Routing in a Cyclic MobiSpace *

Cong Liu and Jie Wu
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431
{cliu8@, jie@cse}.fau.edu

ABSTRACT

A key challenge of routing in delay tolerant networks (DTNs) is to find routes that have high delivery rates and low end-to-end delays. When oracles are not available for future connectivity, 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, but no prior research work has investigated the cyclic delivery probability of messages between nodes. In this paper, we propose to use the *expected minimum delay* (EMD) as a new delivery probability metric in DTNs with repetitive but non-deterministic mobility. Specifically, 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. Finally, we apply the *Markov decision process* to derive the EMDs of the messages at particular times. Our proposed EMD-based routing protocol, called *routing in cyclic MobiSpace* (RCM), outperforms several existing opportunistic routing protocols when simulated using both real and synthetic traces.

Keywords

Cyclic MobiSpace, delay tolerant networks (DTNs), simulation, trace.

1. INTRODUCTION

Delay tolerant networks (DTNs) are occasionally-connected networks that suffer from frequent network partitioning as the result of high mobility, low density, short radio range, intermittent power, interference, obstruction, attacks, etc. Representative DTNs include sensor networks using scheduled connectivity, terrestrial wireless networks that cannot ordinarily maintain end-to-end connectivity, satellite networks with periodic connectivity, and underwater acoustic networks with moderate delays and frequent interruptions.

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. DTNs, in contrast, 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 nature of network connectivity, which significantly complicates the routing decision. In [2, 3], optimal delivery paths in a DTN can be discovered by constructing a time-space graph with oracles. In practical situations where no oracle is available to reveal future contacts, opportunistic routing [4, 5] is proposed in which one or more copies of a message is 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 life-spans 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 [5] and social similarity [6]. One advantage of the 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 the nodes, avoiding expensive and frequent updates. We notice that most real objects have cyclic motion patterns, and therefore it is possible and valuable in practice to use some cyclic metric to increase the accuracy of the estimated delivery probability. In this paper, we propose the use of a cyclic long term metric, called the *expected minimum delay* (EMD), which is the expected time that an optimal forwarding scheme takes to deliver a message at a specific time from a source to a destination, in a network with cyclic and uncertain connectivity.

A MobiSpace 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

*This work was supported in part by NSF grants ANI 0073736, EIA 0130806, CCR 0329741, CNS 0422762, CNS 0434533, CNS 0531410, and CNS 0626240.

MobiHoc'08, May 26–30, 2008, Hong Kong SAR, China.

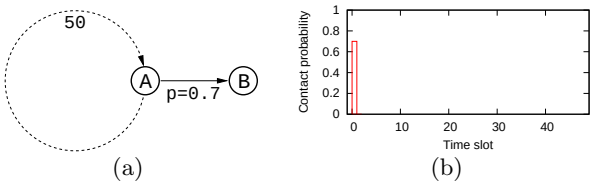


Figure 1: Nodes A and B have a contact with $p = 0.7$ during time slot 0 every 50 time slots.

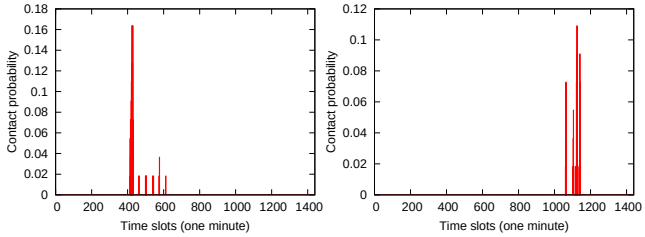


Figure 2: Discretized probabilistic contacts between different sub-shifts in the UMassDieselNet trace.

the probability that they will be in contact at the same time in the next cycle is high. A cyclic MobiSpace can be modeled with a *probabilistic time-space graph* in which an edge between two nodes contains a set of *discretized probabilistic contacts*. Each discretized probabilistic 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 EMD of the state corresponding to the node hosting the message and the current time. With EMDs, we propose a routing protocol, called *routing in cyclic MobiSpace* (RCM).

The contributions of this paper are summarized as follows.

1. We propose the use of a cyclic, long-term delivery probability metric, called EMD, to improve the performance of opportunistic routing in cyclic MobiSpace.
2. We model a cyclic MobiSpace as a probabilistic time-space graph and to remove the time dimension, we devise a lossless translation of this graph model into a probabilistic state-space graph model.
3. We apply the Markov decision process (MDP) to solve the EMDs of the states in our probabilistic state-space graph model. We also extend a recent result on solving MDPs and propose an efficient algorithm for calculating the EMDs in our state-space graph model.
4. We generate synthetic bus traces mimicking the real traces, which will be available to the research community for subsequent research efforts in DTNs.
5. We perform simulations to evaluate and compare RCM with the enhanced versions of some existing protocols, using the NUS student trace, the UMassDieselNet trace, and our synthetic traces. Simulation results show that RCM outperforms the other protocols.

2. 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 so can be the contact probability distribution between two buses. Contrary to a cyclic MobiSpace 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 cycle of most objects are based on human defined or natural cycles of time such as hour, day, week, etc; (2) most objects' motions are repetitive, time-sensitive, and location-related. Note that cyclic MobiSpace is defined without any assumption on the shapes of the trajectories, nor are the trajectories required to be functions of time as in [7].

In this section, we will first model a cyclic MobiSpace as a probabilistic time-space graph, from which we will generate a probabilistic state-space graph, which is a probabilistic graph without time-dependent edges. Our goal is to calculate the EMD of each message from this probabilistic graph, for which we provide an extended efficient algorithm.

