ratio improvement of at least 30% and can distribute packets more fairly within the network.

key words: *delay tolerant network, opportunistic routing, social networks*

1. Introduction

Researchers have long studied routing protocols in Delay Tolerant Networks (DTNs) [1] in an effort to cope with the intermittent connectivity of conventional networks. To facilitate data delivery in a fragmented network, nodes may have to temporarily carry data packets while they move, and then forward the packets to other nodes in an opportunistic hop-by-hop manner.

In such opportunistic networks, a node must decide whether to forward packets to an encountered node. A node typically makes forwarding decisions by evaluating extra information such as contact history [2]–[7], mobility pattern information [8]–[12], or information on social interaction [13]–[22]. Research that computes the delivery metric based on social interaction between nodes in DTNs has become popular in recent years. This approach constructs a social graph based on the contacts of all nodes. Complex network analysis over the social graph [19], [23] reveals the importance each node plays in routing in the network.

Most social-based routing schemes use following two metrics to facilitate message delivery: *centrality* and *similarity*. The centrality metric focuses on how often nodes function as bridges for message delivery. The similarity metric determines if a node is in the neighborhood of the destination node. When two nodes encounter each other, a packet resident in one node is preferably forwarded to the other node if it has greater similarity to the destination node of that packet. If both nodes have the same similarity to a packet, the packet is forwarded to the node with greater centrality (i.e. greater chance to reach the destination). PeopleRank [21] computes the centrality metric, that is the importance of a node in the social graph, in an approach similar to the idea of PageRank in Google Search. SimBet [13] computes *betweeness centrality* by counting how often a node is part of the shortest path between any two nodes, and computes similarity by counting how often destination nodes appear within a two-hop range of a node. BubbleRap [20] uses *degree centrality*, where a node's degree is the number of its direct neighbors and computes similarity by detecting each node's community affiliation.

Community based routing has become more and more popular in recent years. The nature of community structures makes it possible to predict neighborhood similarity between two nodes. However, previous studies have seldom discussed a centrality metric at the community level. Some social communities already exist such as different companies in an office building or different laboratories on a campus. The degree of social interactions between different communities are different. For instance, two labs in the same department on a campus may have a higher social interaction with each other than two labs in different departments. Thus, social interaction between communities is a good basis for the centrality metric design.

In addition, all previous methods forward packets based on a single centrality metric of each node without regard to packet destination. These approaches cause the nodes with higher centrality metric to have a higher probability to be asked to forward packets for other nodes. The invariant centrality metric leads the traffic in social-based routing to concentrate on a small number of high centrality nodes [18]. Under realistic conditions, the optimal routing path of a packet should vary depending on its source and destination nodes. Therefore, the centrality of a node should vary depending on its packets' destinations. Thus, our centrality metric design consider the packets' destinations.

The goal of this study is to develop a routing algorithm based on the community-relevance metric between any two communities in a DTN [24]. The proposed routing scheme [24] assumes that (1) all of the communities

Manuscript received January 13, 2014.
Manuscript revised May 8, 2014.

[†]The author is with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C.
[††]The author is with Information Technology Service Center, National Chiao Tung University, Hsinchu, Taiwan, R.O.C.
a) E-mail: chienchen@cs.nctu.edu.tw

DOI: 10.1587/transcom.E97.B.1875

are known in advance, (2) which node belongs to which community is also known in advance, and (3) a node can be part of multiple communities. These assumptions are feasible since there are many such existing communities in human society such as multiple clubs in a school, multiple branch offices in a company, etc. [25]. A community transition graph is first transformed from the contact graph. The community relevance $R(c, b)$ between two communities c and b is the inverse of the expected random walk distance from c to b in the community transition graph. Although the community relevance is computed based on the community transition graph, this computation can be done in distributed way (i.e. without full knowledge of the community transition graph). This study also proposes a routing scheme, called Community-Relevance based Opportunistic routing (CROP). When a node encounters another node, a packet is forwarded to the node with a higher community relevance to the destination community of that packet. If two nodes have the same community relevance, the packet is forwarded to the node with a higher degree (number of neighbors). Since a community may have a higher relevance to one community but a lower relevance to another community, CROP can provide different centrality values for different packets destined to different communities. Therefore, by using the community-relevance metric, it is possible to distribute traffic more fairly within the network and avoid the traffic being concentrated on a small number of high centrality nodes. In addition, the node with a larger community relevance to a destination community has a shorter expected random walk distance to that community. Therefore, it is more likely in the neighborhood of that destination community. Thus, our community-relevance metric conceptualizes the similarity metric and the centrality metric into one metric design.

This study further compares CROP with another community-based opportunistic routing algorithm (BubbleRap) in terms of (maximum) throughput theoretically. The maximum throughput is formulated as a modified maximum multi-commodity flow problem by adapting the different flow constraints for the two routing schemes. Results indicate that CROP achieves a 50% throughput improvement compared to BubbleRap. A series of proofs show that at least one routing path exists between any two communities in CROP, which may not be true in BubbleRap. Simulation results show that CROP offers an average delivery ratio improvement of 30% compared to other social-based routing schemes. The simulation results further indicate that CROP distributes traffic load fairly within the network, leading to smaller packet dropping rates under the limited buffer scenario.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 introduces the network model. Section 4 presents details of centralized and distributed community-relevance metric computation, and the CROP routing scheme. We analyze the maximum throughputs of the two routing protocols, BubbleRap and CROP in Sect. 5. Section 6 discusses simulation results, and Sect. 7 concludes the paper.

2. Related Work

Many routing protocols attempt to enhance the packet delivery ratio in DTNs. These protocols generally fall into one of the three categories: message ferrying routing, deterministic routing (space-time routing), or opportunistic routing schemes.

Message ferrying routing schemes use several extra mobile nodes, called message ferries (or data mules), to ferry data in a network. The message ferry receives packets from the source node as soon as it comes close to the source node, and disseminates packets to the destination nodes when it is near the destination nodes. The message ferrying scheme plans the trajectory of the ferry route to visit all the nodes in the DTN, and minimizes the message ferry's total travel distance by solving the travelling salesman problem (TSP) [8]–[11].

Deterministic routing schemes assume that each node has a varying level of prior knowledge of the network (e.g., contact time of two nodes, node queue size, available bandwidth of a connection, etc). The authors of [26] designed different routing protocols that depend on different types of given prior knowledge that satisfy different QoS-level constraints. In [27], the authors constructed a space-time graph from nodes' contacts. A shortest path in the graph can be discovered using Dijkstra's shortest path algorithm.

Opportunistic routing schemes prioritize the packets in the buffer to decide which packets should be forwarded to the encounter nodes and which packets should be dropped in case of a buffer overflow. Most schemes use a delivery metric between source and destination under different criteria to evaluate the importance of packets in the network. These schemes can be further categorized as prediction-based and social-based schemes.

Prediction-based schemes compute a delivery metric to forecast the future contact probability between two nodes. The authors of [4] analyzed the path strength between source and destination within k hops. The cost of each hop (i, j) is $1 - f_j^i$, where f_j^i is the contact probability between nodes i and j. The authors of [6] used a probabilistic space-time graph based on a node's mobility pattern. The frequency of the nodes' visits at each possible position is the basis of the future distance calculation in the Euclidean space. The authors of [17] computed a delivery probability $\Pr(i, d, r, h)$ in which node i is the sender node, node d is the destination node, r is the residual time of the packet, and h is the remaining hop counts of the packet. When node i meets node j, node i forwards a packet to node j if $\Pr(i, d, r-1, h) < \Pr(i, d, r-1, h-1) * \Pr(j, d, r-1, h-1)$. In [28], they considered the stochastic future contact time distribution between nodes and formulate it as a time-homogeneous semi-Markov chain. A node can use contact time distribution to find the best forwarding time of a packet to another node.

