Practical Routing in a Cyclic MobiSpace

Cong Liu and Jie Wu, *Fellow, IEEE*

***Abstract*—A key challenge of routing in delay-tolerant networks (DTNs) is finding routes that have high delivery rates and low end-to-end delays. When future connectivity information is not available, opportunistic routing is preferred in DTNs, in which messages are forwarded to nodes with higher delivery probabilities. We observe that real objects have repetitive motions, whereas no prior research work has investigated the time-varying delivery probabilities of messages between nodes at different times during a repetition of motion of the nodes. We propose to use the *expected minimum delay* (EMD) as a new delivery probability metric in DTNs with repetitive but nondeterministic mobility. First, we model the network as a probabilistic time–space graph with historical contact information or prior knowledge about the network. We then translate it into a probabilistic state–space graph, in which the time dimension is removed. With the state–space graph, we apply the *Markov decision process* to derive the EMDs of the messages. We propose an EMD-based probabilistic routing protocol, called *routing in cyclic MobiSpace* (RCM). To make RCM more practical, we show a simple extension that reduces routing information exchanged among nodes. We perform simulations with real and synthetic traces. Simulation results show that RCM outperforms several existing opportunistic routing protocols.**

***Index Terms*—Cyclic MobiSpace, delay-tolerant networks (DTNs), opportunistic routing protocol, simulation, trace.**

I. Introduction

DELAY-TOLERANT NETWORKs (DTNs) [1] allow for data communication in networks that suffer from frequent network partitioning due to reasons such as high mobility, low density, short radio range, intermittent power, interference, obstruction, and attacks. DTNs have been proposed for use in terrestrial wireless networks that cannot ordinarily maintain end-to-end connectivity, deep-space satellite networks with periodic connectivity, underwater acoustic buoy deployments, and many developing region contexts. These scenarios display a wide range of characteristics. We focus on networks with cyclic mobility and communication patterns. Representative examples of these networks include: 1) the National University of Singapore (NUS) student network [2] on which mobile social software [3] and pocket switch networks [4] can be deployed; 2) mobile networks based on moving nodes, such as vehicles or pedestrians [5], which can provide e-mail and file download services; 3) sensor networks where nodes are put to sleep most of the time and wake up periodically to save energy [6].

Routing in DTNs is an active research area. The Delay Tolerant Network Research Group (DTNRG) [1] has designed a complete architecture to support various protocols in DTNs. A DTN can be described abstractly using a time–space graph, in which each edge contains a set of contacts. A *contact* is a period of time during which two nodes can communicate with each other. On the Internet, intermittent connectivity causes loss of data, whereas DTNs support communication between intermittently-connected nodes using the *store–carry–forward* routing mechanism.

Routing in DTNs poses some unique challenges compared to conventional data networks due to the uncertain and time-varying network connectivity. In [7] and [8], optimal routes in a DTN can be discovered by constructing a time–space graph with future connectivity information (oracle). In practical situations where no oracle is available to reveal future contacts, opportunistic routing [9], [10] is proposed, in which one or more copies of a message are sent along different paths and each copy is always forwarded to the node that has a higher delivery probability. Metrics for delivery probability can be either short-term metrics (e.g., the time that has elapsed since the last encounter), which have short lifespans and require frequent updates, or long-term metrics, which are relatively stable over time. Previous works have proposed a variety of long-term metrics including encounter frequency [10] and social similarity [11]. One advantage of long-term delivery probability metrics is that they are relatively stable once generated from historical connectivity information or prior knowledge on the contact pattern of nodes, avoiding the cost associated with frequent updates.

Many real objects have cyclic motion patterns, and it is therefore possible and valuable in practice to use some cyclic metric to increase the accuracy of the estimated delivery probability. In our previous work [12], we use a cyclic long-term metric called the *expected minimum delay* (EMD), which is the expected time an optimal single-copy forwarding scheme takes to deliver a message generated at a specific time from a source to a destination in a network with cyclic and uncertain connectivity. This paper extends this algorithm toward a practical direction by proposing a method that compresses routing information.

A MobiSpace (or MobySpace [13]) is a Euclidean space (or other higher dimensional space) where nodes can be either mobile or static and they can communicate within each other's transmission range. We define a *cyclic MobiSpace* as a MobiSpace where mobility is cyclic. In a cyclic MobiSpace, if two nodes are often in contact at a particular time in previous cycles, then the probability that they will be in contact at around the same time in the next cycle is high. A cyclic MobiSpace can be

Manuscript received February 03, 2009; revised April 21, 2010; accepted July 19, 2010; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor R. Ramjee. Date of publication October 18, 2010; date of current version April 15, 2011. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61003241 and by the National Science Foundation (NSF) under Grants ANI 0073736, EIA 0130806, CCR 0329741, CNS 0422762, CNS 0434533, CNS 0531410, and CNS 0626240.

C. Liu is with Sun Yat-sen University, Guangzhou 510275, China (e-mail: gzcong@gmail.com).

J. Wu is with Temple University, Philadelphia, PA 19122 USA (e-mail: jiewu@temple.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2010.2079944

modeled with a *probabilistic time–space graph* in which an edge between two nodes contains a set of *discrete probabilistic (DP) contacts*. Each DP contact is associated with a time slot and a contact probability. The contact probability is the probability that the two nodes have contacts during the time slot in every cycle. To calculate the EMD of a message, we translate the probabilistic time–space graph into a *probabilistic state–space graph*, where the time dimension is removed from the edges. Then, we apply the *Markov decision process* to calculate the EMDs of the states. With EMDs, we develop a routing protocol called *routing in cyclic MobiSpace* (RCM).

We evaluate the performance of RCM with the variations of several routing protocols, using the NUS student trace, the University of Massachusetts, Amherst, (UMass) DieselNet trace, and additional synthetic bus traces. Simulation results show that RCM outperforms other protocols. Our contributions are summarized as follows.

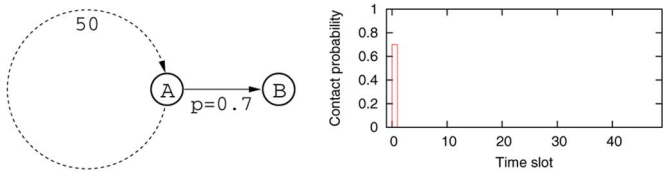
1) We propose a cyclic, long-term delivery probability metric, called EMD, to improve the performance of opportunistic routing in a cyclic MobiSpace.
2) We model the cyclic MobiSpace as a probabilistic time–space graph and translate it into a probabilistic state–space graph.
3) We apply the Markov decision process to solve the EMDs of the states in the probabilistic state–space graph.
4) We discuss a method to reduce the amount of routing information to make our protocol more practical.

This paper is organized as follow. Section II introduces our graph models for the cyclic MobiSpace. Section III proposes the methods to derive EMD in the state–space graph model. Section IV presents a simple extension that reduces routing information exchanged among nodes. Section V shows our simulation methods and results. Section VI summarizes the related works. Section VII concludes the paper with directions for future research.

II. Probabilistic Graph Models

In a cyclic MobiSpace, nodes have cyclic motion and contact patterns, and a common motion cycle T exists for all nodes. For example, the common motion cycle T for a bus system is a day. During each day, the positions of a bus along a regular route at any time can be roughly estimated and can thus be the contact probability distribution between two buses. Comparatively, a general mobile network is a network with completely random mobility where the contact distribution between any pair of nodes over any motion cycle is a uniform distribution. Cyclic MobiSpaces are common in the real world since: 1) the motion cycles of most objects are based on human-defined or natural cycles of time, such as hour, day, and week; 2) most objects' motions are repetitive, time-sensitive, and location-related. Cyclic MobiSpace is defined without any assumption on the shapes of the trajectories, nor are the trajectories required to be deterministic [14].

We will first model a cyclic MobiSpace as a probabilistic time–space graph from which we will transform to a probabilistic state–space graph (a probabilistic graph without time-dependent edges). The purpose of the graph transformation is to apply the Markov decision process (MDP) to calculate the EMD of each message at different states.



(a) (b)

Fig. 1. (a) Nodes A and B have a contact with $p = 0.7$ during time slot 0 every 50 time slots. (b) Contact probability between nodes A and B in each of the 50 time slots.

A. Expected Minimum Delay (EMD)

The EMD is the expected time an optimal single-copy opportunistic forwarding scheme takes to deliver a message generated at a given starting time and a source–destination pair. This single-copy optimal opportunistic scheme always minimizes the delay of each message by forwarding the message in the network to nodes with decreasing EMDs. As shown in Fig. 1(a), assume node A travels a circular trajectory once every 50 time slots, and only in time slot 1, when the network snapshot is shown in the figure, does it have a contact with static node B with probability 0.7. The contact probability between A and B in all of the 50 time slots is shown in Fig. 1(b). In the beginning of each cycle, i.e., time slot 1, the probability that A can send a message to B is 0.7, which makes the probability of A having to store the message for 50 time slots before the next possible forwarding opportunity 0.3. Suppose the transmission delay from A to B is 0.1, then the EMD D_0 of a message generated at time slot 0 and sent from A to B can be calculated by solving the following equation: $D_0 = 0.7 \times 0.1 + 0.3 \times (50 + D_0)$. Clearly, in any time slot t, the EMD is $D_t = D_0 + (50 - (t+1)\%50)$. EMD is an accurate delivery probability metric that reflects the time-varying delivery probability between each pair of nodes within each cycle of motion.

B. Discrete Probabilistic (DP) Contact

We divide the common motion cycle T into small fixed time slots. For each pair of nodes, we introduce a set of conceptual *discrete probabilistic contacts* (or simply DP contacts), which is a tuple (t, p), where t is a time slot in T and p is the contact probability of the two nodes in the time slot t.

We show an example of the DP contact generated from the UMass DieselNet trace [5], [15], where we consider each subshift as a node (see Section V-C), one day as the common motion cycle, and 1 min as the unit of a time slot. The DP contacts between subshifts 01/AM (the morning subshift of shift number 1) and 03/AM are shown in Fig. 2(a). Fig. 2(b) shows those between another pair of subshifts.