2.1 Expected minimum delay (EMD)

The EMD is the expected time an optimal opportunistic forwarding scheme takes to deliver a message given a starting time and a source-destination pair. This optimal scheme always maximizes the delivery probability of each message by forwarding a single copy of the message in the network. As shown in Figure 1(a), assume node A travels a circular trajectory once every 50 time slots, and only in time slot 1 (Figure 1(a)), it has a contact with static node B with probability 0.7. The contact probability between A and B in any 50 time slots is shown in Figure 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 in time slot 0 from A to B can be calculated as follows: $D_0 = 0.7 \times 0.1 + 0.3 \times (50 + D_0)$. Clearly, in any time slot t, the EMD is $D_t = D_0 + (50 - t \mod 50) \mod 50$. We introduce EMD as a new, accurate delivery probability metric because it reflects the time-varying delivery probability between each pair of nodes accurately during each cycle.

2.2 Discretized probabilistic contact

We divide the common motion cycle T into small, fixed time slots. For each pair of nodes, we introduce a set of conceptual *discretized probabilistic contacts*. A discretized probabilistic contact 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.

In the following, we show an example of the discretized probabilistic contact generated from 55 days of contact traces in the UMassDieselNet trace [8, 9], where we consider each

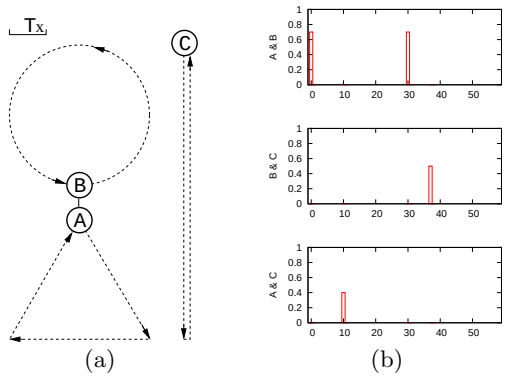


Figure 3: A physical cyclic MobiSpace (a). The discretized probabilistic contacts between each pair of nodes of a common motion cycle T (b).

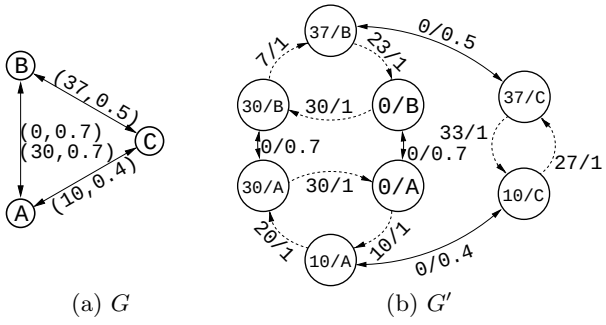


Figure 4: The time-space graph G and the state-space graph G' of the cyclic MobiSpace in Figure 3(a).

sub-shift (see Section 4.3) as a node, one day as the common motion cycle, and one minute as the unit of a time slot. The discretized probabilistic contacts between sub-shifts 01/AM (the morning sub-shift of shift number 1) and 03/AM are shown in Figure 2(a). Figure 2(b) shows those between another pair of sub-shifts.

From these figures, we can see that the discretized probabilistic contacts between two nodes in a realistic cyclic MobiSpace gather around only a few consecutive time slots, and the number of discretized probabilistic contacts is much smaller than the total number of time slots. Note that the choice of time slot size is a trade-off between the accuracy[1] of EMDs and the computation time.

2.3 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. An edge between two nodes is a non-empty set of discretized probabilistic contacts between the two nodes.

Figure 3(a) shows a sample cyclic MobiSpace that we will use throughout this paper. In this figure, nodes A and B move in their circular and triangular 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 non-deterministic in nature due to uncertainty of the nodes' positions, communication failures, etc. The set of discretized probabilistic contacts for each pair of the nodes in this cyclic MobiSpace is shown in Figure 3(b).

The time-space graph G of the network is shown in Figure 4(a). In G, each edge contains the set of discretized probabilistic contacts between its nodes. For example, edge (A, B) contains two discretized probabilistic contacts. One is labeled $(0, 0.7)$, which means its time slot is 0 and contact probability is 0.7.

2.4 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 which 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 discretized probabilistic contacts. For example, three states $0/A$, $10/A$, and $30/A$ are created in G' (Figure 4(b)) for node A in G (Figure 4(a)), since A has three discretized probabilistic contacts $(0, 0.7)$, $(10, 0.4)$, and $(30, 0.7)$ at different time slots. If node u has more than one contact (with different nodes) in the same time slot, then only one state is created for u corresponding to this time slot in G'.

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 to form a ring by three directional links represented by dashed lines as shown in Figure 4(b). The bidirectional link in G' is created corresponding to each discretized probabilistic contact in G. For each discretized probabilistic 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 Figure 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 at node A in 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. On the other hand, 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 message can always be kept in a node without being forwarded. For a bidirectional link, p^{max} is equal to the contact probability of the corresponding discretized probabilistic contact. This is because the forwarding probability of a forwarding node cannot exceed its contact probability with the receiving node. For simplicity, we let $d = 0$ for all bidirectional links assuming that message forwarding is always restricted to a time slot. This assumption is based on the fact that the delay of a message forwarding is generally much smaller than the size of the time slot itself.

[1]Information about the ordering of the physical contacts within the same time slot is lost.