Social-based schemes consider that social relationships

affect human mobility. Complex network analysis [19], [29] over the social graph can reveal the importance of a node as a gateway node for data dissemination. Social-based schemes usually work with two delivery metrics: the centrality metric and the similarity metric. The centrality metric focuses on the ability of a node to act as a bridge node in the network. The similarity metric determines whether the encounter is within the neighborhood of a destination node. For example, SimBet [13] uses both neighborhood similarity and betweeness centrality. SimBet uses neighborhood similarity to detect encounter nodes that are within two hops of the destination node. Betweeness centrality is used to check if an encounter node is a bridge node. In an encounter between two nodes, the packet is forwarded to the node with a higher similarity to the destination node. Otherwise, the packet is temporarily stored at the node with the highest centrality metric. SimBetAge [14] exhibits the same concept of SimBet, but includes aging of contacts in social graphs to achieve a more accurate delivery metric. BubbleRap [20] is the first algorithm to include the community concepts. A packet is forwarded to nodes with high centrality values until it reaches the destination community. The centrality of a node in BubbleRap is the number of the node's neighbors (also called *degree*). Once the packet reaches the destination community, the packet is only forwarded to nodes within that community with high local centrality. LocalCom [15] achieves distributed community detection by considering k-hop information. LocalCom uses a restricted flooding mechanism to decrease the number of transmissions in a DTN. A packet is first forwarded to a gateway node via source routing. When a packet reaches a gateway node between two or more communities, that gateway node delivers one copy of the packet to each neighbor community, and keeps the original packet. PeopleRank [21] adopts the concept of Google's PageRank to calculate the importance of a node in social networks. When a node encounters another node, the packet is forwarded to the encountered node if it has a higher importance than the current node. The authors of [18] showed that packets in social based routing schemes are usually unfairly concentrated on a small number of nodes because of the invariant centrality metric. They designed a delivery metric that considers both the buffer utilization and the centrality of the encounter node to eliminate unfair packet distribution in social based routing. Other researchers [19] showed that the number of contacts used in the creation of the social graph has a significant effect on the accuracy of the metric. Therefore, they conceptualized a density-based contact collection method to identify the optimal number of contacts to collect for different DTNs.

This study proposes a social based routing algorithm which is different from other social based routings in the following aspects. Firstly, our approach computes the community relevance between any two communities instead of the importance of a community itself. Secondly, most of the previous studies consider the centrality metric and the similarity metric, respectively, whereas our community-relevance metric conceptualizes the similarity metric and the centrality metric into single metric design. Finally, most of the previous social routings consider an invariant centrality metric, but our community-relevance metric provides different centrality values for different destination communities.

3. Network Model

Consider a mobile network consisting of n mobile nodes represented by the set $V = \{v_0, v_1, \ldots, v_{n-1}\}$ and k communities represented by the set $C = \{c_0, c_1, \ldots, c_{k-1}\}$. Each node can belong to one or more communities. Since a node can belong to multiple communities, we define $C(i)$ as a set of communities that node i belongs to. Each node knows the community set it belongs to in advance. This study uses $V(c) = \{m \mid m \in V, c \in C(m)\}$ to represent all the nodes in community c. A directed community graph $G = \langle C, E\rangle$ is constructed to compute the community relevance between two communities, where C is the community set and E is the edge set of the community graph. There is an edge $e = (a, b)$ between communities a and b in a community graph (for $a \neq b$) if and only if there exists at least one node in $V(a)$ that is in contact with a node in $V(b)$. The term $N(a) = \{b \mid (a, b) \in E\}$ stands for the neighbor communities of community a. For instance, Fig. 1 is an example of the community graph $G = \langle C, E\rangle$. We have $C = \{c_0, c_1, \ldots, c_4\}$, E is all of the edges in Fig. 1, and $N(c_0) = \{c_1, c_2, c_4\}$. A community graph can be constructed by aggregating contacts within a social network. To reflect the underlying social or mobility structure of the network, not all contacts within a DTN should be used for the community graph. A sliding-window aggregation method as in [20] can be used to aggregate contacts to form a community graph. Only contacts within the last periods of time T, whose contact time is greater than t are considered valid, where T and t are empirically determined. Here, the term γ represents all those valid contacts. The term "contact" in the following is referred to as the valid contact if not specified otherwise. Since a node may belong to multiple communities, a contact between two nodes may cause multiple contacts in the community graph. Specifically, node i in contact with node j generates contacts $\eta_{ij} = \{(a, b) \mid a \in C(i), b \in C(j), a \neq b\}$ in the community graph. An indicator $I_{ij}(a, b)$ is used to show if nodes i and j can generate a contact between communities a and b in the community graph. Hence, $I_{ij}(a, b) = 1$ for all contacts $(a, b) \in \eta_{ij}$ and $I_{ij}(a, b) = 0$ otherwise. The example in Fig. 2 shows that node i belonging to community set $C(i) = \{c_1, c_2\}$, has a contact with node j of community set $C(j) = \{c_2, c_3\}$. From the definition of η_{ij}, contacts (c_1, c_2), (c_1, c_3), and (c_2, c_3) are generated in the community graph. In this example, we have $I_{ij}(c_1, c_2) = I_{ij}(c_1, c_3) = I_{ij}(c_2, c_3) = 1$. A weight $w(a, b)$ is assigned to each edge $(a, b) \in E$, where $w(a, b)$ is the number of contacts between communities a and b. The weight $w(a, b)$ can be calculated easily using a cost function $w(a, b) = \sum_{(i,j)\in\gamma} I_{ij}(a, b)$. The goal of this study is to compute the community relevance using the Markov stochastic process. Thus, this study further transforms the

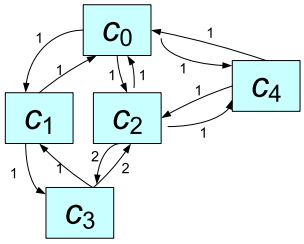


Fig. 1 Community graph.

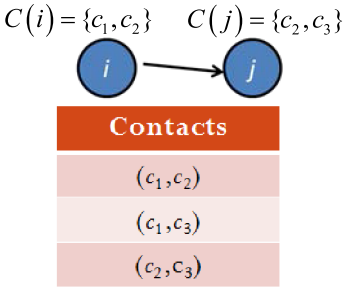


Fig. 2 Contacts between two communities.

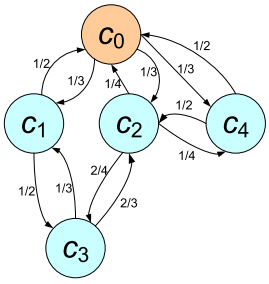


Fig. 3 Community transition graph.

community graph G into a community transition graph G'. In this case, G and G' are isomorphic. The only difference between the two graphs is the edge weight. More precisely, the edge weight in G' is based on a transition probability between two communities. The community transition probability $p(a,b)$ from community a to community b is $w(a,b)/\sum_{(a,v)\in E} w(a,v)$, which is the ratio of the number of contacts from community a to community b to the total number of all contacts of community a with other communities. Figure 3 shows the community transition graph transformed from the community graph in Fig. 1. For example, if we want to compute the transition probability $p(c_2,c_0)$, the number of generated contacts of community c_2 with the communities c_0, c_3, and c_4, are 1, 2, and 1. Based on the above, we can compute $p(c_2,c_0)=\frac{1}{(1+2+1)}=\frac{1}{4}$.

4. Community-Relevance-Based Opportunistic Routing

This section introduces a community-relevance based routing method. Section 4.1 describes a centralized community-relevance metric computation over a community transition graph. Section 4.2 proposes a routing protocol based on the community-relevance metric called CROP. Section 4.3 introduces a distributed algorithm to obtain the community-relevance metric.

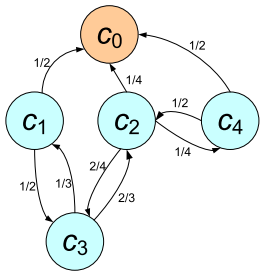


Fig. 4 An example of $G'(C_0)$.

4.1 Community Relevance Computation

For a packet in community a destines to community b, this study defines the community relevance metric $R(a,b)$ between communities a and b as the inverse of the expected *random walk hop distance* from a to b, denoted as $h(a,b)$, over community transition graph G'. When a packet reaches the destination community b, it does not need to random walk to other communities from community b. Therefore, this study removes all of the outgoing edges $\{(b,c)\,|c\in N(b)\}$ of destination community b from G'. Thus, the term $G'(b)$ represents destination community transition graph of community b which is the sub-graph of G' after removing edges $\{(b,c)\,|c\in N(b)\}$. This study computes the community relevance $R(a,b)$ for any destination community $b\in C$ over $G'(b)$. Figure 4 demonstrates $G'(c_0)$, which can be derived from the destination community transition graph of community c_0 in Fig. 3 by removing all of outgoing edges of community c_0.