C. Probabilistic Time–Space Graph

We model a cyclic MobiSpace as a probabilistic time–space graph $G = (V, E, T)$, where V is the set of nodes, E is the set of edges between the nodes, and T is the common motion cycle.

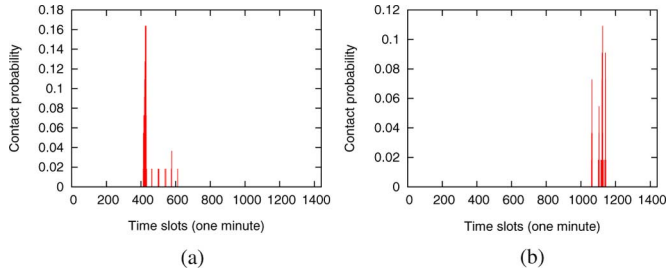


Fig. 2. Discrete probabilistic contacts between different subshifts in the UMass DieselNet trace. (a) Shifts 01/AM and 03/AM. (b) Shifts 32/PM and 21/EVE.

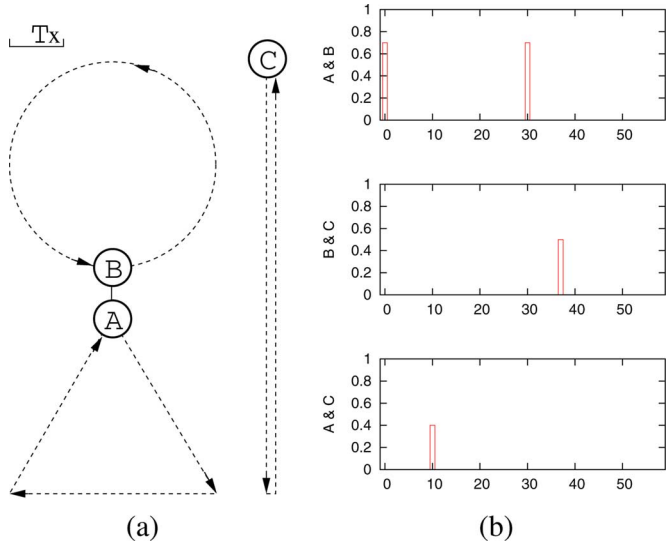


Fig. 3. (a) Physical cyclic MobiSpace. (b) Discrete probabilistic contacts between each pair of nodes of a common motion cycle T.

An edge between two nodes is a nonempty set of DP contacts between the two nodes.

Fig. 3(a) shows a sample cyclic MobiSpace that we will use throughout this paper. In this figure, nodes A and B move in their triangular and circular trajectories, respectively, with a cycle time of 30 units each, while node C travels along its straight-line trajectory with a cycle time of 20 units. Suppose nodes A, B, and C have contacts only during particular time slots in a common motion cycle $T = 60$ units, and these contacts are nondeterministic in nature due to uncertainty in the nodes' positions, communication failures, etc. The set of DP contacts for each pair of nodes in this cyclic MobiSpace is shown in Fig. 3(b).

The time–space graph G of the network is shown in Fig. 4(a). In G, each edge contains the set of DP contacts. For example, edge (A, B) contains two DP contacts. One is labeled (0, 0.7), which means its time slot is 0 and contact probability is 0.7.

D. Probabilistic State–Space Graph

In order to remove the time dimension from the edges in G, we generate a probabilistic state–space graph $G' = (V', E')$, where V' is the set of states and E' is the set of links that are time-independent. G' is generated as follows: For each node u in G, we create a set of states $\{t_i/u\}$ for each time slot t_i when u has one or more DP contacts. For example, three states $0/A$, $10/A$, and $30/A$ are created in G' [Fig. 4(b)] for node A in G [Fig. 4(a)] since A has three DP contacts (0,0.7), (10,0.4), and (30,0.7). If node u has more than one contact (with different nodes) in the same time slot, then only one state is created for u at this time slot.

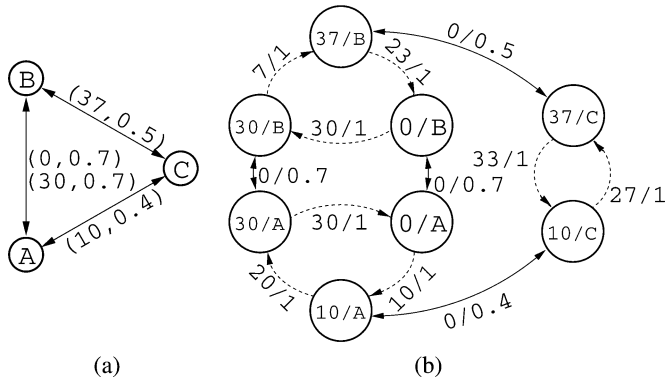


Fig. 4. (a) Time-space graph G and (b) state-space graph G' of the cyclic MobiSpace in Fig. 3(a).

There are two types of links in G': *directional link* and *bidirectional link*. The directional link connects the consecutive states of a node into a ring. For example, the three states of node A in G' are connected into a ring by three directional links represented by dashed lines, as shown in Fig. 4(b). The bidirectional link in G' is created corresponding to each DP contact in G. For each DP contact between nodes u and v in time slot t, a bidirectional link is created in G' between states t/u and t/v [shown as a solid line in Fig. 4(b)].

Each state in G' is a possible state of a message in the network, and each link in G' is a possible state transition of a message. A message is in state $30/A$ if it is in node A at time slot 30. If the message is kept in node A from time slot 30 to time slot 0 (which is a time span of 30 slots), then the message transits from states $30/A$ to $0/A$ via a *directional* link. If the message is forwarded in time slot 30 from node A to node B, then it transits from states $30/A$ to $30/B$ via a *bidirectional* link.

Both types of links are labeled $d/p^{\max}$, where d is the transition delay and $p^{\max}$ is the maximal transition probability. For a directional link, $p^{\max}$ is always equal to 1, which means a node can always choose not to forward a message. For a bidirectional link, $p^{\max}$ is equal to the contact probability of the corresponding DP contact, which means that the forwarding probability cannot exceed the contact probability. For simplicity, we let $d = 0$ for all bidirectional links, assuming that message forwarding is always restricted to a time slot.

E. Remarks

In G', a message can only be at a state in a time slot. This results in an important difference between G and G': Links in G' are time-independent, and they are always available when messages are in their source states. For example, whenever a message is in state $30/A$, it always has a chance of being transited to $30/B$ through the DP contact available at time slot 30. In the rest of this paper, we use the EMD of a state to refer to the EMD of the message in this state.

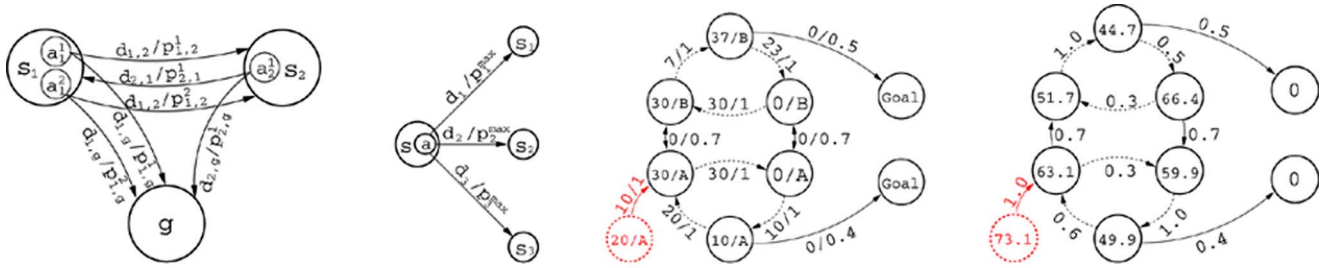


Fig. 5. (a) Markov decision process (MDP) model. (b) Implicit action. (c) MDP of G' in Fig. 4(b) with C as destination. (d) Values and optimal actions after applying value iteration in (c).

Our probabilistic graph models differ from the models defined in previous papers [7], [8], where a contact is a deterministic connection during a specific period of time, while a DP contact is conceptual and is drawn from the cyclic contact history or from the prior knowledge of the cyclic contact pattern between a pair of nodes. The purpose of the discretization is the creation of our graph model G' with time-independent links where EMDs are calculated. Note that the translation from G to G' is lossless since G can be reproduced from G' by combining the states of every node.

III. Expected Minimum Delays in a State–Space Graph

This section reviews and reformulates a variation of the Markov decision process. Using MDP, the values associated with the states in G' are updated iteratively and finally converge to their EMDs.

A. Markov Decision Process (MDP)

State–space searching is a very common problem in AI planning (or decision making), which is similar to routing. The MDP [16] provides a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under control of the decision maker. MDP is a generalized Dijkstra's algorithm for probabilistic graphs.

We reformulate a variation of MDP as a 5-tuple (S, A, T, D, S_G), as shown in Fig. 5(a) and explained below. At any given time, the system can be at only one state s in the set of all states S. In Fig. 5(a), $S = \{s_1, s_2, g\}$. Each state s has a set of actions $A_s \subseteq A$ (A is the set of all actions). In Fig. 5(a), s_1 has actions a_1^1 and a_1^2. Only one action a is allowed to take effect at a time. The effect of applying action a is that the system transits from state s to another state. The transition probability function $T_a(s, s')$ specifies the probability of transiting from state s to state s' when applying action a. In Fig. 5(a), the effect of applying action a_1^1 is that the system transits from s_1 to s_2 with probability $T_{a_1^1}(s, s_2) = p_{1,2}^1$, and transits from s_1 to g with probability $T_{a_g^1}(s, g) = p_{1,g}^1$. Note that for any state s and any of its actions a, the probabilities that s transits to other states when applying a must sum up to 1. That is, $\sum_{s' \in S} T_a(s, s') = 1$. The delay function $D(s, s')$[1] specifies the delay of the transition from state s to state s'. For example, the delay associated with the transition from s_1 to s_2 is $D(s_1, s_2) = d_{1,2}$, and the delay associated with the transition from s_1 to g is $D(s_1, g) = d_{1,g}$.