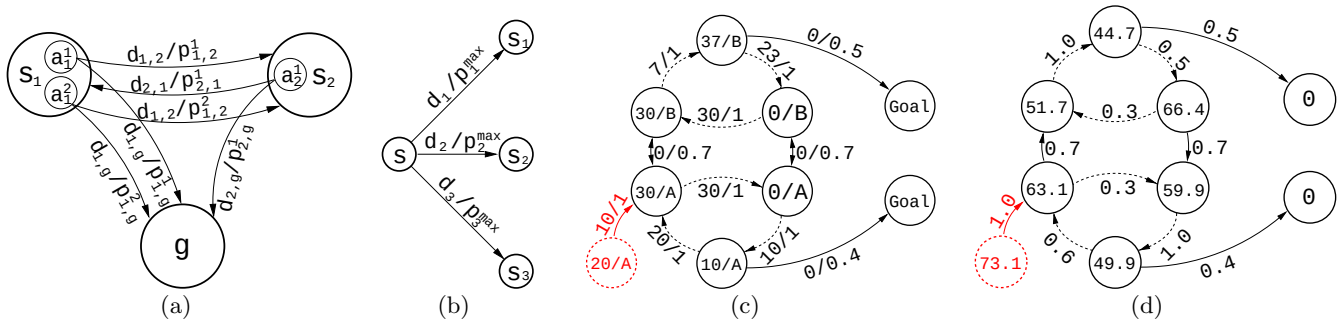


Figure 5: (a) A Markov decision process (MDP) model (b) Implicit action (c) The MDP of G' in Figure 4(b) with C as destination (d) The values and optimal actions after applying value iteration in Figure 5(c).

2.5 Remarks

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

Our probabilistic graph models differ from the models defined in previous papers [2, 3] in that a contact in the previous models is a deterministic connection during a specific period of time, while a discretized probabilistic contact is conceptual and is drawn from the 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 can be calculated. Note that the translation from G to G' is lossless since G can be reproduced from G' by combining the states of the every node.

The number of states in G' depends on the number of nodes and the discretized probabilistic contacts of each node which is bounded by the number of time slots in T. Methods to reduce the number of states includes increasing the size of time slots and dropping the discretized probabilistic contacts whose probabilities are below a certain threshold.

3. EXPECTED MINIMUM DELAYS IN A STATE-SPACE GRAPH

This section reviews and reformulates a variation of the Markov decision process (MDP). Using MDP, the values associated with the states in G' are updated iteratively and finally converge to their EMDs. An efficient approach is also proposed to calculate EMDs by extending a recent work in solving MDPs.

3.1 Markov Decision Process (MDP)

State-space searching is a very common problem in AI planning (or decision-making), which is similar to routing. The Markov decision process (MDP) [10, 11] provides a mathematical framework for modeling decision-making in situations where outcomes are partly random and partly under the 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 explained below. At any given time, the system can be at only one state s in the set of all states S. Each state s has a set of actions $A_s \subseteq A$ (the set of all actions). 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. Note that for any s and a, $\sum_{s' \in S} T_a(s, s') = 1$. The delay function $D(s, s')$[2] specifies the delay of transiting from state s to state s'. A value function $V(s)$, gives the expected minimum total delay for transiting from s to any goal state $g \in S_G$ (the set of all goal states). $V(g) = 0$. An MDP is illustrated in Figure 5(a), where s_1, s_2, and $g \in S_G$ are different states. s_1 has two actions a_1^1 and a_1^2. If a_1^1 takes effect, the system will transit from s_1 to state s_2 with delay $d_{1,2}$ and probability $p_{1,2}^1$.

The MDP problem is to find the set of expected minimum total delay values $V(s)$ for each state s to reach any $g \in S_G$ and the corresponding optimal action chosen at each state in order to achieve the minimum values. Value iteration [12] solves MDPs by iteratively updating the value functions (Equation 1) of every state until they converge. In each round of the iteration, based on the resulting values in the last iteration, the value $V(s)$ of each state $s \notin S_G$ is updated by choosing an action $a \in A_s$ such that $V(s)$ is minimized. For an action a, the value of s is the sum of the delays $D(s, s') + V_t(s')$ (for each possible next state s') are weighed by $T_a(s, s')$ (refer to Figure 5(a)).

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

The value functions are considered to converge sufficiently when the maximum difference between the values of all states in two consecutive iterations is less than some threshold value. When the values of the states are properly initialized (such as iteratively applying Equation 2, which is essentially the Bellman-Ford algorithm), the value iteration is guaranteed to converge with the minimum values in the states [13].

[2]In the traditional MDP models, cost function C is used. We change it to delay function D in the networking context.

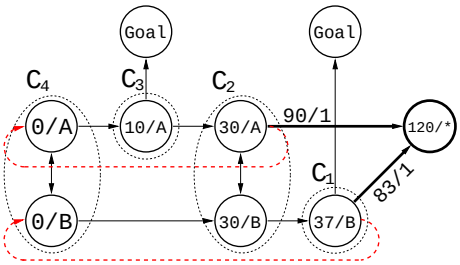


Figure 6: An efficient approach for the MDP in Figure 5(c).

Table 1: Comparison of value iteration (VI) and our extended topological value iteration (E-TVI).

(a) Resulting value.

states	0/A	0/B	10/A	30/A	30/B	37/B
VI	59.87	66.41	49.88	63.14	51.70	44.70
E-TVI	59.92	66.46	49.91	63.19	51.73	44.73

(b) Computation time.

nodes	100	200	300	400	500
VI	0.515	1.256	3.515	7.194	10.25
E-TVI	0.137	0.360	0.668	1.225	2.426

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

3.2 Deriving EMDs using MDP

Our probabilistic time-space graph model G' differs from the MDP model in that the states in G' are not associated with actions which tell the actual state transition probabilities. However, implicit actions can be determined by the node's forwarding preferences. A forwarding preference shows the current node's decision on which node to forward the message to (or to keep the message in the current node) when connected with multiple nodes. Inspired by a research work on robotics [14], we derive the optimal implicit action without enumerating all implicit actions.