For a packet in community a whose destination community is b, suppose that community a chooses one of its neighboring community c as the next hop. Then, the expected random walk hop distance $h(a,b)$ from a to b through c is equal to $h(c,b)+1$. Therefore, we can calculate $h(a,b)$ by the following recurrence:

$$h(a,b)=\begin{cases}\sum\limits_{c\in N(a)} p(a,c)\cdot(h(c,b)+1) & \text{if } a\neq b\\ 0 & \text{if } a=b\end{cases} \quad (1)$$

where $p(a,c)$ is the transition probability from communities a to c, and $N(a)$ is the neighbor communities of community a. We can further derive:

$$h(a,b)=\left(\sum_{c\in N(a)} p(a,c)\cdot(h(c,b))\right)+1.$$

In Eq. (1), $h(a,b)$ can be derived by solving the linear equations as follows. All the expected random walk hop distances from any community to the destination community b can be described by a vector $H(b)=[h_0(b)\ h_1(b)\cdots h_a(b)\ \cdots\ h_{k-1}(b)]^T_{k\times 1}$, where the value of

$h_a(b)$ represents the result of $h(a,b)$. The term $\Pi(b) = \left(\pi_{ij}(b)\right)_{k\times k}$ denotes a $k \times k$ community transition probability matrix in $G'(b)$. The value $\pi_{ij}(b)$ is assigned as follows:

$$\pi_{ij}(b) = \begin{cases} p(i,j) & \text{if edge } (i,j) \text{ exists in } G'(b) \\ 0 & \text{otherwise} \end{cases}, \quad (2)$$

where $p(i,j)$ is the transition probability from communitiy i to community j. The distance vector can be defined as $D(b) = [d_0(b)\ d_1(b) \cdots d_a(b) \cdots d_{k-1}(b)]^T_{k\times 1}$, where $d_a(b) = 0$ for $a = b$ and $d_a(b) = 1$ for $a \neq b$. Equation (1) can be transformed into the following matrix equation:

$$H(b) = (\Pi(b) \times H(b)) + D(b) \quad (3)$$

It is then easy to solve $H(b)$ in the following equation;

$$H(b) = (I - \Pi(b))^{-1} \times D(b) \quad (4)$$

where I is a $k\times k$ identity matrix. The relevance $R(a,b)$ from communities a to b is obtained by $(h(a,b))^{-1}$ by solving $H(b)$. For instance, the transition probability matrix $\Pi(c_0)$ of $G'(c_0)$ and distance vector $D(c_0)$ in Fig. 4 are:

$$\Pi(c_0) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{4} & 0 & 0 & \frac{1}{2} & \frac{1}{4} \\ 0 & \frac{1}{3} & \frac{2}{3} & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \end{bmatrix} \quad \text{and } D(c_0) = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \end{bmatrix}^T$$

Then $H(c_0) = [0\ \ 3.56\ \ 4.32\ \ 5.06\ \ 3.16]^T$ can be derived by solving Eq. (4).

To compute all community-relevance values between any two communities, this study iteratively selects a destination community $b \in C$ and calculates the community relevance from all communities to selected destination community b. It is then possible to deduce the community relevance between any two communities.

4.2 CROP Routing Protocol

This study proposes a community-relevance based opportunistic routing protocol called CROP. The CROP is a single-copy routing protocol such that only one copy of a packet is forwarded within the network during routing process. The reason of choosing single-copy scheme is because the buffer space on a node in DTN is precious. We do not want to waste extra space for multiple copies of a packet. CROP consists of two parts: 1) a forwarding scheme and 2) a buffer management scheme. These two schemes are both based on the community-relevance metric.

1) Forwarding scheme: The CROP forwarding scheme is designed based on the community-relevance metric. Each node i assigns a relevance cost, $R_i(\rho) = \max_{a\in C(i), b\in C(d)} (R(a,b))$ to a packet ρ whose destination is node d, where $C(i)$ and $C(d)$ are the community sets that nodes i and d belong to, respectively. Since each node in CROP may belong to several communities, $R_i(\rho)$ represents the highest community relevance among all possible community relevances between the communities of node i (i.e., $C(i)$) and the communities of destination node d of packet ρ (i.e., $C(d)$). Node i then decides whether to send the packets ρ from its buffer to node j according to the following rules.

- **Case 1**: node i immediately sends all of the packets whose destination is node j in its buffer to node j. Then, for the remaining packet ρ, we have:
- **Case 2** $\left(R_i(\rho) < R_j(\rho)\right)$: node j has a higher relevance to packet ρ than node i. Therefore, node i sends packet ρ to node j.
- **Case 3** $\left(R_i(\rho) = R_j(\rho)\right)$: nodes i and j have the same relevance to packet ρ. In this case, the number of neighbors of node i and node j are compared. If the degree of node j is larger than node i, then node i sends packet ρ to node j. Otherwise, ρ remains in the buffer of node i. This study defines a node u as a neighbor of node i if the contact time between node u and node i is larger than t within last period time T [20].
- **Case 4** $\left(R_i(\rho) > R_j(\rho)\right)$: because node i has a larger relevance to packet ρ, it keeps packet ρ in its buffer.

This study supposes that node i can exchange all the packets in its buffer with node j when nodes i and j encounter one another. Since the buffer space of each node is a restricted resource, a packet received from another node may cause buffer overflow. Hence, CROP also considers a simple buffer management scheme to control buffer overflow.

2) Buffer Management: The buffer management scheme is also designed based on the community-relevance metric. When a buffer overflow occurs in node i, the packet ρ with the smallest $R_i(\rho)$ is removed from i's buffer until the buffer overflow disappear.

4.3 Distributed Community Relevance Computation

Since Sect. 4.1 computes community relevance in a centralized manner, this subsection describes a distributed cluster based scheme to deduce community relevance using Eq. (1). A cluster head CH_a of community a is responsible for computing the estimated community-relevance vector $R'(a) = \left[r'_0(a)\ \ r'_1(a) \ldots r'_b(a)\ \ \ldots\ \ r'_{k-1}(a)\right]_{1\times k}$, where $r'_b(a)$ is the estimated community relevance from community a to community b. There are some cluster head selection algorithms for DTNs [15], but the methods of selecting the best cluster heads in DTNs are beyond the scope of this study.

In the beginning of the algorithm, all values of $r'_b(a) = 1$ for $a \neq b$ and $r'_b(a) = \infty$ for $a = b$. Based on Eq. (1), the following equation can be obtained:

$$R'(a) = \left(\left(\sum_{c\in N(a)} p'(a,c) \cdot (R'(c))^{-1}\right) + D(a)\right)^{-1}, \quad (5)$$

where $p'(a,c)$ is the approximate transition probabilities between community a and its neighbor community c. $X^{-1} = \left[(x_1)^{-1}(x_2)^{-1} \ldots (x_k)^{-1}\right]$ is the inverse function for vector $X = [x_1 x_2 \ldots x_k]$, and $D(a) = [d_0(a)\, d_1(a) \cdots d_b(a) \cdots d_{k-1}(a)]_{1\times k}$ is the distance vector,

where $d_b(a) = 0$ if $a = b$, otherwise $d_b(a) = 1$. From Eq. (5), CH_a can compute its community-relevance vector based on the local information of its neighbor communities only (i.e., without full knowledge of the community transition graph). Specifically, CH_a needs further obtain 1) the approximate transition probabilities of community a to all its neighbor communities and 2) the estimated community-relevance vectors of all its neighbor communities. CROP attaches these two information to the community contact messages.

Community contact messages are generated to obtain community contact information. Each community contact message m has a 4-tuple (src_community, dst_community, dst_relevance, TimeStamp). When node i has a contact with node j with a contact time larger than a constant t within the last time period T, node i generates a set of community contact messages M_{ij} at current time ψ:

$$M_{ij} = \left\{(u, v, R'(b), \psi) \mid (u, v) \in \eta_{ij}\right\},$$

where $\eta_{ij} = \{(u, v) \mid u \in C(i),\ v \in C(j),\ u \neq v\}$. Note that since node i and node j could belong to multiple communities, multiple contact messages could be generated by a contact between nodes i and j. When two nodes encounter one another, they exchange the contact messages stored in their buffers. For every T time slots, cluster head CH_a computes approximately $p'(a, c) = \omega/\Omega$ to compute $R'(a)$ based on community contact messages χ it received. Where ω is the number of the contact messages in χ which are from communities a to c and Ω is the number of the contact messages in χ whose src_community is community a. After computing $p'(a, c)$, cluster head CH_a updates the latest estimated community-relevance vector $R'_{new}(a)$ based on the previous estimated community-relevance vector $R'_{\text{old}}(a)$ and the currently received contact messages by the following weight function:

$$R'_{\text{new}}(a) = \beta \cdot R'_{\text{old}}(a) + (1-\beta) \cdot \left(\left(\sum_{c \in N'(a)} p'(a, c) \cdot (R'(c))^{-1}\right) + D(a)\right)^{-1} \quad (6)$$

where β is the weight that user predefined parameter (we use $\beta = 0.5$ in our experiments), $R'(c)$ is the latest estimated community-relevance vector of community c attached to the contact messages. Then, the latest $R'_{\text{new}}(a)$ is sent through the DTN. Upon receiving the $R'_{\text{new}}(a)$, a node belonging to community a updates its local relevance vector $R'(a)$ to become the latest $R'_{\text{new}}(a)$. To let a contact message have an opportunity to be computed at least once by cluster heads, all contact messages in M_{ij} generated at time ψ must expire at time $\psi + T$. To avoid generating too many community contact messages, the time interval to generate two consecutive community contact message sets for the contact between nodes i and j must exceed Δ time slots. Specially, if M_{ij} is generated at time ψ, then the next time to generate M_{ij} must be after $\psi + \Delta$.