The result of applying MDP is a value function $V(s)$ that gives the expected minimum total delay it takes to transit from s to any goal state $g \in S_G$ (the set of all goal states). The value of any goal state g is always 0, i.e., $V(g) = 0$ for any $g \in S_G$. With $V(s)$, the optimal action chosen at each state can also be determined.

Value iteration [17] solves MDPs by iteratively updating the value functions in (1) (Bellman equation) for all states until their values converge. In each round of the iteration $t+1$, based on the resulting values from the previous iteration t, the value $V_{t+1}(s)$ of each state $s \notin S_G$ is updated by choosing an action $a \in A_s$ such that $V_{t+1}(s)$ is minimized. In the right side of (1), when taking an action a, the value of state s is the expected delay $D(s, s') + V_t(s')$ (assuming that s' is the next state) weighted by the probability $T_a(s, s')$ of the transition s to s'

$$V_{t+1}(s) = \min_{a \in A(s)} \sum_{s' \in S} \{T_a(s, s') \times [D(s, s') + V_t(s')]\}. \quad (1)$$

The value functions of all states are considered to converge sufficiently when, after a number t of iterations, for every state s, the difference between values $V_{t+1}(s)$ and $V_t(s)$ is less than some threshold value. It is well known that when the values of the states are properly initialized, the value iteration is guaranteed to converge to the minimum expected values of the states. Proper initialization includes iteratively applying (2) in the state–space graph, which is essentially the Bellman–Ford algorithm

$$V_0(s) = \min_{a \in A(s)} \left\{ \min_{s' : T_a(s, s') > 0} [D(s, s') + V_0(s')] \right\}. \quad (2)$$

B. Deriving EMDS Using MDP

Our probabilistic time–space graph G' differs from the MDP model in that states in G' are not associated with actions that tell the actual state transition probabilities. However, implicit actions can be determined by the node's forwarding preferences, the number of which can be very large. A forwarding preference is an ordered list of nodes that tells, when a node is connected with multiple nodes, whether it should forward the message, and, if yes, to which node to forward it. Inspired by a research work on robotics [18], we derive the optimal implicit action without enumerating all possible implicit actions.

[1] In the traditional MDP models, reward function R or cost function C is used. We change it to delay function D in the context of networking.

Recall that each link in G' represents a possible state transition, which is associated with a delay d and a maximal transition probability $p^{\max}$ (see Section II-D). In Fig. 5(b), state s may transit to three other states for which the corresponding delays and the maximal probabilities are given. If the preferential order of state transitions is $\{s_1, s_2, s_3\}$ (which means s will transit to s_1 whenever possible and will transit to s_3 only when the transition to neither s_1 nor s_2 is possible), the implicit action (which indicates the probabilities p_1, p_2, and p_3 of transiting from s to s_1, s_2, and s_3, respectively) is calculated as $p_1 = p_1^{\max}$, $p_2 = (1 - p_1) \times p_2^{\max}$, and $p_3 = (1 - p_1 - p_2) \times p_3^{\max}$. Note that the sum of all transition probabilities of an action is always 1 since, for each state s, there is at least one directional link, and thus at least one $p^{\max}$ is 1. Also, some transition probabilities might be zero. For example, $p_3 = 0$ if $p_2^{\max} = 1$.

All implicit actions for a state s can be obtained from the permutation of the preferential order of state transitions from state s. Fortunately, for an optimal forwarding protocol, this order is always an increasing order of the expected costs of the transitions. For instance, if s can transit to states s_1 and s_2 and their costs $D(s, s_1) + V(s_1) < D(s, s_2) + V(s_2)$, then state s_1 precedes state s_2 in the preferential order of s. This is because the value of state s is the weight sum $p_1 \times [D(s, s_1) + V(s_1)] + p_2 \times [D(s, s_2) + V(s_2)]$, and to minimize this weight sum given $p_1 + p_2 = 1$, p_1 has to be maximized.

To calculate the EMD of a message in time slot t from node A to node C in the G' [Fig. 4(b)], we modify G' [Fig. 5(c)] as follows: We replace all states of C with goal states *Goal*, whose EMDs are 0, and we remove all outgoing links from *Goal*. If state t/A does not exist in G', then we add state t/A to G' and add a link from t/A to the consecutive state of A. For example, if $t = 20$, the resulting MDP from G' is shown in Fig. 5(c). Applying value iteration on this MDP with implicit actions, we get the values (shown inside the states) and the optimal actions (labeled by the transition probabilities on the links) of each state in Fig. 5(d). Note that in Fig. 5(d), some of the links are removed, such as the one from states $0/A$ to $0/B$. This shows the action in which a message is not forwarded from a node with a lower EMD to a node with a higher EMD.

For an efficient approach to calculate EMDs, please refer to our previous work [12].

C. Routing and Analysis

We propose the opportunistic routing protocol called *routing in cyclic MobiSpace*, which uses EMD as a metric of delivery probability. Its forwarding rule is simple: 1) in the single-copy forwarding case, a node u always forwards messages to node v that it encounters if and only if the message has a smaller EMD in v than in u at the forwarding time; 2) in the multiple-copy forwarding case, tickets are redistributed among nodes proportional to the reciprocal of their EMDs, i.e., a node with a smaller EMD gets more tickets.

RCM uses the long-term metric EMD, and it can have a small amortized overhead for routing information when requiring frequent updates of routing information is not necessary. For example, for a DTN built on a bus system that operates for several years, the contacts of the buses in the first few weeks can be gathered, and the generated probabilistic state–space graph can be disseminated in the network once and for all. Addition or removal of bus routes can be reflected in the graph through incremental updates. The following theorems show the optimality of our algorithm:

Theorem 1: The value iteration and the extended TVI guarantee that the values of the states converge to EMDs.

Proof: The algorithms converge to EMDs because: 1) the values are nondecreasing in each iteration; and 2) the values are minimum upper-bounded by EMDs initially and in each iteration. ■

Suppose D_{AG} is the EMD from nodes A to G at the current time, and D_{ACG} is the EMD from nodes A to C then to G at the current time. Then, $D_{AG} \leq D_{ACG}$ since an additional constraint, passing C, is placed on the possible minimum delay paths between A and G.

Theorem 2: The single-copy opportunistic forwarding scheme proposed is the optimal single-copy opportunistic forwarding scheme in terms of expected delivery latency.

Proof: We need to prove that each forwarding in RCM maximizes the EMD of the message. Let the EMD from nodes A and B to destination G at the current time be D_{AG} and D_{BG}, respectively, and $D_{AG} > D_{BG}$. Then, there does not exist another node C such that the expected delay of forwarding the message through C is smaller than that through B. This is because $D_{BG} < D_{AG} < D_{ACG}$. ■

In RCM, we require that the motion cycle of the network be *a priori* knowledge. Here, we provide a trivial extension of RCM for the networks without a cyclic connectivity pattern. When there is no cyclic pattern, we can regard that the common motion cycle is infinitely small. In this case, the common motion cycle contains a single time slot (whose size does not matter), and each node has a single state in the state–space graph that will be identical to the time–space graph. The MDP performing on the state–space graph and the routing protocol will remain the same for the noncyclic case. Since it employs an optimal single-copy delivery probability, this extension of RCM can have better performance than other single-copy forwarding protocols in noncyclic networks.

IV. Contact Information Reduction

This section presents a simple extension that reduces routing information exchanged among nodes. The number of states in G' depends on the number of nodes and the DP contacts of each node. Methods to reduce the number of states include increasing the size of time slots, dropping DP contacts whose probabilities are below a certain threshold, and reducing the number of DP contacts between each pair of nodes through DP contact clustering, which will be presented here. Reducing the amount of contact information has two advantages in making RCM more scalable in particular networks: 1) networks whose characteristics change frequently and the overhead of contact information updates cannot be ignored; 2) networks of larger sizes and where the computation overhead of the RCMs is significant.

A. Contact Information Reduction Algorithms

Although the number of DP contacts is usually much smaller than the number of time slots, as shown in Fig. 2(a) and (b), there is still a substantial number of them when the time-slot

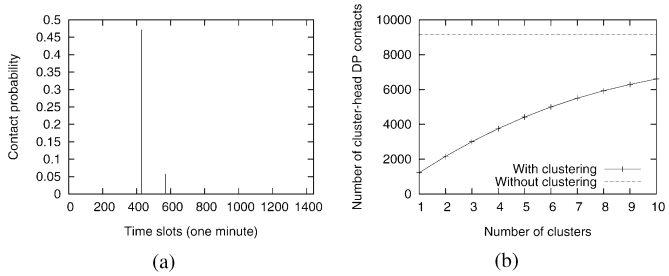


Fig. 6. Discrete probabilistic contacts before and after contact clustering. (a) Cluster-head DP contacts for the DP contacts in Fig. 2(a) and (b). (b) Number of cluster-head DP contacts versus number of clusters.

size is reasonably small. Suppose that, in the UMass DieselNet trace, two buses have a chance to encounter each other at any minute around ± 20 min of 6 a.m. at a bus station due to uncertain factors, such as traffic conditions. There will be 40 DP contacts between these two buses for this single meeting location in total. To reduce the DP contact information, we perform clustering on the DP contacts between each pair of nodes. For each set of DP contacts whose time slots are close, we use a *cluster-head DP contact* to replace them, such that this virtual cluster-head DP contact represents the physical event causing this set of DP contacts.