Remember that each link in G' represents a possible state transition and it has a delay d, and a maximal transition probability p^{max} (see Section 2.4). In Figure 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 it 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 calculated in this way is always 1 since for each state s there is at least one bidirectional 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 can be obtained from the permutation of the preferential order of state transitions from this state. 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 only 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 is preferred over state s_2. 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' (Figure 4(b)), we modified G' 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 Figure 5(c). Applying value iteration on this MDP, we get the values (shown inside the states) and the optimal actions (labeled on the links) of each state in Figure 5(d). Note that in Figure 5(d), some of the links are removed such as the one from states $0/A$ to $0/B$. This shows the situations where a message is not forwarded from a node with a lower EMD to a node with a higher EMD.

3.3 An efficient approach for EMDs

In MDPs, if a state s is a successor state of s' (there is a transition from s' to s), then the value $V(s')$ is dependent on $V(s)$. This causal relation is transitive. In [13], topology value iteration (TVI) is used to reduce the number of rounds of updates performed to each state in the value iteration. In TVI, Kosaraju's [15] algorithm is used to find the set of strongly-connected components $\{C_i\}$ and their topological order. If component C_1 is a successor of C_2 (i.e., for any $u \in C_1$ and $v \in C_2$, u is a successor of v), then the values of the states in C_2 are dependent on those in C_1 (but not vice versa). That is, components C_1 and C_2 can be calculated separately in a reverse topological order. TVI significantly reduces the number of rounds in value iteration by breaking the computation of the whole graph into the computation of small components.

TVI cannot be directly applied to our graph model G' since in G' the non-goal states are likely to be connected as a strongly-connected component. As in Figure 5(c), (1) states of the same node are strongly-connected by directional links, and (2) states of different nodes are connected by bidirectional links. However, if we group the states of the same time slots into components (not necessarily strongly-connected components), we find that these components form a partial order if, given a time slot t_b, each link starting at a time slot before t_b and ending at time slot in or after t_b is removed. For example, if we change the layout of Figure 5(c), and remove the corresponding links by setting $t_b = 0$, we get the components $C_4, \ldots, C_1$ in a partial order as shown in Figure 6. In this figure, the set of removed links are shown by dashed lines which we denote as L_b.

Our extended TVI algorithm has two steps. In step one, the states are updated by TVI after modifying G' with the following steps: (a) we let $t_b = 0$ and remove L_b from the graph, (b) we add a state s^* with its time slot being $\frac{3}{2}T$, (c) for each state s which has an outgoing link removed (in set L_b), we add a link from s to s^*. Figure 6 shows an example where the state added is $120/*$ and the links added are shown by thick lines. The TVI is then applied to update the component in their reversed topological order,

i.e., $C_1, \ldots, C_4$. In step two of our extended TVI, we recover the original graph by removing the added state s^* and the added links, and adding back the links in L_t. Then, we continuously update the components in the same topological order, i.e., $C_1, \ldots, C_4$, until the values of states in all of these components converge. Similar to TVI, our extended TVI is efficient since it breaks the whole graph into a large number of small components and update them separately.

Table 1(a) compares the resulting values of the states in the MDP in Figure 6. We can see that the values computed by both methods are almost identical. Table 1(b) compares their computation time in networks with different number of nodes generated as described in Section 4.2.

3.4 Routing and analysis

We propose our opportunistic routing protocol called *routing in cyclic MobiSpace* (RCM), which uses EMD as the metric of delivery probability. Its forwarding rule is simple: (1) for the single-copy forwarding case, a node u always forwards message to node v that it encounters iff the message has a smaller EMD in v than in u at the current time; (2) for the multiple-copy forwarding case, tickets are allocated among nodes proportional to the reciprocal of their EMDs.

RCM uses our long-term metric EMD and it has a small amortized overhead for metadata (routing information), since it does not require frequent updates of metadata unlike algorithms that use short-term metrics. For example, for a DTN based 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 update.

The following theorems show the optimality of our algorithm. Proofs are simplified due to space limitation.

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

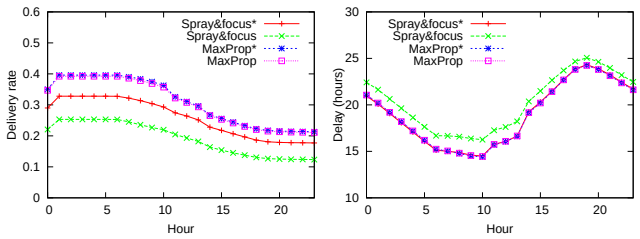
PROOF. The algorithms converge to EMDs because (1) the values are non-decreasing in each iteration, and (2) the values are minimum upper bounded by EMDs initially and in each iteration. □

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} ($D_{AG} > D_{BG}$) respectively. Then A should forward the message to B in order to maximize the EMD of the message if they meet. The assumption that there is a node C through which the message has a larger EMD than B is contradicted by $D_{BG} < D_{AG} \leq D_{ACG}$. □

4. SIMULATION



(a) Delivery rate. (b) Delay.

Figure 7: Performance comparison of existing protocols and their extended versions using the UMassDieselNet trace.

We evaluate our protocol, RCM, in the context of other routing algorithms using a wide variety of traces: NUS student contact trace, UMassDieselNet trace, and our synthetic bus traces. We will describe these network traces and the our simulation methods using these traces in respective subsections. Simulation results show that RCM improves the delivery rate and lowers the end-to-end delay in all traces as than the compared protocols.

4.1 Protocols in comparison

We compare RCM against several other approaches. These algorithms are based solely on deciding which messages to forward during a contact with a given peer. Most of these algorithms approximate delivery probability as the likelihood of the existence of a delivery path. For our focus on the efficiency of the delivery probability metrics and for fairness in the comparison, we use the enhanced versions of these protocols that make use of the same level of prior knowledge of historical connectivity patterns as RCM does.

Epidemic [16]. A node copies a message to every new node it encounters that has not got a copy already, until its copy of the message times out, or it is notified of delivery.