When nodes i and j encounter each other, they exchange their community-relevance vectors. Based on the received relevance vectors from node j, node i can compute the relevance costs for all of the packets in its buffer to node j. Node i then sends the packets in its buffer to node j, when node j has higher relevance costs to those packets than node i. In the same way, node j sends the packets in its buffer to node i, when node i has higher relevance costs to those packets.

5. Protocol Analysis

This section analyzes the maximum throughputs of the two routing protocols, BubbleRap and CROP. We formulate the maximum throughput as the maximum multi-commodity over a contact graph. To compute the maximum multi-commodity flow for both protocols, we first transform a contact graph into the corresponding color graphs for both BubbleRap and CROP. Section 5.1 introduces the construction of color graphs. Each community is assigned a unique color and each node in the same community has the same color in the color graph. Additionally, each edge (i, j) contains a color attribute set to represent which destination communities of packets can be forwarded from nodes i to j based on the BubbleRap and CROP forwarding rules, respectively. Section 5.2 describes the modified maximum multi-commodity flows over the color graphs. Section 5.3 proves that there always exists a routing path between any two communities in CROP, but not the case in BubbleRap.

5.1 Color Graph

A contact graph can be generated by accumulating all contacts between any two nodes over time. Since this graph does not consider the time sequences of contacts, a routing path between a source and a destination in the contact graph may not exist under realistic conditions. Hence, the proposed analysis is an upper bound of the maximum throughput within a period of time.

To simplify the analysis, we assume that each node in the DTN belongs to a single community. It is easy to apply the proposed analysis scheme to a multiple-community case. To measure the maximum throughputs of BubbleRap and CROP, the contact graph is first transformed to the corresponding color graph. The term $G_c = \langle V, E' \rangle$ represents a color graph, where V is all of the nodes in the network and E' is the directed edge set. Each node i in G_c is assigned a color C_i to represent the community to which node i belongs. Each edge (i, j) in G_c is assigned a color set Λ_{ij}. The term Λ_{ij} indicates which communities of destination nodes of the packets can be forwarded from nodes i to j based on different forwarding rules of routing protocols. For example in Fig. 5, packets whose destination node belongs to one

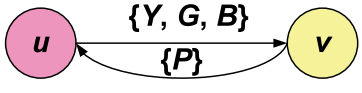


Fig. 5 Color graph.

of the communities Y, G, or B are allowed to be forwarded via edge (u, v), and packets belonging to destination node in community P are forwarded via edge (v, u). Both BubbleRap and CROP have their own color graphs. Based on their routing principles, the color set Λ_{ij} on each edge (i, j) can be assigned in advance.

Assuming that γ denotes a set of contacts during a period of time. For each contact (u, v) in γ, two directed edges (u, v) and (v, u) are generated in the color graphs of both BubbleRap and CROP. The color sets on (u, v) and (v, u) for both BubbleRap and CROP are assigned by considering the following two conditions:

1). Nodes u and v belong to the same community: All packets are forwarded to the node with the larger number of neighbors in the network for both BubbleRap and CROP. If the number of neighbors of node v, denoted as d_v, is larger than the degree of node u, denoted as d_u, then both CROP and BubbleRap set color attributes $\Lambda_{uv} = C$ and $\Lambda_{vu} = \{\emptyset\}$ on their corresponding color graphs, where $C = \{c_0, c_1, \ldots, c_{k-1}\}$ is the community set in the DTN. Otherwise, both CROP and BubbleRap set $\Lambda_{vu} = C$ and $\Lambda_{uv} = \{\emptyset\}$ on their corresponding color graphs.

2). Nodes u and v belong to different communities: The color attribute assignments for BubbleRap and CROP are different in this case. The forwarding rule in BubbleRap requires that all packets except the ones with the same destination community as the sender's community are forwarded to the node with higher degree. If $d_v > d_u$, all packets in node u except those with destinations in community C_u are forwarded from u to v. Therefore, BubbleRap sets $\Lambda_{uv} = C - \{C_u\}$ and $\Lambda_{vu} = \{C_u\}$ in its color graph. Otherwise, BubbleRap sets $\Lambda_{uv} = \{C_v\}$ and $\Lambda_{vu} = C - \{C_v\}$. In CROP, packets are forwarded to the node with higher community relevance to the packet destination community. Thus, CROP sets in its color graph:

$$\begin{cases} \Lambda_{uv} = \{b \mid b \in C, R(C_u, b) < R(C_v, b)\} \\ \Lambda_{vu} = \{b \mid b \in C, R(C_u, b) \geq R(C_v, b)\} \end{cases}$$

5.2 Modified Multi-Commodity Flow in a Color Graph

This study formulates throughput as a modified maximum multi-commodity flow problem in a color graph. A commodity set M is assigned in the beginning, where the source-destination pair of each commodity is randomly selected among the DTN nodes. For each commodity $m \in M$, nodes m^+ and m^- represent the source and destination nodes of commodity m, respectively. $C(m^-)$ represents the community of the destination node of commodity m. For each community $c \in C$, $M(c)$ denotes the set of commodities with destination nodes in community c. In the color graph, all flows of commodity m are allowed to travel on the edge whose color set contains the destination community of commodity m. Hence, for each commodity $m \in M$ and each node i, only the incoming and outgoing edges in node i, which can carry commodity m, are considered. Here, $E_i^+(m) = \left\{(i, j) \mid (i, j) \in E', C(m^-) \in \Lambda_{ij}\right\}$ represents the outgoing edge set of node i which can carry commodity m, and $E_i^-(m) = \left\{(j, i) \mid (j, i) \in E', C(m^-) \in \Lambda_{ji}\right\}$ represents the incoming edge set of node i which can carry commodity m. L_{ij} is the restricted capacity on edge (i, j), which represents the number of contacts between nodes i and j. To compute the maximum multi-commodity flow in the color graph, a variable $X_{i,j}^m$ represents the total flows of commodity $m \in M$ on edge(i, j), where $X_{i,j}^m = 0, 1, \ldots, L_{ij}$. Table 1 lists all of the notations used in this study. The linear-programming equations of the modified multi-commodity flow are introduced as follows.

Table 1 Notation table of multi-commodity flow in the color graph.

Parameters	
Notations	**Explanation**
M	the set of commodities
V	the node set of the color graph
E'	the edge set of the color graph
C	the community set
m^+	the source node of commodity $m \in M$
m^-	the destination node of commodity $m \in M$
$M(c)$	the commodity set whose destinations are in community c
$C(m^-)$	the community which node m^- belongs to
L_{ij}	the capacity of the edge $(i,j) \in E'$
$E_i^+(m)$	$\left\{(i, j) \mid (i, j) \in E', C\left(m^-\right) \in \Lambda_{ij}\right\}$: the outgoing edge set of node i which commodity m is allowed to travel on
$E_i^-(m)$	$\left\{(j, i) \mid (j, i) \in E', C\left(m^-\right) \in \Lambda_{ji}\right\}$: the incoming edge set of node i which commodity m is allowed to travel on
$X_{i,j}^m$	the flow of commodity m on edge (i, j)

Objective Function

$$\max\left(\sum_{m \in M}\sum_{(m^+, j) \in E_{m^+}^+(m)}\left(x_{m^+,j}^m\right)\right) \tag{7}$$

Equation (7) demonstrates the main objective function that maximizes the sum of outgoing flows of the commodities from their corresponding source nodes.