As an example, the clustering result of the DP contacts in Fig. 2(a) are shown in Fig. 6(a), where two cluster-head DP contacts are used to represent all original contacts that are close to each other. After replacing clusters of DP contacts with two cluster-head DP contacts, we calculate EMDs from the state–space graph constructed from the cluster-head DP contacts. We desire these EMDs to be close to those calculated in the original time–space graphs with nonreduced DP contacts. To this end, we replace a set C_i of DP contacts with a cluster-head DP contact c_i as follows: 1) the contact probability $p(c_i)$ of c_i is the joint probability of the probabilities $p(c_j)$ of the contacts c_j in C_i; and 2) the time slot $t(c_i)$ of c_i is at the mean of the time slots $t(c_j)$ of the contacts c_j in C_i weighted by the probabilities $p(c_j)$. That is

$$p(c_i) = 1 - \prod_{c_j \in C_i} (1 - p(c_j)), t(c_i) = \frac{\sum_{c_j \in C_i} p(c_j) \times t(c_j)}{\sum_{c_j \in C_i} p(c_j)}.$$

We use several agglomerative hierarchical clustering algorithms [19] and the k-mean partitional clustering [20] to perform contact clustering. The distance measurement between two clusters includes: 1) maximum distance; 2) minimum distance; and 3) mean distance. The distance between elements (DP contacts) is measured by the difference of their time slots.

In the ideal situation, the number of cluster-head DP contacts between two nodes should equal the number of physical events causing DP contacts between these two nodes. However, the number of these physical events is usually unknown. We will use a fixed number, k, of clusters to perform contact clustering in the case that the number of DP contacts is larger than k, then we observe how the routing performance degrades as k decreases. The total amount of contact information, in terms of the number of total DP contacts between all pairs of nodes with k ranging from 1 to 10, is shown in Fig. 6(b).

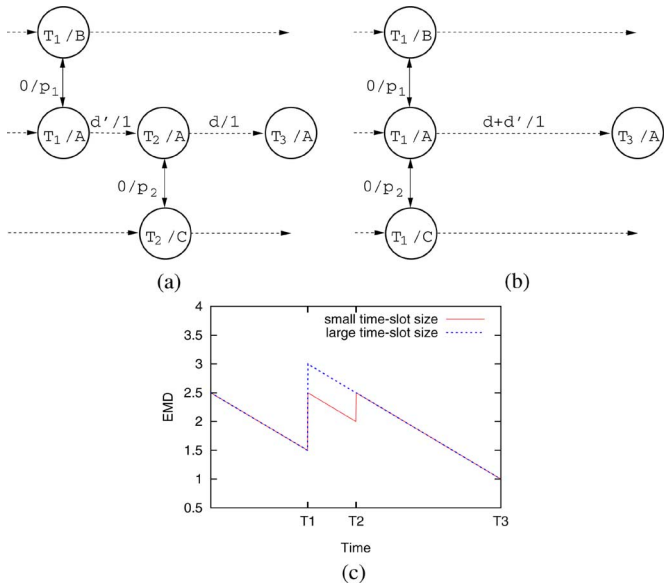


Fig. 7. Effect of time-slot size on EMD accuracy. (a) Small time slot. (b) Large time slot. (c) EMD value of node A.

B. Contact Information Reduction versus EMD Accuracy

In this section, we will analyze the effect of contact information reduction on the accuracy of EMD. We will first discuss the effect of using a larger time-slot size, which is followed by the effect of DP contact clustering, as described in Section IV.

Time-Slot Size: We will show that a larger time-slot size d'' results in larger EMD values during some time span whose length d' is bounded by d''. When using time-slots, we effectively move all the time of the DP contacts within a time slot to the starting time of the time slot. In Fig. 7(a), when using an infinitely small time-slot size, node A has a DP contact with node B at T_1 and a DP contact with node C at T_2, and the difference between them is d'. Let's suppose that when using a larger size for time slots, as shown in Fig. 7(b), the DP contact between nodes A and B remains at T_1, but the DP contact between nodes A and C is moved from T_2 to T_1. This change can be caused by the use of a time slot starting at T_1 and whose size d'' is larger than d'. We will examine the EMD values of a message in node A at different times using the state–space graphs in Fig. 7(a) and (b), respectively. We assume that the EMD values of states T_1/B, T_2/C, and T_3/A [$V(T_1/B)$, $V(T_2/C)$, and $V(T_3/A)$] are fixed, and $V(T_1/C) = d' + V(T_2/C)$.

Clearly, the EMD values of node A at any time t between T_2 and T_3 depend only on the EMD value of states T_3/A and are therefore identical in the two figures. We will show that the EMD values of state T_1/A are identical in the two figures. We can calculate the EMD value $V(T_1/A)$ in the following six different cases: 1) $d+d'+V(T_3/A) < d'+V(T_2/C) < V(T_1/B)$; 2) $d + d' + V(T_3/A) < V(T_1/B) < d' + V(T_2/C)$; 3) $V(T_1/B) < d' + V(T_2/C) < d' + d + V(T_3/A)$; 4) $V(T_1/B) < d' + V(T_2/C) < d' + d + V(T_3/A)$; 5) $V(T_1/B) < d' + d + V(T_3/A) < d' + V(T_2/C)$; and 6) $d' + V(T_2/C) < d' + d + V(T_3/A) < V(T_1/B)$. It is not difficult to see that, in all of the above six cases where the optimal implicit actions are different, the values of $V(T_1/A)$ in

Fig. 7(a) are identical to that in Fig. 7(b). Therefore, the EMD values of state T_1/A are identical in the two figures. Since the EMD values of node A before state T_1/A only depend on $V(T_1/A)$, they are also identical.

As shown in Fig. 7(c), the EMD value decreases continuously when moving from one state to the next state (e.g., from T_2 to T_3), and it has a discrete increase at each state. Let V denote the EMD value in Fig. 7(a) (small time-slot size), and V' denote the EMD value in Fig. 7(b) (large time-slot size) since $V(T_1^- /A) = V'(T_1^- /A)$, and V and V' only differ from each other in $[T_1, T_2)$, as shown in Fig. 7(c). Since $d' = T_2 - T_1$ is bounded by d'', and the EMD value V' at any $t \in [T_1, T_2]$ equals $T_2 - t + V(T_2) < d' + V(T_2)$, we can always make the EMD values satisfactorily accurate by choosing a small d''. The analysis in this section also holds for the following cases: 1) T_1 is not in the beginning of a time slot (both DP contacts are moved ahead when a large time-slot size is used); or 2) state T_1/A does not exist.

DP Contact Clustering: The routing performance will be affected by the fixed number of k clusterhead DP contacts we choose. Let k' be the number of real events that produce the contacts between two nodes. When $k < k'$, at least two events at t_1 and t_2 will be represented by the same cluster-head DP contact at another time t' between t_1 and t_2, which causes a smaller EMD between t_1 and t' and a larger EMD between t' and t_2.

When $k \geq k'$, the DP contacts between the nodes can be abstractly represented by the physical connection events. However, the information about probability distribution of each event across time, as represented by a cluster of DP contacts, is removed. If a cluster of DP contacts are closely clustered around a cluster-head DP contact at $t \pm \Delta$ and Δ is very small, the error induced can be negligible. We will examine the effect of DP contact clustering on real traces with different routing protocols in Section V. As a more advanced method, we can use a dynamic number of cluster-heads and set a maximum distance between a DP contact and its cluster-head DP contact. For simplicity, we adopt a fixed number of cluster-heads in our simulation.

We only perform simulations using the DP contact clustering as our contact information reduction algorithm. This is because enlarging the time-slot size is actually a special case of DP contact clustering with a restriction that each cluster-head must be at the beginning of the corresponding enlarged time slots.

V. Simulation

We evaluate our protocol RCM in the context of other routing algorithms using a wide variety of traces: NUS student contact trace, UMass DieselNet trace, and our synthetic bus traces. We will first describe these network traces and our simulation methods using these traces. Then, we will present simulation results where variations of RCM are implemented using full routing information (DP contacts) and reduced routing information, respectively. Simulation results show that RCM improves the delivery rate and lowers the end-to-end delay in all traces when compared to other protocols.

A. Protocols in Comparison

We compare RCM against several other forwarding protocols. For our focus on the efficiency of delivery probability metrics and for fairness in the comparison, we use variations of these protocols that make use of the same level of prior knowledge of historical connectivity patterns, as RCM does. These algorithms differ from each other only in the delivery probability used to forward messages when a node comes in contact with another node. All algorithms use the same forwarding strategy: A node with a worse delivery probability metric always forwards messages to the node with a better one. When forwarding opportunity is limited due to short contact duration, we prioritize messages with smaller numbers of tickets, followed by their delivery probabilities, when forwarded.

Epidemic [21]: A node copies a message to every node it encounters that does not have a copy already until its copy of the message times out.

Spray&wait [22]: This protocol differs from Epidemic in that it controls the number of copies of each message in the network. A number L of logical *tickets* is associated with each original message. When a message is forwarded, the tickets are split between the two copies in the sender ($\lfloor L/2 \rfloor$ tickets) and the receiver ($L - \lfloor L/2 \rfloor$ tickets). Except to the destination, forwarding is not allowed when a message owns only one ticket.

Spray&focus [23]: It is an extension of Spray&wait. Its difference to Spray&wait is that a message with one ticket is allowed to be forwarded from node i to j and then removed from j if the delivery probability of j to the destination is higher. In [23], a node with the later *last encounter time* with a destination has a higher delivery probability. Transitivity [10] is used to improve predictability in their mobility models. We use a variation, Spray&focus*, in which the average meeting interval drawn from the contact history is used to indicate delivery probability.

MaxProp [5]: A cost is assigned to each node for each destination. Each node i keeps track of a probability f_j^i of the next meeting node being j and disseminates it to every node in the network. The delivery probability from a source to a destination is the total cost $\sum_{(i,j \in P)}(1 - f_j^i)$ on their shortest path P, where the cost of each hop (i, j) is $1 - f_j^i$. In our simulation, we use a variation, MaxProp*, which differs from Spray&focus* in that it uses costs instead of meeting intervals. We draw all f_j^i's from contact history.