Spray&wait [17]. 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* are associated with each original message. A node i can only copy a message to another node j it encounters if the message in i owns $N > 1$ tickets. The new copy in j will have L tickets ($L = \lfloor N/2 \rfloor$), and $N - L$ tickets will remain with the message in i.

Spray&focus [18]. This is an extension of Spray&wait where a message with one ticket can still be forwarded from node i to j if the delivery probability of j to the destination is higher. A node with the later *last encounter time* with a destination has a higher delivery probability. Transitivity [5] is used to improve predictability in their mobility models. We use a enhanced, $Spray\&focus^*$, in which the average meeting interval drawn from the contact history is used to indicate delivery probability.

MaxProp [8]. 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 disseminate it to every node in the network. The delivery probability from a source to a destination is the total cost on their shortest path, where cost of each hop (i, j) is $1 - f_j^i$. In our simulation, we use a copy-controlled version using logical tickets, which differs from Spray&focus only in the metric of delivery probability. In our enhanced version $MaxProp^*$, we let $f_j^i / f_k^i = t_j^i / t_k^i$

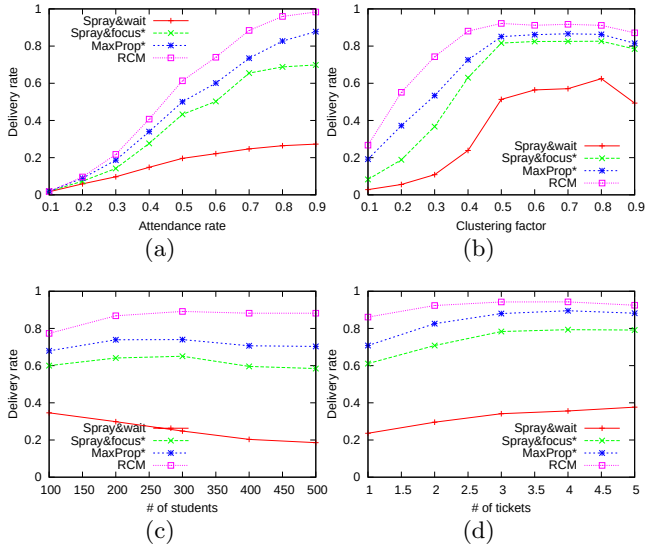


Figure 8: Comparison of delivery rate using the NUS trace.

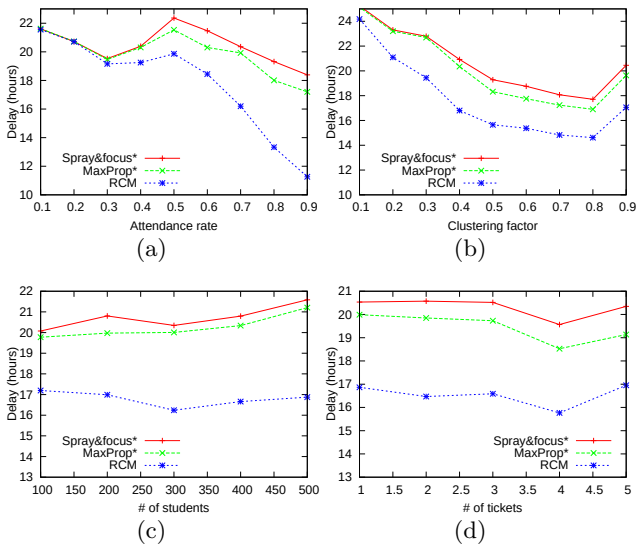


Figure 9: Comparison of delay using the NUS trace.

where t_j^i, t_k^i are the historical encounter times of i with j and k respectively.

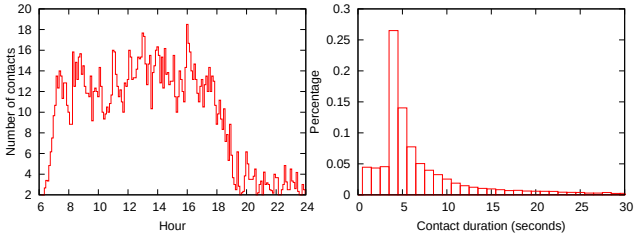
Note that all of the protocols that we implement aim to compare different metrics for delivery probability, and all other optimizations that have orthogonal effects on the performance of the protocols are not implemented. The orthogonality means that these optimizations can be added to all of our implemented algorithms and they are expected to provide an equal level of improvement in their routing performance. Such optimizations may include buffer management [8], global estimation of message delivery probability [19] and social centrality of the nodes [6], geometric information [20], etc. Similarly, we assume that an ideal mechanism serves to clear out buffers in the network of delivered data.

First, we conduct simulation studies to compare the performance of Spray&focus, MaxProp, and their extended versions using the UMassDieselNet trace. The simulation environment and settings will be the same as specified in the Section 4.3. As shown in Figure 7, $Spray\&focus^*$ is much better than $Spray\&focus$ in both delivery ratio and delay, while $MaxProp^*$ slightly outperforms $MaxProp$. Note that we use an ideal version of MaxProp where we assume each node knows the cost function f of the other nodes, which gives an upper performance bound. In practice, MaxProp has a much bigger amortized overhead in communication and computation than $MaxProp^*$ since the latter has a constant cost function f.

4.2 NUS student contact trace

Accurate information of human contact patterns is available in scenarios such as university campuses. As shown by the National University of Singapore (NUS) student contact trace model [21], when the 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 iff 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 the contact duration. (3) Only the contacts that take place during class hours are used. Non-class hours are removed to compress time. For example, suppose the last class session on Monday ends at 9pm, and the first session on Tuesday starts at 8am. If Monday 8pm to 9pm corresponds to the 10th hour, then Tuesday 8am to 9am is the 11th hour. 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 (several orders of magnitude larger than any reported real traces) over a long period. The schedules of the 4,885 classes and enrollment of 22,341 students for each of the class for each week of 77 class hours are publicly available on [22].