Constraints:

$$\sum_{(i,j) \in E_i^+(m)} x_{i,j}^m - \sum_{(j,i) \in E_i^-(m)} x_{j,i}^m = 0 \ \forall m \in M, i \in V - \{m^+, m^-\} \tag{8}$$

Equation (8) demonstrates the flow conservation constraints on nodes. Except for the source node and the destination node of commodity m, the sum of incoming flows of commodity m in node i is equal to the sum of outgoing flows of commodity m in node i.

$$\sum_{c \in \Lambda_{ij}}\sum_{m \in M(c)} x_{i,j}^m \leq L_{i,j} \quad \forall (i, j) \in E \tag{9}$$

Equation (9) demonstrates the capacity constraints on each

edge. The sum of flows on each edge (i, j) must be less or equal to the capacity of edge (i, j).

$$\sum_{(j,m^+)\in E^-_{m^+}(m)} x^m_{j,m^+} = 0 \quad \forall m \in M \tag{10}$$

$$\sum_{(m^-,j)\in E^+_{m^-}(m)} x^m_{m^-,j} = 0 \quad \forall m \in M \tag{11}$$

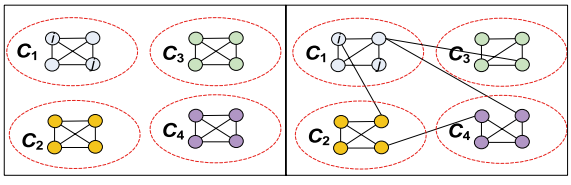
Equations (10) and (11) demonstrate the constraints on source nodes and destination nodes. For the source node m^+ of commodity m, there is no incoming flow of commodity m on node m^+. Similarity, there is no outgoing flow of commodity m on destination node m^-.

Integer programming analysis is time consuming. The computation time of this analysis may increase dramatically when the number of nodes increases. We can use the Lagrangian relaxation method to obtain an approximate result. However, it is beyond the scope of this paper.

5.3 Analysis Results

In this subsection, we obtain the maximum multi-commodity flows in the color graphs of BubbleRap and CROP. Since social graph has the small-word behavior [30], we assume that this social graph can govern the contact of two nodes over time. Hence, we use the CAVE model [19] (i.e., a small-world social graph) to generate synthetic contacts in the contact graph. Then, a corresponding color graph can be derived from the contact graph for both BubbleRap and CROP.

The CAVE model is a social network comprising N nodes which are grouped in cliques (caves) of size g (i.e., the number of the nodes in each clique). Two nodes i and j in the same cave n are categorized as being in the same community C_n. Thus, there are $k = \lceil N/g \rceil$ distinct communities (caves). Figure 6(a) illustrates the CAVE model for $N = 16$ and $g = 4$. In this case, nodes i and j belong to C_1. To create the small-world property in the CAVE model, some shortcuts are created via the rewiring of some edges, where each edge has probability p of being rewired[†]. Fig. 6(b) illustrates the CAVE model after rewiring 4 edges. Nodes i and j are friends if there is an edge (i, j) in the CAVE model. Otherwise, they are strangers. In this simulation, each node generates 10 contacts to form a contact graph. Node i has probability q of having a contact with its friends, and has probability $(1 - q)$ of having a contact with its strangers. Parameter q is equal to 0.5 in each simulation. Each simulation involves 10 commodities, and the source and destination nodes of commodities are randomly selected. The parameter ranges are as follows: N is set from 100 to 200 ($N = 100$ by default), g is set from 10 to 25 ($g = 10$ by default), and p is set from 0.1 to 0.25 ($p = 0.1$ by default). In each simulation, only one parameter functions as a variable, while the other parameters are set at their default values.



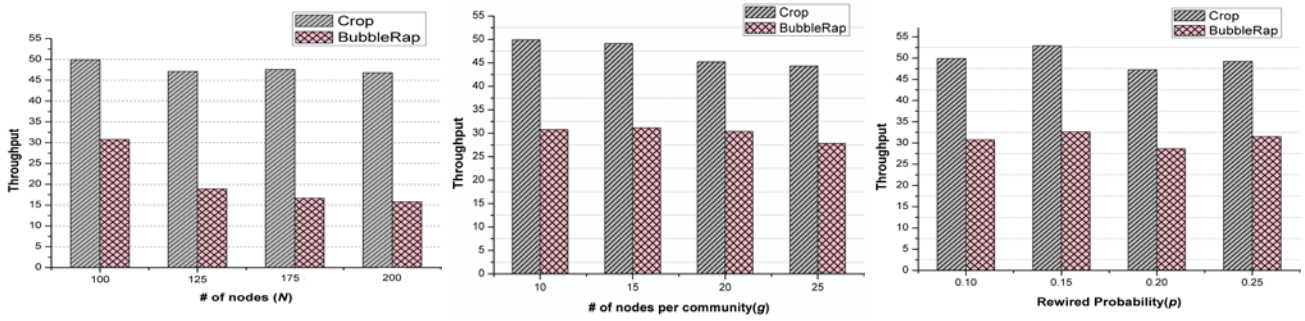
(a) Original graph (b) Rewired-edge graph

Fig. 6 Caveman model.

[†]Probabilistically, there are $N \cdot (k - 1) \cdot p$ rewired edges in the CAVE model.

To compute the community-relevance values, CROP generates a community graph based on the contact graph generated from the CAVE model with rewired edge. If nodes i and j, belonging to different communities, have a contact to each other, then two contacts $\left(C_i, C_j\right)$ and $\left(C_j, C_i\right)$ are generated in the community graph, where C_i and C_j are the communities nodes i and j belong to, respectively. Both BubbleRap and CROP require the nodes' neighbor information. According to the contact generation rule in this simulation, a node has more chance to have contact with its friend. Therefore, the simulations in this study use the number of friends of node i in the CAVE model as the number of neighbors of node i for both routing protocols. After computing the community relevance between any two communities and obtaining the neighbor information of the nodes, BubbleRap and CROP can each generate their corresponding color graphs based on the principles described in Sect. 5.1.

Figure 7(a) compares the throughputs (i.e., maximum multi-commodity flows) under different numbers N of nodes. The CROP throughput is greater than that of BubbleRap. Specifically, CROP achieves a 60–200% improvement in throughput compared to BubbleRap. Since the diameter of the network increases as N increases, it takes more hops for a commodity to reach its destination. It is easier for a commodity to stuck in the intermediate node with the highest centrality metric among its neighborhood for both BubbleRap and CROP. Hence, the number of feasible paths for a commodity should decrease for both routing protocols. Under this circumstance, the throughputs of both BubbleRap and CROP decrease as N increases. Since BubbleRap uses a single centrality metric while CROP supports multiple centrality metrics for different destination communities in message delivery, the decreasing feasible paths should be less severe in CROP compared to BubbleRap. Thus, the difference between the throughputs of these two protocols increases as N increases. For number of nodes N=175, CROP maintains almost the same throughput as in N=125, this may be due to our multiple-centrality metric design that can help to spread traffic loads more widely to keep a good throughput. Figure 7(b) compares the throughputs under different sizes g of nodes per community. CROP has approximately 50–60% throughput improvement over BubbleRap. As g increases, the number of communities decreases. Thus, the throughputs of both routing protocols slightly decrease, because the decreasing number of communities affects the ac-



(a) Throughput under different amounts of nodes

(b) Throughput under different amounts of nodes per community

(c) Throughput under different rewired probability

Fig. 7 The throughput comparison between BubbleRap and CROP under different network environment.

curacy of the delivery metric on both protocols. Figure 7(c) compares the throughputs under different rewired probability p values on each edge. CROP achieves approximately 50–60% greater throughput than BubbleRap. For $p = 0.15$, both CROP and BubbleRap have the largest throughput When $p = 0.2$ and $p = 0.25$, the throughputs slightly decease for both protocols. This phenomenon may be affected by the number of intra-community and inter-community connections. As p increases, the inter-community connections increase but intra-community connections decrease. Then, it may increase the throughput for inter-community traffic, but decrease the throughput for intra-community traffic. These two factors may cause the total throughput to fluctuate.

5.4 CROP Protocol Property Analysis

This study shows that there is always a path between any two communities in the CROP protocol. The following theorems prove this statement.

Theorem 1. For any community i and destination community b where $i \neq b$, there exists a neighbor community $j \in N(i)$ of community i such that the expected random walk distance $h(j,b)$ between community j and destination community b is smaller than $h(i,b)$ between the communities i and b.

Proof. This theorem can be proven by contradiction. Suppose that$h(i,b)$ is the minimum among all of its neighbor communities: $h(i,b) \leq \min_{j \in N(i)} (h(j,b))$. Since $h(i,b) = \sum_{j \in N(i)} p_{ij} \cdot h(j,b) + 1$, we can derive:

$$h(i,b) = \sum_{j \in N(i)} p_{ij} \cdot h(j,b) + 1 \geq \sum_{j \in N(i)} p_{ij} \cdot h(i,b) + 1 \quad (12)$$

Since the sum of all transition probabilities from community i to its neighbor communities equals 1, the inequality (13) can be derived:

$$h(i,b) = \sum_{j \in N(i)} p_{ij} \cdot h(j,b) + 1 \geq h(i,b) + 1 \quad (13)$$

Since $h(i,b) \geq h(i,b) + 1$ is never true, the assumption that $h(i,b) \leq \min_{j \in N(i)} (h(j,b))$ is contradicted. Therefore, there exists at least one neighbor community j such that $h(j,b) < h(i,b)$. □

Corollary 1. For any community i and destination community b where $i \neq b$, there exists a neighbor community $j \in N(i)$ such that $R(j,b) > R(i,b)$.