All the protocols that we implement only aim to compare different delivery probability metrics. All other optimizations that have orthogonal effects on the performance of the protocols are not implemented. The orthogonality means that these optimizations can be added to all of our implemented algorithms and are expected to provide an equal level of improvement in their routing performance. Such optimizations include buffer management [5], global estimation of message delivery probability [24], social centrality of the nodes [11], geometric information [25], delivery quality threshold [26], acknowledgement mechanism [24], etc.

B. NUS Student Contact Trace

Accurate information of human contact patterns is available in scenarios such as university campuses. As shown by the NUS student contact trace model [2], when class schedules and student enrollment for each class on a campus are known, accurate information about contact patterns between students over large time scales can be obtained without a long-term contact data

collection. Their contact model is simplified in several ways: 1) Two students are in contact with each other if and only if they are in the same classroom at the same time. 2) Sessions start on the hour and end on the hour, which means that hour is the unit of time for contact duration. The advantages of the trace synthesized in this model are that they exhibit the same set of characteristics to those observed in the real world, and it provides contact patterns of a large population over a long period. The schedules of the 4885 classes and enrollment of 22 341 students for each class (77 class hours a week in total) are publicly available on [27].

We generate networks for our experiments by selecting part of the students instead of using all students in the network for the following two reasons: 1) the generated networks allow us to perform experiments with networks that have different characteristics, including network size and degree of connectivities; 2) the storage requirement, which is $O(N^2)$ in a network of N students, and the corresponding computation overhead in the MPD value iterations are overwhelming when $N = 22\,341$. We select a number of N $(100 \leq N \leq 1000)$ students in different simulations. We define a *clustering factor* C to determine the degree of connectivity of the nodes in the network. Specifically, $C = |S_1|/(|S_1| + |S_2|)$, where, for each student s, two sets of students, S_1 and S_2, are defined such that s is similar to the students in S_1 and is dissimilar to those in S_2. Here, the similarity between two students is defined in terms of the number of common class sections in which they enrolled.

We generate nondeterminant traces by taking absentees into consideration. Each student that attends a class has an attendance probability P_{attend}. Our data processing includes the following steps.

1) We break each class session into several 1-h class sessions and reassign unique IDs to them. A total of 159 conflicting enrollments (one student is enrolled in more than one class at the same hour) are resolved by removing the enrollment with a higher class session ID.
2) The selection of N students is not straightforward. If selected randomly, the network becomes too sparse for messages to be delivered. On the other hand, if students are selected by maximizing their similarities, the network becomes overconnected. To prevent the above extremes and maintain the small-world property in the size-reduced student networks, we use the following process. We select the first student randomly. To select the kth student, we randomly divide the $k-1$ selected students into two groups, S_1 and S_2, such that $\frac{S_1}{S_1+S_2} = C$ (the clustering factor), and select the kth student s as the one with the highest score, which is defined by $\sum_{s_1 \in S_1} \text{sim}(s, s_1) - \sum_{s_2 \in S_2} \text{sim}(s, s_2)$ among the students that are not yet selected, where the similarity function sim is defined as the number of common class sessions enrolled by two students.
3) If two students enroll in the same class session, they have a DP contact in the hour of the class session with probability P^2_{attend}. With the DP contacts, the probabilistic state–space graph can be built.
4) Finally, we generate traces by creating contacts with probability P^2_{attend} for each pair of students and each class session in which they enroll.

TABLE I
SETTINGS FOR NUS STUDENT TRACE

parameter name	default	range
number of students (N)	400	100∼1000
attendance rate (P_{attend})	0.8	0.1∼0.9
clustering factor (C)	0.2	0.1∼0.9
tickets (L)	2	2∼6
simulation time	7 days	
simulation times	100	
message time-to-live (T_r)	7 days	1∼7 days
total messages	$500 \times N$	

The settings in our simulation are shown in Table I. In different simulations, we vary one of the five variable parameters within their respective ranges, as shown in the table. For each data point in the results, 100 rounds of simulations are run. In the beginning of each simulation, every node sends messages to five randomly selected nodes, and the total number of messages is $500N$, where N is the number of students. The total simulation time in each experiment is one week. In these traces, we assume unlimited messages can be forwarded in each contact whose duration is 1 h.

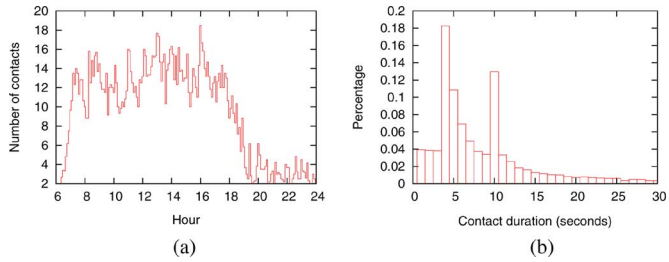
C. UMass DieselNet Trace

Before presenting our simulation method, we give a brief description of the UMass DieselNet [5], [15] test bed and the traces collected on this test bed. We then describe how we preprocess these traces to meet our needs.

In the UMass DieselNet bus system, the bus-to-bus contacts are logged. Our experiments are performed on traces collected over 55 days during the Spring 2006 semester with weekends, spring break, and holidays removed due to reduced schedules. The bus system serves approximately 10 routes. There are multiple shifts serving each of these routes. Shifts are further divided into morning (AM), midday (MID), afternoon (PM), and evening (EVE) subshifts. Drivers choose buses at random to run the AM subshifts. At the end of the AM subshift, the bus is often handed over to another driver to operate the next subshift on the same route or on another route. Unfortunately, the all-bus-pairs contacts provided in the original traces show no discernible pattern. Significant effort is needed to obtain contacts at a subshift level, which do exhibit periodic behavior.

We obtain the subshift level contact by the following steps. Each subshift has a fixed starting time (TIME_AT_GARAGE) and a fixed ending time (DRVR_CHNG) everyday. We obtain a mapping from subshifts to these times by parsing one of the dispatch records *DA_all.txt*. For example, subshift 21/AM (the AM subshift of shift 21) starts at 6:10 a.m. and ends at 10:30 a.m. We also obtain a mapping from day and bus to the subshifts served by the bus on that day by parsing *DB_sheet.txt*. With the above two mappings, we translate 55 days of bus-to-bus contacts into contacts between subshifts. A virtual contact is created between two subshifts if a bus is handed over from one subshift to another. Fig. 8(a) and (b) shows the distribution of all contacts over a day and the distribution of the contact duration at the subshift level.

The DP contacts between any pairs of subshifts are then generated from the 55 days of subshift-based contacts. Examples of DP contacts between two pairs are shown in Fig. 2(a) and (b). We set the time slot to be 1 min. In the trace of a particular day,



(a) (b)

Fig. 8. Statistics in the UMass DieselNet traces. (a) Contact distribution. (b) Duration distribution.

TABLE II
SETTINGS FOR UMASS DIESELNET TRACE

parameter name	default	range
number of sub-shifts (N)	92	
tickets (L)	2	1∼5
simulation times	100	
message time-to-live (T_r)	5 days	3∼7 days
number of messages	$10 \times N$	$2 \sim 20 \times N$
fowarding rate (T_r)	100 msgs/second	
simulation time	T_r days	
total messages	$5000N$	

if two subshifts have one or more contacts during the same time slot as a discrete probability contact, then the contact probability of the discrete probability contact is increased by 1/55. With DP contacts, we generate the probabilistic time–space graph and the probabilistic state–space graph for RCM, the intermeeting time for Spray&focus*, and the next meeting probability f for MaxProp*. Details are omitted here.

The settings of the UMass DieselNet trace simulation are shown in Table II. In different simulations, messages are created with one to five tickets (two tickets by default), respectively. We use the 55 days of traces to run 100 simulations under each simulation setting. Each message is created with a time to live (T_r), ranging from three to seven days (five days by default). The length of a simulation is T_r days when the message time to live is T_r. Every node (subshift) sends 2 to 20 messages (10 messages by default) for random destinations in the beginning of each simulation. That is, each node sends a total of 200–2000 messages in each simulation setting.

We further process the traces such that all communication links between the nodes become bidirectional. In these traces, the average number of contacts per subshift per day is 8.5, and the contact duration is 12 s on average. We restrict the number of messages forwarded in each contact opportunity to 100 messages per second of contact duration. Messages are created in the beginning of each simulation. To vary the level of per-message forwarding opportunity in the simulations, the number of messages created by each node ranges from 2 to 20 (10 by default).

D. Synthetic Bus Trace

We will compare RCM and other protocols with two more sets of synthetic bus traces we generated. These synthetic bus traces are generated from metro maps found on the Internet, as shown in Fig. 9. We develop a tool to help the extraction of routes from the maps. On each route, we simulate a number of buses traveling back and forth.

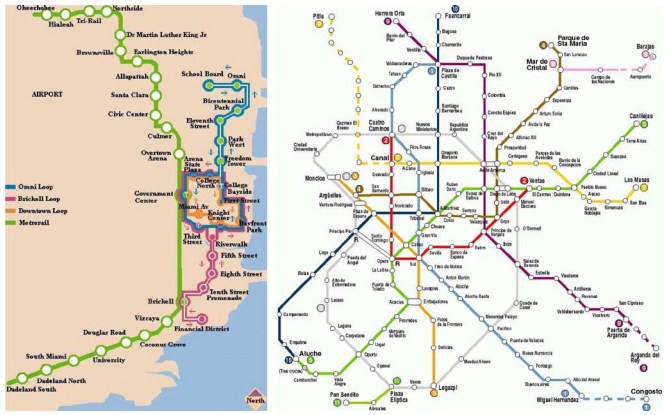


Fig. 9. Our synthetic bus traces are generated from metro maps. (Left) Miami, FL. (Right) Madrid, Spain.