We select a number of students N ($100 \leq N \leq 500$) in each of experiment due to the memory constraint in the simulation environment. Contacts related to the non-selected students are ignored. We generate non-determinant traces by taking absentees into consideration. Each student attends a class with a attendance probability P_{attend}. Our data processing include the following steps. (1) We break each class session into several one hour class sessions and reassigned unique IDs to them. 159 conflicting enrollments are resolved by removing those enrollment with a lower class session ID. (2) The selection of N students is tricky. If selected randomly, the network becomes too sparse for messages to be delivered. On the other hand, when students are selected by maximizing their similarities (the number of common classes they are enrolled in), the network becomes over connected. 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 k^{th} student, we divide the $k-1^{th}$ selected students into two groups S_1 and S_2, and select the k^{th} student s as the one with the highest score $\sum_{s_1 \in S_1} sim(s, s_1) - \sum_{s_2 \in S_2} sim(s, s_1)$ among the stu-



(a) Contact distribution (b) Duration distribution

Figure 10: Statistics in the UMassDieselNet traces.

dents that are not yet selected, where the similarity function *sim* is defined as the number of common class sessions enrolled by two students. We define a clustering factor $C = |S_1|/(|S_1|+|S_2|)$ which determines the degree of connectivity in the network. (3) If two students enroll in the same class session, they have a discretized probabilistic contact in the hour of the class session with probability P^2_{absent}. With all of the discretized probabilistic contacts, the probabilistic state-space graph can be built. (4) Finally, we generate traces by generating contacts with probability P^2_{attend} for each pair of students and each class session they enroll in.

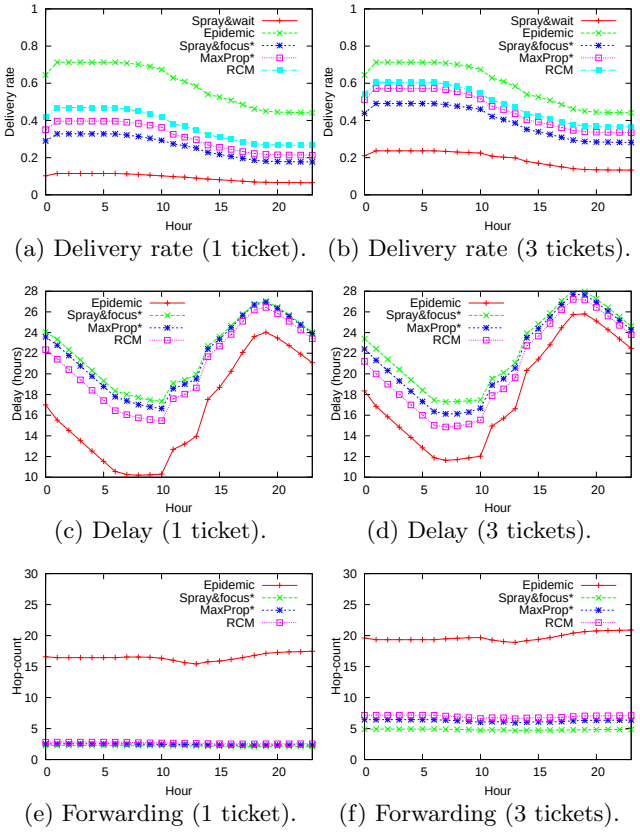
The default setting in our simulation is $P_{attend} = 0.8$, $C = 0.5$, 300 students, one ticket per message. In every minute, every node sends messages to 20 randomly selected nodes. As shown in Figures 8 and 9, for every protocols, the delivery rate increases and the delay decreases as P_{attend} increases. All protocols achieve their best performance when C is approximately 0.6, and the performance of the protocols improves as N decreases and L increases. The delivery rate of RCM is approximately 25% on average (and up to 50%) greater than that of the second best protocol $MaxProp^*$, and the delay of RCM is 15% on average (and up to 60%) smaller.

4.3 UMassDieselNet trace

Before presenting our simulation method and results, we give a brief description of the UMassDieselNet [8, 9] testbed and the traces collected on this testbed. We then describe how we pre-process these traces.

In the UMassDieselNet bus system consisting of 40 buses, the bus-to-bus contacts (the durations of which are relatively short) 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 ten routes. There are multiple shifts serving each of these routes. Shifts are further divided into morning (AM), midday (MID), afternoon (PM), and evening (EVE) sub-shifts. Drivers choose buses at random to run the AM sub-shifts. At the end of the AM sub-shift, the bus is often handed over to another driver to operate the next sub-shift 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 the contacts at a sub-shift level which do exhibit periodic behavior.

We obtain the sub-shift level contact by the following steps. Each sub-shift has a fixed starting time (TIME_AT_GARAGE) and a fixed ending time (DRVR_CHNG) everyday. We obtain a mapping from sub-shifts to these times by



(a) Delivery rate (1 ticket). (b) Delivery rate (3 tickets).

(c) Delay (1 ticket). (d) Delay (3 tickets).

(e) Forwarding (1 ticket). (f) Forwarding (3 tickets).

Figure 11: Comparison of different performance metrics using the UMassDieselNet trace.

parsing one of the dispatch records *DA_all.txt*. For example, sub-shift 21/AM (the AM sub-shift of shift 21) starts at 6:10AM and ends at 10:30AM. We also obtain a mapping from day and bus to the sub-shifts served by the bus on that day by parsing *DB_sheet.txt*. For example, on 3/1/2006, bus 3007 serves sub-shifts 21/AM, 21/MID, 21/PM, and 21/EVE. With the above two mappings, we translate 55 days of the bus-to-bus contacts into contacts between sub-shifts. A virtual contact is created between two sub-shifts if a bus is handed over from one sub-shift to another. Figures 10(a) and 10(b) show the distribution of all contacts over a day and the distribution of the contact duration at the sub-shift level.