Proof. Since community relevance $R(i,b)$ is assigned $(h(j,b))^{-1}$, the corollary 1 can be derived directly from the result of Theorem 1.

Theorem 2. Under the condition of Corollary1, there exists a path from any community i to a destination community b in the CROP protocol.

Proof. This theorem can be proven by mathematical induction. First, sort all community relevances $R(i,b)$ to a destination community b for $i \in C = \{c_1, c_2, \ldots, c_k\}$ in decreasing order, where C is the community set in the DTN. The term $\langle j_1 = b, j_2, \ldots, j_k \rangle$ represents this decreasing order, where:

$$R(j_1,b) > R(j_2,b) \geq R(j_3,b) \ldots \geq R(j_k,b).$$

Let $P(n)$ denote a path between communities j and b for $j \in \{j_1, j_2, \ldots, j_n\}$. For $n = 2$, there are two communities $j_1 = b$ and j_2. By Corollary 1, there exists a neighbor community of j_2 whose community relevance is larger than the one of community j_2. Therefore, there must be a path from community j_2 to community $j_1 = b$. Hence, $P(2)$ is correct. Suppose that $P(m)$ is correct. The correctness of $P(m+1)$ can be shown as follows. By Corollary 1, one neighbor community v of community j_{m+1} has a higher community relevance to destination community b than community j_{m+1}. Therefore, the packet can be forwarded from j_{m+1} to v in CROP. Since v is in $\{j_1, j_2, \ldots, j_m\}$, there is a routing path $P_{v \to b}$ from community v to destination community b, while $P(m)$ holds. Therefore, there also exists a path from community j_{m+1} to destination community b (i.e., $P_{j_{m+1} \to v \to b}$). Hence, $P(m+1)$ also holds. □

The combination of Theorems 1 and 2, shows that a routing path exists between any two communities in the

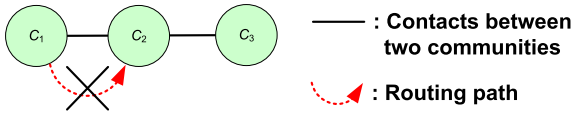


Fig. 8 An example of BubbleRap showing that a routing path may not always exist.

CROP protocol. However, a path between any two communities may not exist in BubbleRap. For example, in Fig. 8, suppose that there are total of three communities in a DTN. There are several contacts from C_1 to C_2 and C_2 to C_3 in the community graph and there are packets being generated in C_1 destined to C_3. The routing path from C_1 to C_2 does not exist in BubbleRap if all the nodes in C_1 have more neighbors than the nodes in C_2. That is because packets in BubbleRap are only forwarded to nodes of higher degrees. Therefore, there is no routing path from C_1 to C_3 in this case. On the other hand, CROP provides multiple centrality metrics for different destination communities. Since community C_1 has only one neighbor community C_2, the rank of relevance to destination community C_3 is $C_3 > C_2 > C_1$ according to Corollary 1. For the same reason, the rank of the relevance to destination community C_1 is $C_1 > C_2 > C_3$. Hence, there always exists a routing path from C_1 to C_3 or from C_3 to C_1.

6. Performance Evaluation

This study further evaluates and compares the performance of CROP with two other social based forwarding protocols: SimBet and BubbleRap using real mobility trace. Section 6.1 introduces the simulation settings. Section 6.2 shows simulation results of CROP for a limited buffer size on each node, where each node only belongs to a single community. Section 6.3 shows simulation results of CROP for a limited buffer size on each node, where each node can belong to multiple communities.

6.1 Simulation Settings

This subsection compares CROP with two well-known social forwarding algorithms, namely, SimBet [13] and BubbleRap [20]. We also measure the performances of the centralized CROP (C_CROP) and distributed CROP (D_CROP) algorithms. The simulation settings are as follows. Each node generates ρ packets every hour, and the destination node of a packet is randomly selected. The range of ρ is set from 1 to 64 (ρ = 10 by default). To measure the effects of buffer sizes in different social forwarding protocols, each node has a buffer of b packets, where b ranges from 500 to 4000 (b = 1000 by default). For D_CROP, if a contact has an accumulated time larger than 100 sec over the last 1hr, then a contact message is generated in the network. For a cluster head of a community, the time interval to update the community-relevance vector is set to 2 hrs. This study measures three performance metrics: delivery ratio, delay, and drop rate.

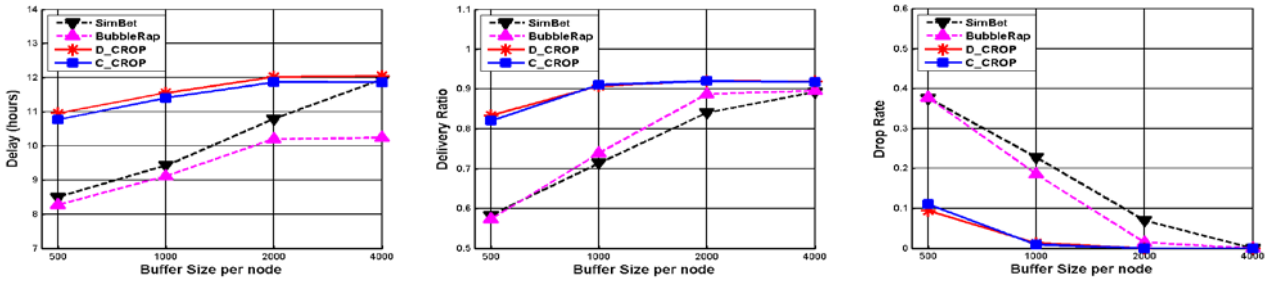
- **Delivery ratio**: The proportion of successfully delivered packets to all packets sent in the network.
- **Delay:** The amount of time (i.e., in hours) it takes a successfully delivered packet to be forwarded from its source node to its destination node.
- **Drop rate:** The proportion of dropped packets to all packets sent in the network due to buffer overflow.

Different from the throughput analysis results in Sect. 5.3, the simulation results in this section focus on following perspectives. Firstly, the simulation results evaluate performance of the social based routing in DTNs under real mobility traces, whereas the analysis results use synthetic data. Secondly, the simulation results consider the occurrence time of contacts in DTNs, whereas the analysis results do not consider this factor. Thirdly, simulation results consider the buffer size on a node as a limited factor but the transmission ability is an infinity factor (i.e., a node can transmit all the packets in its buffer to other nodes when they are in contact with each other). On the other hand, the analysis results consider the transmission ability as a limited factor but the buffer size is an infinity factor on a node. Although throughput analysis is performed in a more ideal condition, analysis result can be used for forecasting the relative performance between BubbleRap and CROP in terms of delivery ratio and drop rate.

6.2 Node Belonging to Single-Community Case

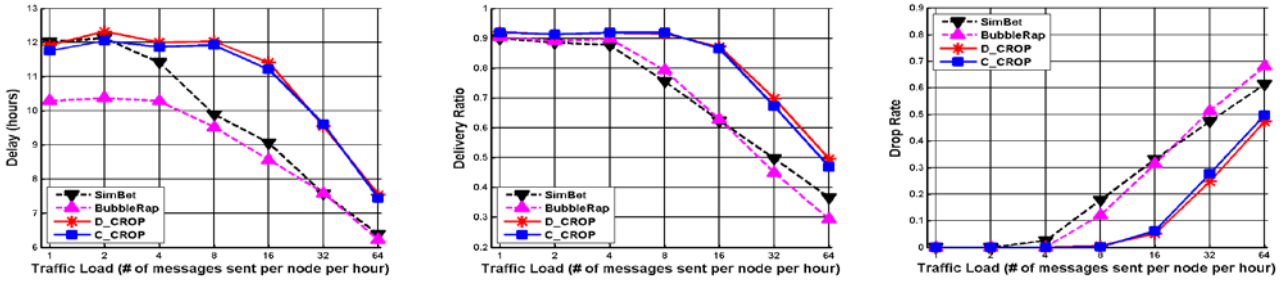
This subsection discusses the protocols' performances under the condition that a node in the network belongs to a single community. The simulation for the single-community case is set up using the real mobility trace (INFO'05) [31] available in [32]. There are a total of 41 nodes in INFO'05. Since there is no community information in INFO'05, preprocessing was performed in advance using Newman's fast algorithm [33] to assign a community to each node. Newman's fast algorithm divides the social graph's nodes into communities based on the group of nodes within a community that have more dense connections. The algorithm tries to test a particular grouping using a quality or modularity index. Since the number of grouping can grow exponentially, a desicion tree is constructed to minimize the searching space. Newman's fast algorithm categorizes the nodes in INFO'05 into 6 communities.