TABLE III
SETTINGS FOR OUR SYNTHETIC BUS TRACE

parameter name	Miami trace	Madrid trace
routes	4	11
motion cycle	240(mins)/4(hrs)	360(mins)/6(hrs)
nodes	48	235
messages	1152	84600
simulation time	40(hrs)	60(hrs)

Each bus travels a route starting from one of the stations on the route. When several buses travel on the same route, their starting stations are dispersed evenly along the route. If the stations of a route are arranged as a line instead of a circle, the motion cycle of the buses in this route is the round-trip time on this route. The time it takes for each bus to travel between two consecutive stations is in a Poisson distribution with a mean of 5 min. The time a bus stays at a station is 1 min. Buses are in contact with each other only when they are in the same station. Unlike the UMass DieselNet, in our synthetic trace both buses and bus stations are nodes, where bus stations function as Info-stations [28]. We assign motion cycles to buses on each route according to the number of stations on that route. For the Miami, FL, map, the motion cycles for different routes are 60, 120, and 240 min. For the Madrid, Spain, map, the motion cycles are 30, 120, 180, and 360 min. When a bus finishes a round trip on its route, but takes less time to complete the route than the motion cycle of its route, the bus must stop in its first station for the rest of the time.

The settings of our synthetic bus traces simulation are shown in Table III. In the Miami traces, there are four routes and 48 nodes (including buses and bus stations). In the Madrid traces, there are 11 routes and 235 nodes. Each node sends one message whose destination is another node selected randomly. The total simulation time is 10 motion cycles for both traces.

E. Simulation Results With Full Routing Information

NUS Student Contact Trace. The delivery rates of the forwarding algorithms are compared in Fig. 10(a), (d), (g), (j), and (m) with different attendance rates, clustering factors, message times to live, numbers of students, and numbers of initial tickets per messages. The delivery rates of all protocols increase as the

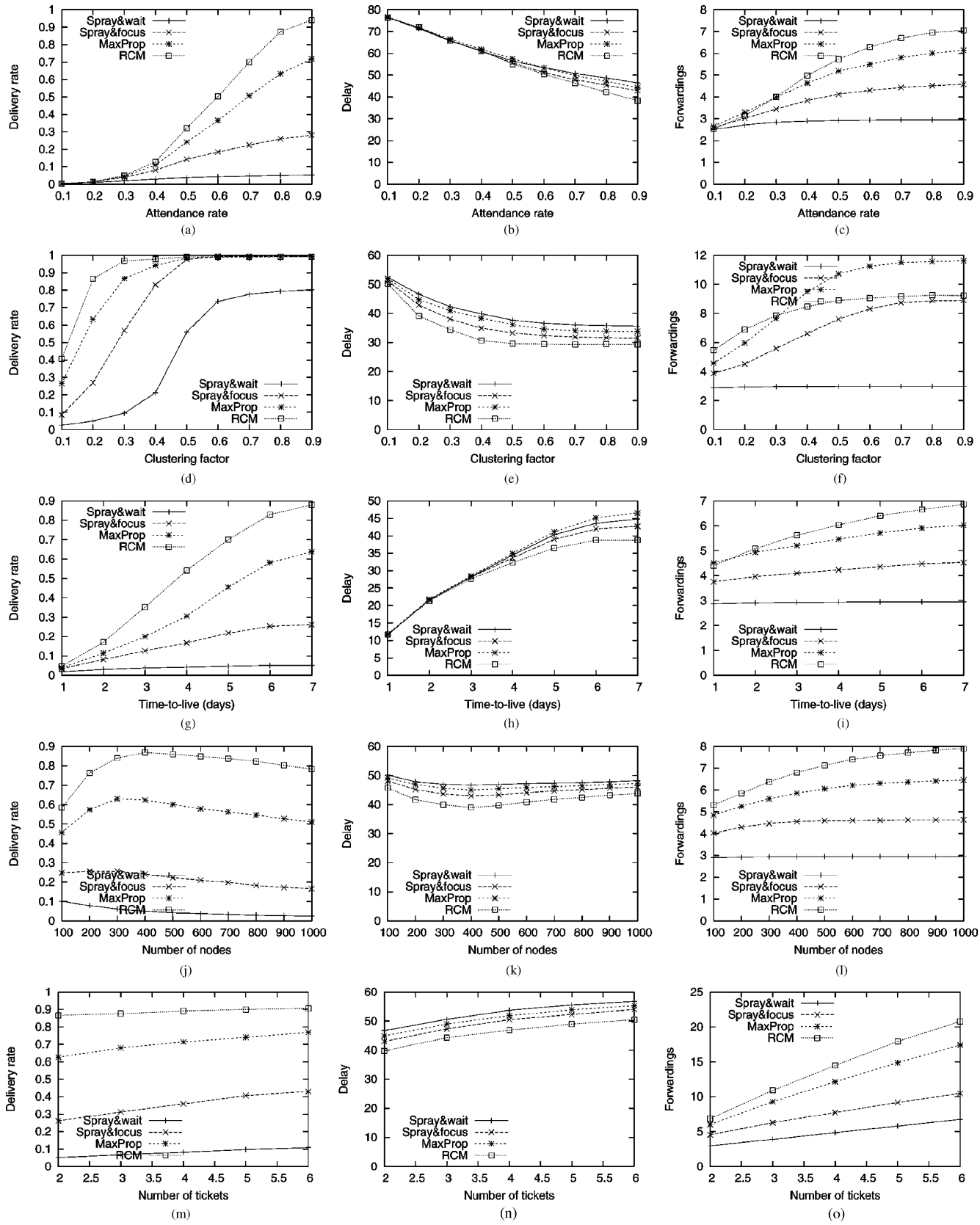


Fig. 10. Results in the NUS student contact traces. (a) Attendance rate versus delivery rate. (b) Attendance rate versus delay. (c) Attendance rate versus forwardings. (d) Clustering factor versus delivery rate. (e) Clustering factor versus delay. (f) Clustering factor versus forwardings. (g) Time to live (days) versus delivery rate. (h) Time to live (days) versus delay. (i) Time to live (days) versus forwardings. (j) Number of nodes versus delivery rate. (k) Number of nodes versus delay. (l) Number of nodes versus forwardings. (m) Number of tickets versus delivery rate. (n) Number of tickets versus delay. (o) Number of tickets versus forwardings.

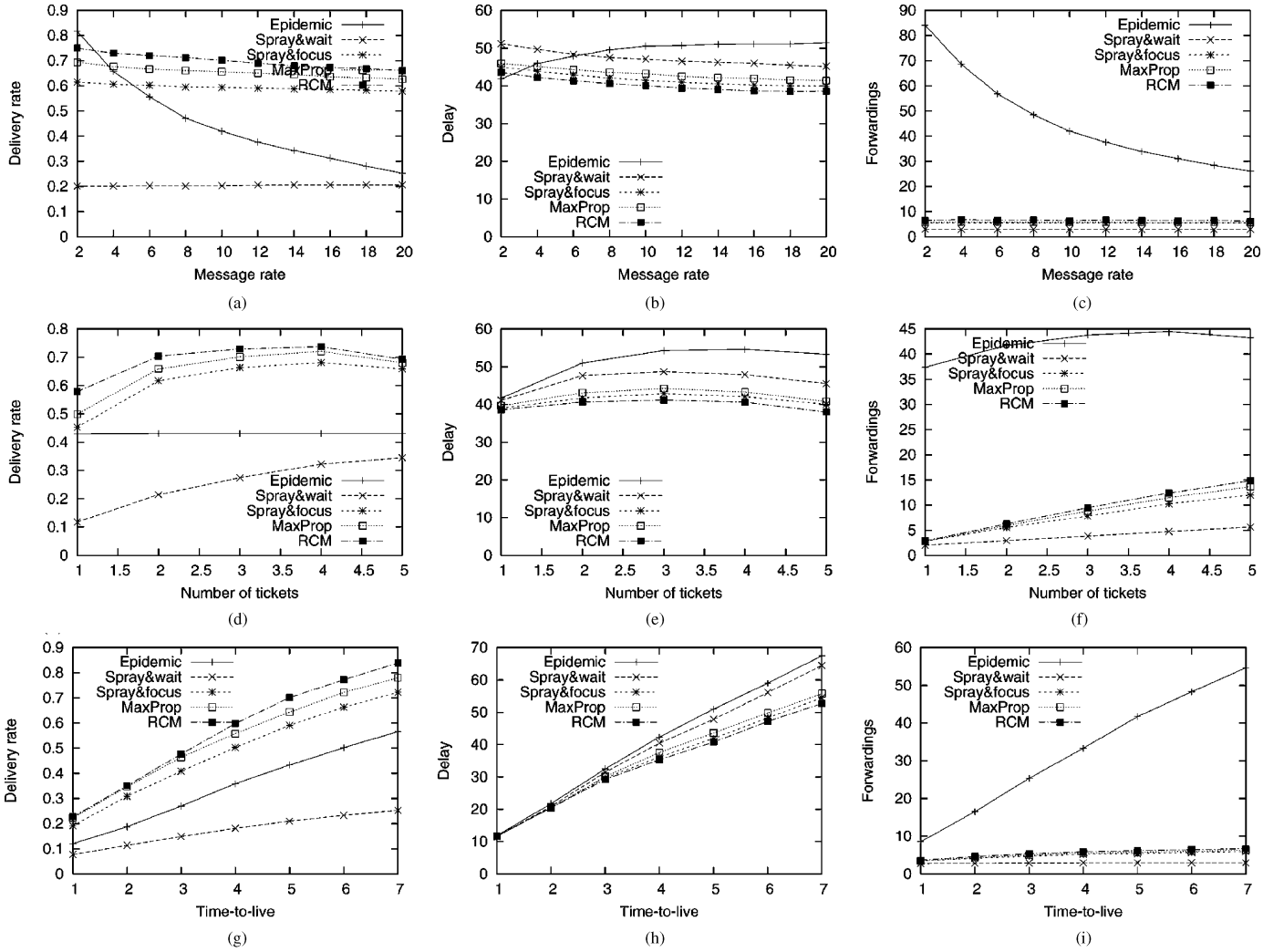


Fig. 11. Results in the UMass DieselNet traces. (a) Message rate versus delivery rate. (b) Message rate versus delay. (c) Message rate versus forwardings. (d) Number of tickets versus delivery rate. (e) Number of tickets versus delay. (f) Number of tickets versus forwardings. (g) Time to live versus delivery rate. (h) Time to live versus delay. (i) Time to live versus forwardings.

attendance rate, message time to live, and number of tickets increase, which causes the increase of forwarding opportunity in the network. The delivery rate of RCM is larger than all other routing protocols in all settings. In the best cases, RCM is approximately 40% larger than MaxProp* (the second best) and is 250% larger than Spray&focus*. These results show the efficiency of our time-varying delivery probabilistic metric. In Fig. 10(j), the delivery rates of all protocols start to drop after the number of nodes reaches a certain value, which is because the network diameter also increases as the number of nodes increases.