The discretized probabilistic contacts between any pairs of shifts is then generated from the 55 days of sub-shift based contacts. Examples of the discretized probabilistic contacts between two pairs are shown in Figures 2(a) and 2(b). We set the time slot to be one minute. If in the trace of particular day, two sub-shifts have one or more contact during the same time slot as a discretized probability contact, then the contact probability of the discretized probability contact is increased by $\frac{1}{55}$. With the discretized probabilistic contacts, we can generate the probabilistic time-space graph and the probabilistic state-space graph for RCM, the inter-meeting time for $Spray\&focus^*$, and the next meeting probability f for $MaxProp^*$.

Messages are generated from every node (sub-shift) to ev-

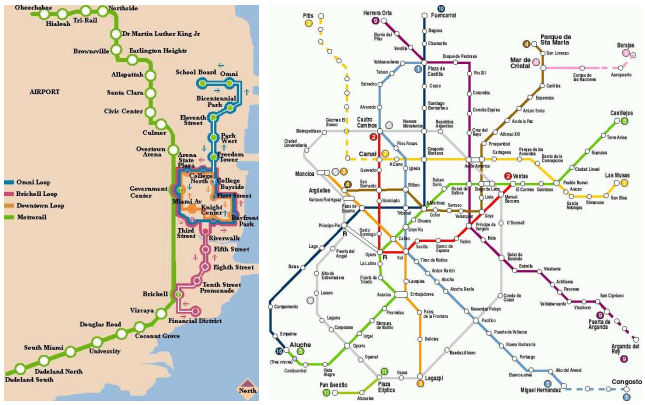


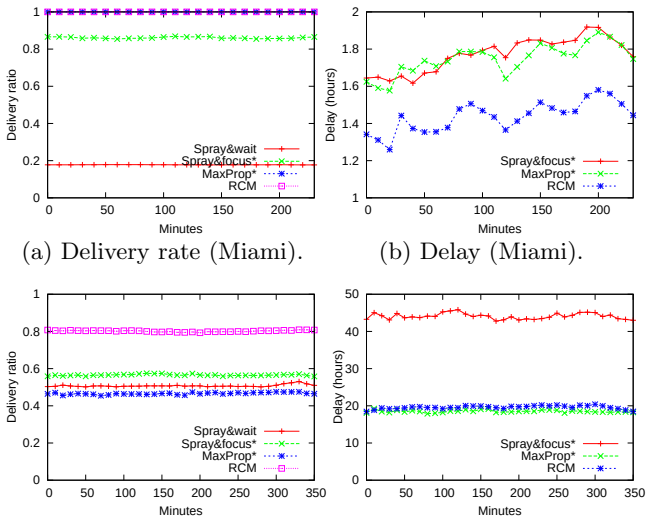
Figure 12: Our synthetic bus traces are generated from metro maps: Miami (left) and Madrid (right).

ery other nodes every 10 minutes. The TTL of each message is 3 days. In different group of simulations, messages are created with 1 ticket and 3 tickets respectively. Figures 11(a) to 11(b) show that RCM outperforms all other protocols except Epidemic in terms of delivery rate by approximately 5-15%. Similarly, Figure 11(c) and 11(d) shows that RCM outperforms all other protocols except Epidemic in terms of delay by approximately 5-10%. The insignificant improvement is because, the contact between the nodes are highly irregular in these traces. As we see in the simulation performed on our synthetic bus traces, the improvement is much more significant when bus stations are also considered as nodes. From Figures 11(e) and 11(f), we can see that RCM has a small overhead compared to Epidemic, but it is on par with all the other protocols. Note that in Figures 11(c) to 11(f) the delay and hop-count of the messages are averaged over the messages that are delivered by all protocols except *Spray&wait*.

4.4 Synthetic bus trace

In this section, we will compare RCM and the other protocols with two more synthetic bus traces we generated. Inspired by the NUS student contact trace, we generate two sets of synthetic bus traces from maps of the metros found in the Internet as shown in Figure 12. We develop a tool to extract the routes from the maps and then on each route we simulate a number of traveling buses.

Each bus travels a route starting from one of the station on the route. When several buses travel on the same route, their starting stations are disperse evenly along the route. If the stations of a route 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 minutes each bus takes to travel between two consecutive stations is in a Poisson distribution with mean being 5 minutes. The time that a bus stay at a station is 1 minutes. Buses are in contact with each other when they are in the same station. Unlike the UMassDieselNet, both stations and buses are nodes in our synthetic trace. We assign motion cycles to the buses on each route according to the number of station on that route. For the Miami map, the motion cycles for different routes are 60, 120, and 240 minutes. For the Madrid map, the motion cycles are 30, 120, 180, and 360 minutes. When a bus finish a round trip on its route but it takes a time less than the



(a) Delivery rate (Miami). (b) Delay (Miami).

(c) Delivery rate (Madrid). (d) Delay (Madrid).

Figure 13: Comparison of different performance metrics using our synthetic bus traces.

motion cycle of its route, the bus must stop in its first station for the rest of the time. These synthetic bus contact traces and more traces generated from the other maps in the future will be available for the research community.

As shown in Figures 13(a) and 13(b), in the Miami traces, RCM and $MaxProp^*$ deliver all the messages, while $Spray\&focus$ has a delivery rate of 85%. The delay of RCM is 15% less than those of $MaxProp^*$ and $Spray\&focus$. On the Madrid traces which is much larger, RCM has a significant (more than 40%) of improvement over all the other protocols, and the delay is only about 40% of that of $Spray\&focus$. Though the delay is a little larger than that of $MaxProp^*$, considered together with the delivery rate, RCM shows a clear superiority over the other protocols.