Figure 9 shows the results of different routing protocols under different buffer sizes b on each node. The delivery ratio of CROP (either C_CROP or D_CROP) outperforms those of BubbleRap and SimBet (Fig. 9(a)). D_CROP achieves a 3–45% delivery improvement compared to BubbleRap and SimBet. The delays of all routing protocols increase as b increases (Fig. 9(b)), while the drop rates of all the routing protocols decrease as b increases (Fig. 9(c)). Since CROP has higher delivery ratio, the packets with longer delay have a higher chance of reaching their destinations. Therefore, CROP has a longer average delay. Since CROP designs the community relevance based on different



(a) Delivery ratio under different b (b) Delay under different b (c) Drop rate under different b

Fig. 9 Performances of routing protocols under different buffer sizes for each node belonging to a single community.



(a) Delivery ratio under different ρ (b) Delay under different ρ (c) Drop rate under different ρ

Fig. 10 Performances of routing protocols under different packet load for each node belonging to a single community.

centrality metrics for different communities, its packet drop rate results are the smallest of all three protocols. Speicifally, the drop rate is approximately 80% less than those of BubbleRap or SimBet. The perfomrance results of C_CROP and D_CROP are almost the same in delivery ratio, delay, and drop rate.

Figure 10 shows the results of different routing protocols under different traffic loads. The delivery ratio decreases as the traffic load increases in all routing protocols (Fig. 10(a)). CROP achieves a 30% average delivery ratio improvement compared to the other routing protocols. The delays in all protocols decrease as the traffic load increases (Fig. 10(b)). Since each node is equipped with a fixed-size buffer, the occurrence of buffer overflow increases as traffic load increases (Fig. 10(c)). Therefore, only short-delay packets can survive in a heavy load environment. In this case, the average delay decreases as the traffic load increases because long-delay packets are dropped more often. CROP provides different centrality metrics for different communities, leading to an even traffic load distribution within the network. This is why CROP achieves a 30–90% improvement in the drop rate. Similar as Fig. 9, all of the performance results of C_CROP and D_CROP are almost the same.

In order to understand the impact of variant centrality values to the traffic distribution on a small number of nodes. Figure 11 shows the percentage of the traffic load in the DTN distributed over the top-k loaded nodes for $b = 500$ and $\rho = 10$, where $k = 1, 2 \ldots, 11$. The top-k loaded nodes

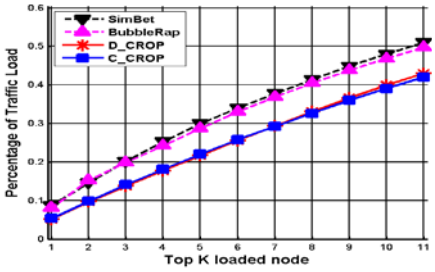
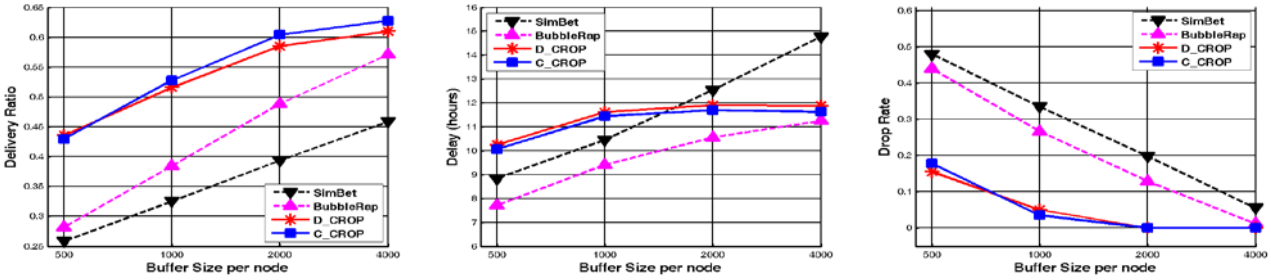


Fig. 11 Traffic load of top k loaded nodes.

in CROP (either C_CROP or D_CROP) have lower total traffic load than the other two social based routing algorithms. When $k = 11$, CROP has 40% of the traffic load centralized on those top 11 loaded nodes, while BubbleRap and SimBet have about 50% of traffic load centralized on those 11 nodes. This result indicates that the multi-centrality metric design in CROP can help to distribute the traffic load in the DTN more fairly.

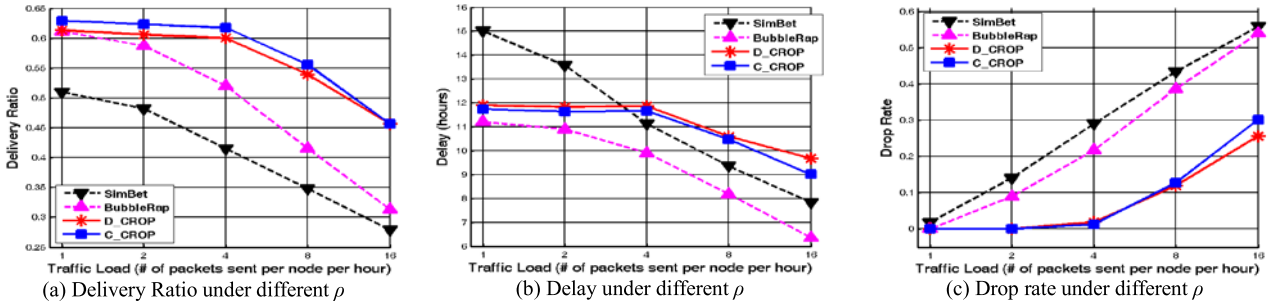
6.3 Node Belonging to Multiple-Community Case

The simulation is set up using the real mobility trace INFO'06 [31]. There are total 98 nodes in INFO'06. Since the INFO'06 trace does not contain the community information, preprocessing was performed. This study uses the α-clique community detection algorithm over a contact graph to assign multiple communities to each node. The authors of [23] defined the α-clique community as a union of all α-



(a) Delivery Ratio under different b (b) Delay under different b (c) Drop rate under different b

Fig. 12 Performances of routing protocols under different buffer size for each node belonging to multiple communities.



(a) Delivery Ratio under different ρ (b) Delay under different ρ (c) Drop rate under different ρ

Fig. 13 Performance of routing protocols under different traffic load for each node belonging to multiple communities.

cliques (complete sub-graphs of size α) in a network that can reach each other via a sequence of adjacent α-cliques. Two α-cliques are said to be adjacent if they share $\alpha - 1$ nodes. This simulation does not use all the trace's contacts for community detection. Only a contact between two nodes whose accumulated contact time is larger than 40000 seconds is taken into consideration. If a node does not belong to any community, then a community is created for that node. The simulations set $\alpha = 3$ and detected 37 communities using the α-clique community detection algorithm.

Figure 12 compares the performances of the three social forwarding protocols under different buffer sizes for each node (i.e., different b values). The delivery ratios of all routing protocols increase as b increases (Fig. 12(a)). D_CROP outperforms the other social forwarding protocols by approximately 9–52% and 36–66% delivery ratio improvement compared to BubbleRap and SimBet respectively. D_CROP alomst has the same delivery ratio compared to C_CROP. The average delays of all routing protocols increase with increasing buffer size b (Fig. 12(b)). This is because long-delay packets have a better chance of reaching their destinations with larger buffer size b. As expected, the drop rates decrease with increasing buffer size b (Fig. 12(c)). The drop rates of C_CROP and D_CROP are smaller than those of Bubble_Rap and SimBet. The differences in the drop rates between CROP (C_CROP or D_CROP) and other social based routings are 10–30% on average. This is because CROP has different community-relevance values for different communities. On the other hand, BubbleRap and SimBet only consider a single centrality metric. Thus, CROP can distribute the traffic load in a balanced manner within the network, unlike BubbleRap and SimBet.

Table 2 Successful rate under different β.

Weight	Successful Rate
β=0.1	0.54
β=0.5	0.55
β=0.9	0.55

Figure 13 compares the performances of all routing protocols under different traffic loads (i.e., different ρ values). The delivery ratios of all routing protocols decrease as the traffic load increases (Fig. 13(a)). D_CROP has a delivery improvement of approximately 2–45% and 20–63% compared to BubbleRap, and SimBet, respectively. The delivery ratio of D_CROP is almost the same as that of C_CROP. The average delays decrease as the traffic load increases (Fig. 13(b)). This is because the drop rate increases (Fig. 13(c)), and long-delay packets have less opportunity to reach their destination nodes. Thus, the average delay of packets that actually reach their destinations is less under these circumstances. The drop rates of all protocols increase with higher traffic loads (Fig. 13(c)). The difference in the drop rate between CROP and other routing protocols also increases as the traffic load increases. This is because CROP uses a multiple-centrality metric to forward packets.