The delays of the protocols are compared in Fig. 10(b), (e), (h), (k), and (n). All delay curves show reversed trends compared to curves of the delivery rate, except for Fig. 10(h), which is because we only consider the delay of delivered messages, and more messages with larger delays are delivered as the time to live of the messages increases. Again, the delay of RCM is smaller than all other routing protocols in all settings. In the best cases, RCM is approximately 10% smaller than MaxProp* (the second best) and is 15% smaller than Spray&focus*.

The number of forwardings (overhead) of the protocols are compared in Fig. 10(c), (f), (i), (l), and (o). On average, RCM forwards 20% more messages than MaxProp* and 50% more than Spray&focus*. However, compared to the 40% and 250% increases in delivery rate, the additional overhead can be justified. Note that, for the same routing protocol, 20% more message forwardings cannot result in a 20% increase in delivery rate. The additional forwardings in RCM might be a result of the time-varying forwarding metrics: The order of the EMD values of a message in two nodes might reverse at different times.

UMass DieselNet trace. Next, we show our simulation results in the UMassDieselNet trace. The delivery rates of the five compared forwarding algorithms are compared in Fig. 11(a), (d), and (g) with different message creation rates, number of initial tickets per message, and message time to live. The delivery rates of all protocols decrease as message rate increases and increase as message time to live increases. In Fig. 11(d), the delivery rates first increase and then decrease as the number of message tickets increases. The increases are due to the increase in the number of copies per message as a result of the increase of message tickets, and creating message copies occu-

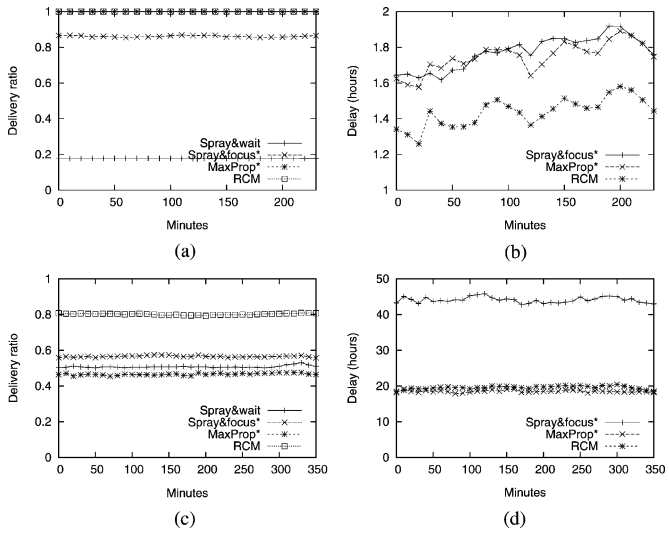


Fig. 12. Results in our synthetic bus traces. (a) Delivery rate (Miami). (b) Delay (Miami). (c) Delivery rate (Madrid). (d) Delay (Madrid).

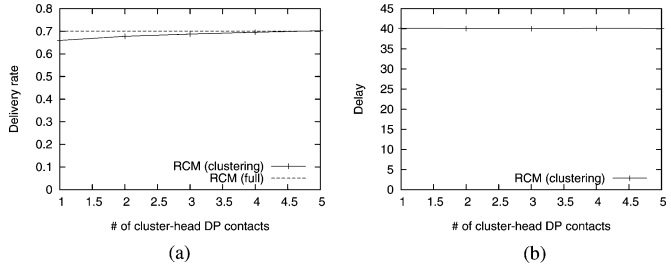


Fig. 13. Results with different level of reduced contact information. (a) Delivery rate. (b) Delay.

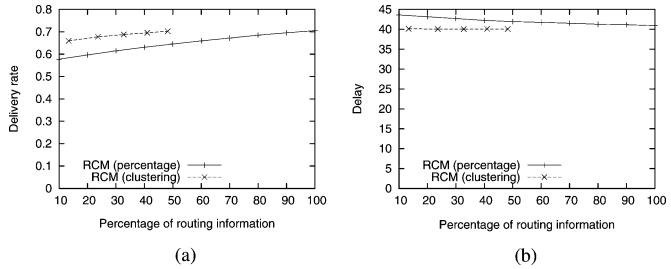


Fig. 14. Results with reduced contact information. (a) Delivery rate. (b) Delay.

pies the limited forwarding opportunity used by the forwarding algorithms. The delivery rate of RCM is larger than all other routing protocols in all settings, except being smaller than Epidemic at extremely low message rates. In the best cases, RCM is approximately 10% larger than MaxProp* and is 20% larger than Spray&focus*. Compared to the NUS student contact trace, the relatively insignificant improvement is because the contact pattern in the UMass DieselNet trace does not show a very high degree of regularity.

The delays of the forwarding algorithms are compared in Fig. 11(b), (e), and (h). All results on delays, except those for Epidemic, follows the trend of the delivery rates. Again, RCM has the smallest delay across all settings. The number of forwardings of the forwarding algorithms are compared in Fig. 11(c), (f), and (i). The results show that the number of RCM forwardings is very close to those of MaxProp* and Spray&focus*.

Synthetic Bus Trace. We show the simulation results of our synthetic bus traces that are generated using the Miami metro map and the Madrid metro map. There are differences between these two synthetic bus traces due to the geographic nature of the bus trace and the number of bus routes. In the Miami trace, there are a fewer number of buses, and contacts between buses are less regular.

As shown in Fig. 12(a) and (b), in the Miami traces, RCM and MaxProp* deliver all messages, while Spray&focus* has a delivery rate of 85% and Spray&wait has the lowest delivery rate, as expected. While delivering the same number of messages, RCM has a 20% smaller delay than MaxProp*. In this simulation, MaxProp* is better than Spray&focus* because, while delivering 15% more messages, MaxProp* has an equal delay as Spray&focus*.

On the Madrid traces, which are much larger, RCM has a significant (more than 40%) improvement in delivery rate over all other protocols, where the best of them, *Spray&focus*, delivers only 10% more messages than the worst, MaxProp*. The delay of all protocols, except RCM and MaxProp*, are almost equal. The delay of RCM is 55% less than that of Spray&focus* and only about 40% of that of MaxProp*. In this simulation result, RCM is clearly superior over other protocols for having both the highest delivery rate and the smallest delay.

F. Simulation Results With Reduced Routing Information

This section presents simulation with partial routing information. We perform simulation with reduced contact information in the UMass DieselNet trace, in which each node still collects contact information from all nodes, but the amount of contact information sent from each node contains a constant number k of contacts, where k ranges from 1 to 5. Since all the four clustering methods have similar results, we only show the results when using the maximum-distance clustering.

As shown in Fig. 13(a), the delivery rate of contact clustering is very close to that with full contact information when $k = 3$. The delivery rate when $k = 5$ is about 99% of that with full information. Even when $k = 1$, the delivery rate is still above 90% of that with full information. Fig. 13(b) shows that the delay is hardly affected by the reduced contact information. However, it decreases very slowly due to the decreased delivery rate.

To highlight the efficiency of our contact clustering method, we perform a simulation to compare it to another routing information reduction method, which simply drops contact information randomly. As shown in Fig. 14(a) and (b), our contact information reduction method using maximum distance clustering is by far better than the random contact information removing method.

G. Summary of Simulation

To sum up, RCM outperforms the compared routing protocols with historical connectivity information, in terms of both delivery rate and delay. As shown by simulation results, RCM has more improvement over other protocols when the contact pattern between nodes are more regular, as in the NUS student trace and our synthetic bus trace, where both buses and bus

stations are nodes. RCM also shows an increasing relative improvement in its performance when the size of the network and the complexity in the contact pattern increases. Finally, we show that RCM keeps a superior routing performance using reduced routing information, which improves the scalability of RCM in practical situations.

VI. Related Work

In [7], Jain *et al.* presented a comprehensive investigation on the DTN routing problem with different levels of prior knowledge about the network. Specifically, Dijkstra's algorithm (with future connectivity information) or the linear programming approach with information of future connectivity, traffic demands, etc., is used to obtain an optimal path between a source and a destination. In [8], Merugu *et al.* proposed a DTN routing algorithm that is similar in spirit to Dijkstra's algorithm in [7]. In [14], Liu and Wu propose a model for DTNs with repetitive mobility. A hierarchical routing is further proposed to make routing in such DTN models scalable.

Among the approaches in deterministic routing, [29], [30], and [31] exploited deterministic trajectories of mobile nodes to help deliver data, improve data delivery performance, and reduce energy consumption in nodes. In [32], Wu *et al.* used semideterministic trajectories of mobile nodes to achieve deterministic results of several routing schemes. Such a trajectory is selected from a set of predefined hierarchical structured routes. Throwboxes [33], [28], which are stationary, battery-powered nodes with storage and processing, are proposed to enhance the capacity of DTNs.

Epidemic routing [21] is the first flooding-based routing algorithm. Gossip [9] forwards with probability p. Opportunistic routings, such as [10], forward messages based on the delivery probability. Different delivery probability metrics are proposed, including encounter frequency [10], time elapsed since last encounter [5], [23], [24], [34]–[36], location similarity [13], expected delay [37], social similarity [11], [38], and geometric distance [25]. Delegation forwarding [26] uses forwarding thresholds to reduce the total number of copies of each message in the network.