4.5 Summary of simulation

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

5. RELATED WORKS

In [2], Jain, Fall, and Patra 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 (when with information of future connectivity, traffic demands, etc.) is used to obtain an optimal path between a source and a destination. In [3], Merugu, Ammar, and Zegura proposed a DTN routing algorithm that is similar in spirit to Dijkstra's algo-

rithm in [2]. In [7], 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.

Epidemic routing [16] is the first flooding-based routing algorithm. Gossip [4], forwards with probability p. Opportunistic routings, such as [5], forward based on the delivery probability. Different delivery probability metrics are proposed including encounter frequency [5], time elapsed since last encounter [8, 18, 19, 23, 24], social similarity [6], location similarity [25], geometric distance [20], etc.

Traces data [22] available for the research community include UMassDieselNet trace, the NUS student contact trace, etc. In [26], 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 [5] and [8] in spirit.

6. CONCLUSION

In this paper we have presented our first research investigating a new cyclic delivery probability metric, expected minimum delay (EMD), and provided methods to achieve it in a cyclic MobiSpace. Our proposed probabilistic routing, routing in cyclic MobiSpace routing (RCM) is evaluated and compared with the enhanced versions of some existing DTN routing protocols using the NUS student trace, the UMassDieselNet trace, and our synthetic traces. The simulation results demonstrate that RCM outperforms the compared protocols in terms of delivery rate and delay.

Due to space limitation, we left some performance analysis of our protocol to the future work, which includes the impact of size of time slots on the performance of RCM. We will also create more synthetic traces with different maps of metro routes and generalize the conditions where RCM can provide most significant improvement.

7. REFERENCES

[1] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss. Delay-tolerant Network Architecture. In *Internet draft: draft-irrf-dtnrg-arch.txt, DTN Research Group*, 2006.

[2] S. Jain, K. Fall, and R. Patra. Routing in a Delay Tolerant Network. In *Proc. of ACM SIGCOMM*, 2004.

[3] S. Merugu, M. Ammar, and E. Zegura. Routing in Space and Time in Network with Predictable Mobility. In *Technical report: GIT-CC-04-07, College of Computing, Georgia Tech*, 2004.

[4] J. Haas, J. Y. Halpern, and L. Li. Gossip-Based Ad Hoc Routing. In *Proc. of IEEE INFOCOM*, 2002.

[5] A. Lindgren, A. Doria, and O. Schelen. Probabilistic Routing in Intermittently Connected Networks. *Lecture Notes in Computer Science*, 3126:239–254, August 2004.

[6] D. Elizabeth and H. Mads. Social Network Analysis for Routing in Disconnected Delay-Tolerant MANETs. In *Proc. of ACM MobiHoc*, 2007.

[7] C. Liu and J. Wu. Scalable Routing in Delay Tolerant Networks. In *Proc. of ACM MobiHoc*, 2007.

[8] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networking. In *Proc. of IEEE INFOCOM*, 2006.

[9] 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. of ACM MobiCom*, 2007.

[10] R. Bellman. *Dynamic Programming*. Princeton Univerity Press, 1957.

[11] R. A. Howard. *Dynamic Programming and Markov Processes*. The MIT Press, 1960.

[12] B. Bonet and H. Geffner. Faster Heuristic Search Algorithms for Planning with Uncertainty and Full Feedback. In *Proc. of IJCAI*, 2003.

[13] P. Dai and J. Goldsmith. Topological Value Iteration Algorithm for Markov Decision Processes. In *Proc. of IJCAI*, 2007.

[14] A. J. Briggs, C. Detweiler, D. Scharstein, and A. Vandenberg-Rodes. Expected Shortest Paths for Landmark-Based Robot Navigation. *International Journal of Robotics Research*, 23(7-8):717–728, July-August 2004.

[15] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Second Edition*. The MIT Press, 2001.

[16] A. Vahdate and D. Becker. Epidemic Routing for Partially-connected Ad Hoc Networks. In *Technical Report, Duke University*, 2002.

[17] T. Spyropoulos, K. Psounis, and C. Raghavendra. Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks. In *Proc. of ACM WDTN*, 2005.

[18] T. Spyropoulos, K. Psounis, and C. Raghavendra. Spray and Focus: Efficient Mobility-Assisted Routing for Heterogeneous and Correlated Mobility. In *Proc. of IEEE PerCom*, 2007.

[19] A. Balasubramanian, B. N. Levine, and A. Venkataramani. DTN Routing as a Resource Allocation Problem. In *Proc. ACM SIGCOMM*, 2007.

[20] U. Acer, S. Kalyanaraman, and A. Abouzeid. Weak State Routing for Large Scale Dynamic Networks. In *Proc. of ACM MobiCom*, 2007.

[21] V. Srinivasan, M. Motani, and W. T. Ooi. Analysis and Implications of Student Contact Patterns Derived from Campus Schedules. In *Proc. of ACM MobiCom*, 2006.

[22] CRAWDAD data set. Downloaded from http://crawdad.cs.dartmouth.edu/.

[23] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli. Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Encounter Ages. In *Proc. of ACM MobiHoc*, 2003.

[24] M. Grossglauser and M. Vetterli. Locating Nodes with Ease: Last Encounter Routing in Ad Hoc Networks through Mobility Diffusion. In *Proc. of IEEE INFOCOM*, 2003.

[25] J. Leguay, T. Friedman, and V. Conan. DTN Routing in a Mobility Pattern Space. In *Proc. of ACM WDTN*, 2005.

[26] L. Song and D. F. Kotz. Evaluating Opportunistic Routing Protocols with Large Realistic Contact Traces. In *Proc. of ACM CHANTS*, 2007.