Table 2 evaluates the successful rates of D_CROP with different weighted sum parameter β in Eq. (6) for $b = 1000$ and $\rho = 10$. The results show that β does not affect the suc-

cessful rates. This may be because the community relevance between two communities over time does not change very much.

7. Conclusion

Researchers have developed delay tolerant networks (DTNs) to address the problem of intermitent end-to-end network connectivity. This study computes the community-relevance metric $R(a,b)$ between any two communities a and b. The term $R(a,b)$ is formulated as the inverse of the random walk distance from a to b over the community transition graph. This study proposes a routing scheme, called CROP, based on the community-relevance metric. When two nodes encounter each other, CROP forwards a packet to the node that has a higher community relevance to the destination community of that packet. Since one community may have a higher relevance to one community but have a lower relevance to another community, different centrality metrics can be supported for different communities. The throughput analsyis is formulated as a modified multi-commodity flow problem. The results show that CROP achieves a 50% throughput improvement compared to BubbleRap. A series of proofs show that the path-existence property between any two communities always holds in CROP, but this property may not hold in other routing schemes. Simulation results also show that CROP enjoys at least a 30% delivery ratio improvement compared to other social based routing schemes. Since CROP considers multiple centrality metrics when forwarding packets, CROP achieves a smaller packet drop rate than other social based routing schemes. Since a communtiy may have some sub-communities, hierarchical social community structures may exist in the networks. Our future research on this topic includes computing community relevance over hierarchical social communities.

References

[1] Delay Tolerant Networking Research Group, Available: http://www.dtnrg.org, 2011.
[2] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," Lect. Notes Comput. Sci., vol.3126, pp.239–254, 2004.
[3] B. Burns, O. Brock, and B. Levine, "MV routing and capacity building in disruption tolerant networks," Proc. 24th IEEE Int'l Conf. on Computer Comm. (INFOCOM'05), 2005.
[4] J. Burgess, B. Gallagher, D. Jensen, and B. Levine, "Maxprop: Routing for vehicle-based disruption-tolerant networks," Proc. 25th IEEE Int'l Conf. on Computer Comm. (INFOCOM'06), 2006.
[5] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot, "Delegation forwarding," Proc. 9th ACM Int'l Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc'08), 2008.
[6] C. Liu and J. Wu, "Practical routing in a cyclic MobiSpace," IEEE/ACM Trans. Netw., vol.19, pp.369–382, 2011.
[7] C. Liu, J. Wu, and I. Cardei, "Message forwarding in cyclic MobiSpace: The multi-copy case," Proc. 6th IEEE Int'l Conf. on Mobile Ad-hoc and Sensor Systems (MASS'09), 2009.
[8] W. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," Proc. 5th ACM Int'l Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc'04), 2004.
[9] W. Zhao, M. Ammar, and E. Zegura, "Controlling the mobility of multiple data transport ferries in a delay-tolerant network," Proc. 24th IEEE Int'l Conf. on Computer Comm. (INFOCOM'05), 2005.
[10] M.M. Bin Tariq, M. Ammar, and E. Zegura, "Message ferry route design for sparse ad hoc networks with mobile nodes," Proc. 7th ACM Int'l Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc'06), 2006.
[11] Y. Chen, W. Zhao, M. Ammar, and E. Zegura, "Hybrid routing in clustered DTNs with message ferrying," Proc. 1st International MobiSys Workshop on Mobile Opportunistic Networking (MobiOpp'07), 2007.
[12] Z. Zhang and Z. Fei, "Route design for multiple ferries in delay tolerant networks," Proc. IEEE Wireless Comm. and Networking (WCNC'07), 2007.
[13] E.M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant manets," Proc. 8th ACM Int'l Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc'07), 2007.
[14] J.A. Bitsch Link, N. Viol, A. Goliath, and K. Wehrle, "SimBetAge: Utilizing temporal changes in social networks for pocket switched networks," Proc. 1st ACM Workshop on User-Provided Networking: Challenges and Opportunities, 2009.
[15] F. Li and J. Wu, "LocalCom: A community-based epidemic forwarding scheme in disruption-tolerant networks," Proc. IEEE 6th Ann. IEEE Comm. Soc. Conf. Mesh and Ad Hoc Comm. and Networks (SECON'09), 2009.
[16] W. Gao, Q. Li, B. Zhao, and G. Cao, "Multicasting in delay tolerant networks: A social network perspective," Proc. 10th ACM Int'l Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc'09), 2009.
[17] C. Liu and J. Wu, "An optimal probabilistic forwarding protocol in delay tolerant networks," Proc. 10th ACM int'l symp. on Mobile ad hoc networking and computing (MobiHoc'09), 2009.
[18] J. Pujol, A. Toledo, and P. Rodriguez, "Fair routing in delay tolerant networks," Proc. 28th IEEE Int'l Conf. on Computer Comm. (INFOCOM'09), 2009.
[19] T. Hossmann, T. Spyropoulos, and F. Legendre, "Know thy neighbor: Towards optimal mapping of contacts to social graphs for DTN routing," Proc. 29th IEEE Int'l Conf. on Computer Comm. (INFOCOM'10), 2010.
[20] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay tolerant networks," IEEE Trans. Mobile Comput., vol.10, pp.1576–1589, 2010.
[21] A. Mtibaa, M. May, C. Diot, and M. Ammar, "Peoplerank: Social opportunistic forwarding," Proc. 29th IEEE Int'l Conf. on Computer Comm. (INFOCOM'10), 2010.
[22] W. Gao and G. Cao, "User-centric data dissemination in disruption tolerant networks," Proc. 30th IEEE Int'l Conf. on Computer Comm. (INFOCOM'11).
[23] G. Palla, I. Dernyi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," Nature, vol.435, pp.814–818, 2005.
[24] J.-W. Chang and C. Chen, "CROP: Community-relevance-based opportunistic routing in delay tolerant networks," Wireless Communications and Networking Conference (WCNC), 2013.
[25] Y. Wang, W.-S. Yang, and J. Wu, "Analysis of a hypercube-based social feature multipath routing in Delay tolerant networks," IEEE Trans. Parallel Distrib. Syst., vol.24, pp.1706–1716, 2013.
[26] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," Comput. Commun. Rev., vol.34, pp.145–158 2004.
[27] S. Merugu, M. Ammar, and E. Zegura, "Routing in space and time in networks with predictable mobility," Technical Report GIT-CC-04-7, Georgia Institute of Technology 2004.
[28] Q. Yuan, I. Cardei, and J. Wu, "Predict and relay: an efficient routing in disruption-tolerant networks," Proc. 10th ACM int'l symp. on Mobile ad hoc networking and computing (MobiHoc'09), 2009.
[29] M.E.J. Newman, "The structure and function of complex networks," SIAM Review, pp.167–256, 2003.

[30] D. Watts and S. Strogatz, “The small world problem,” Collective Dynamics of Small-World Networks, vol.393, pp.440–442, 1998.
[31] 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.
[32] CRAWDAD data set. Available: http://crawdad.cs.dartmouth.edu/index.html, 2014.
[33] M.E.J. Newman, “Fast algorithm for detecting community structure in networks,” Phys. Rev. E, vol.69, p.066133, 2004.

Je-Wei Chang received his B.S. degree in Management Science from National Chiao Tung University, Hsinchu, Taiwan, in 2003 and the M.S. degree in Computer Engineering from National Tsing Hua University, Hsinchu, Taiwan in 2005. He is currently working toward the Ph.D. degree in the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan. His current research interests include delay tolerant networks, opportunistic networks, social networks, and software defined network.

Chien Chen received his B.S. degree in Computer Engineering from National Chiao Tung University in 1982 and the M.S. and Ph.D. degrees in Computer Engineering from University of Southern California and Stevens Institute of Technologies in 1990 and 1996. Dr. Chen hold a Chief Architect and Director of Switch Architecture position in Terapower Inc., which is a terabit switching fabric SoC startup in San Jose, before joining National Chiao Tung University as an Assistant Professor in August 2002. Prior to joining Terapower Inc., he is a key member in Core Network, responsible for a next-generation IP/MPLS switch architecture design. He joined Lucent Technologies, Bell Labs, NJ, in 1996 as a Member of Technical Staff, where he led the research in the area of ATM/IP switch fabric design, traffic management, and traffic engineering requirements. His current research interests include vehicular ad-hoc networks, delay tolerant networks, and content centric networks.