A cyclic model was used as a predictable connectivity model in [6], where sensor nodes are put to sleep most of the time and wake up periodically to save energy and reduce contention. The observed connectivity is used to predict future connectivity in order to facilitate successful communication. While some heuristic forwarding algorithms are proposed in [6], we applied MDP to derive optimal forwarding metrics. Our paper is also the first one to use a cyclic probability metric to optimize routing performance in delay-tolerant networks. A MobySpace [13] is a Euclidean space (or other higher dimensional space) where nodes can be either mobile or static and they can communicate within each other's transmission range.

Trace data available to the research community [27] includes the UMass DieselNet trace, the NUS student contact trace, the Haggle project [39], and the MIT Reality Mining [40]. In [40], several opportunistic routing algorithms are simulated in large realistic contact traces. A timely contact probability metric is proposed in this paper, which captures the contact frequency of mobile nodes and is similar to [10] and [5] in spirit.

VII. Conclusion

In this paper, we present the first research investigating a new cyclic delivery probability metric, expected minimum delay (EMD), and provide methods to achieve it in a cyclic MobiSpace. The proposed probabilistic routing algorithm, routing in cyclic MobiSpace (RCM), is evaluated and compared to the enhanced versions of some existing DTN routing protocols using the NUS student trace, the UMass DieselNet trace, and our synthetic traces. Simulation results demonstrate that RCM outperforms the compared protocols in terms of delivery rate and delay. In the future, we will create more synthetic traces with different metro maps and use analysis to generalize the conditions where RCM can provide the most significant improvement.

References

[1] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-tolerant network architecture," DTN Research Group, Internet Draft: Draft-irrf-dtnrg-arch.txt, 2006.

[2] V. Srinivasan, M. Motani, and W. T. Ooi, "Analysis and implications of student contact patterns derived from campus schedules," in *Proc. ACM MobiCom*, 2006, pp. 86–97.

[3] M. Motani, V. Srinivasan, and P. Nuggenhalli, "PeopleNet: Engineering a wireless virtual social network," in *Proc. ACM MobiCom*, 2005, pp. 243–257.

[4] A. Chaintreauy, P. Hu, J. Crowcroft, C. Dioty, R. Gassy, and J. Scotty, "Pocket switched networks: Real-world mobility and its consequences for opportunistic forwarding," Tech. Rep. UCAM-CL-TR-617, 2005.

[5] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "MaxProp: Routing for vehicle-based disruption-tolerant networking," in *Proc. IEEE INFOCOM*, 2006, pp. 1–11.

[6] Y. Gu and T. He, "Data forwarding in extremely low duty-cycle sensor networks with unreliable communication links," in *Proc. ACM SenSys*, 2007, pp. 321–334.

[7] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *Proc. ACM SIGCOMM*, 2004, pp. 145–158.

[8] S. Merugu, M. Ammar, and E. Zegura, "Routing in space and time in network with predictable mobility." College of Computing, Georgia Tech, Atlanta, Tech. rep. GIT-CC-04-07, 2004.

[9] J. Haas, J. Y. Halpern, and L. Li, "Gossip-based ad hoc routing," in *Proc. IEEE INFOCOM*, 2002, vol. 3, pp. 1707–1716.

[10] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," *Lecture Notes Comput. Sci.*, vol. 3126, pp. 239–254, Aug. 2004.

[11] D. Elizabeth and H. Mads, "Social network analysis for routing in disconnected delay-tolerant MANETs," in *Proc. ACM MobiHoc*, 2007, pp. 32–40.

[12] C. Liu and J. Wu, "Routing in a cyclic mobispace," in *Proc. ACM MobiHoc*, 2008, pp. 351–360.

[13] J. Leguay, T. Friedman, and V. Conan, "MobySpace: Mobility pattern space routing for DTNs," presented at the ACM SIGCOMM, 2005.

[14] C. Liu and J. Wu, "Scalable routing in delay tolerant networks," in *Proc. ACM MobiHoc*, 2007, pp. 51–60.

[15] X. Zhang, J. F. Kurose, B. Levine, D. Towsley, and H. Zhang, "Study of a bus-based disruption tolerant network: Mobility modeling and impact on routing," in *Proc. ACM MobiCom*, 2007, pp. 195–206.

[16] R. A. Howard, *Dynamic Programming and Markov Processes*. Cambridge, MA: MIT Press, 1960.

[17] B. Bonet and H. Geffner, "Faster heuristic search algorithms for planning with uncertainty and full feedback," in *Proc. IJCAI*, 2003, pp. 1233–1238.

[18] A. J. Briggs, C. Detweiler, D. Scharstein, and A. Vandenberg-Rodes, "Expected shortest paths for landmark-based robot navigation," *Int. J. Robot. Res.*, vol. 23, no. 7–8, pp. 717–728, Jul.–Aug. 2004.

[19] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer, 2009.

[20] J. A. Hartigan, *Clustering Algorithms*. New York: Wiley, 1975.

[21] A. Vahdate and D. Becker, "Epidemic routing for partially-connected ad hoc networks," Duke Univ., Durham, NC, Tech. Rep., 2002.

[22] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks," in *Proc. ACM WDTN*, 2005, pp. 252–259.

[23] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility," in *Proc. IEEE PerCom*, 2007, pp. 79–85.
[24] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "DTN routing as a resource allocation problem," in *Proc. ACM SIGCOMM*, 2007, pp. 373–384.
[25] U. Acer, S. Kalyanaraman, and A. Abouzeid, "Weak state routing for large scale dynamic networks," in *Proc. ACM MobiCom*, 2007, pp. 290–301.
[26] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot, "Delegation forwarding," in *Proc. ACM MobiHoc*, 2008, pp. 251–260.
[27] "CRAWDAD data set," Dartmouth Univ., Hanover, NH [Online]. Available: http://crawdad.cs.dartmouth.edu
[28] T. Small and Z. Haas, "The shared wireless infostation model: A new ad hoc networking paradigm (or where there is a whale, there is a way)," in *Proc. ACM MobiHoc*, 2003, pp. 233–244.
[29] W. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," in *Proc. ACM MobiHoc*, 2004, pp. 187–198.
[30] M. M. B. Tariq, M. Ammar, and E. Zegura, "Message ferry route design for sparse ad hoc networks with mobile nodes," in *Proc. ACM MobiHoc*, 2006, pp. 37–48.
[31] B. Burns, O. Brock, and B. N. Leivine, "MORA routing and capacity building in disruption-tolerant networks," *Ad Hoc Netw.*, vol. 6, no. 4, pp. 600–620, Jun. 2008.
[32] J. Wu, S. Yang, and F. Dai, "Logarithmic store–carry–forward routing in mobile ad hoc networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 6, pp. 735–748, Jun. 2007.
[33] N. Banerjee, M. D. Corner, and B. N. Levine, "Design and field experimentation of an energy-efficient architecture for DTN throwboxes," *IEEE/ACM Trans. Netw.*, vol. 18, no. 2, pp. 554–567, Apr. 2010.
[34] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, "Age matters: Efficient route discovery in mobile ad hoc networks using encounter ages," in *Proc. ACM MobiHoc*, 2003, pp. 257–266.
[35] M. Grossglauser and M. Vetterli, "Locating nodes with ease: Last encounter routing in ad hoc networks through mobility diffusion," in *Proc. IEEE INFOCOM*, 2003, vol. 3, pp. 1954–1964.
[36] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "Replication routing in DTNs: A resource allocation approach," *IEEE/ACM Trans. Netw.*, vol. 18, no. 2, pp. 596–609, Apr. 2010.
[37] M. Demmer and K. Fall, "DTLSR: Delay tolerant routing for developing regions," in *Proc. ACM NSDR*, 2007, Article no. 5.
[38] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay tolerant networks," in *Proc. ACM MobiHoc*, 2008, pp. 241–250.
[39] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD data set Cambridge/Haggle (v. 2006-09-15)," Sep. 2006 [Online]. Available: http://crawdad.cs.dartmouth.edu/cambridge/haggle
[40] L. Song and D. F. Kotz, "Evaluating opportunistic routing protocols with large realistic contact traces," in *Proc. ACM CHANTS*, 2007, pp. 35–42.

Cong Liu received the B.S. degree in micro-electronics from South China University of Technology, Guangzhou, China, in 2002; the M.S. degree in computer software and theory from Sun Yat-sen (Zhongshan) University, Guangzhou, China, in 2005; and the Ph.D. degree in computer science from Florida Atlantic University, Boca Raton, in 2009.

He is an Assistant Professor with the Department of Computer Science, Sun Yat-sen University. He has served as a reviewer for many conferences and journal papers. His main research interests include routing in mobile ad hoc networks (MANETs) and delay-tolerant networks (DTNs), deep packet inspection, and transaction processing.

Jie Wu (F'09) received the B.S. degree in computer engineering and the M.S. degree in computer science from Shanghai University of Science and Technology (now Shanghai University), Shanghai, China, in 1982 and 1985, respectively, and the Ph.D. degree in computer engineering from Florida Atlantic University, Boca Raton, in 1989.

He is a Chair and a Professor with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA. Prior to joining Temple University, he was a Program Director with the National Science Foundation, Arlington, VA. He has published more than 500 papers in various journals and conference proceedings. His research interests include wireless networks and mobile computing, routing protocols, fault-tolerant computing, and interconnection networks.

Dr. Wu has served as an IEEE Computer Society Distinguished Visitor and is the Chairman of the IEEE Technical Committee on Distributed Processing (TCDP). He serves on the Editorial Board of the IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON MOBILE COMPUTING, and the *Journal of Parallel and Distributed Computing*. He is Program Co-Chair for IEEE INFOCOM 2011. He was also General Co-Chair for IEEE MASS 2006, IEEE IPDPS 2008, and DCOSS 2009.