Scalable Routing in Delay Tolerant Networks *

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

ABSTRACT

The non-existence of an end-to-end path poses a challenge in adapting the traditional routing algorithms to delay tolerant networks (DTNs). Previous works include centralized routing approaches based on deterministic mobility, ferry-based routing with deterministic or semi-deterministic mobility, flooding-based approaches for networks with general mobility, and probability-based routing for semi-deterministic mobility models. Unfortunately, none of these methods can guarantee both scalability and delivery. In this work,we investigate scalable deterministic routing in DTNs. Instead of routing with global contact knowledge, we propose a simplified DTN model and a routing algorithm which routes on contact information compressed by three combined methods. Analytical studies and simulation results show that the performance of our proposed routing algorithm, *DTN Hierarchical Routing* (DHR), approximates that of the optimal time-space Dijkstra's algorithm in terms of delay and hop-count. At the same time, the per node storage overhead is substantially reduced and becomes scalable. Although our work is based on a simplified DTN model, we believe this approach will lay a groundwork for the understanding of scalable routing in DTNs.

Categories and Subject Descriptors

C.2.2 [**Computer Systems Organization**]: COMPUTER-COMMUNICATION NETWORKS—*Network Protocols*

General Terms

Algorithms, Performance

Keywords

Contact, delay tolerant networks (DTNs), delivery, hierarchical routing, motion cycle, scalability, simulation

*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'07, September 9–14, 2007, Montréal, Québec, Canada.

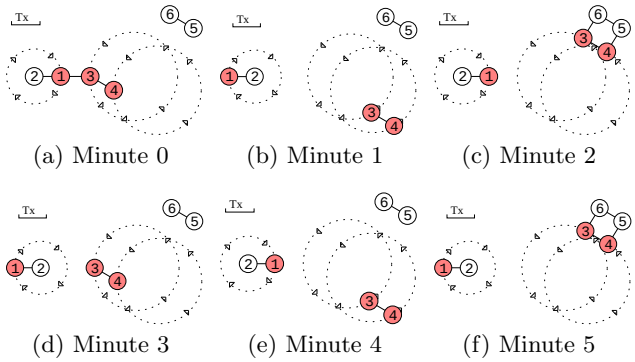


Figure 1: A DTN in its motion cycle of 6 minutes. (a)-(f) are the snapshots of the network at the minute mark.

1. INTRODUCTION

Delay tolerant networks (DTNs) are occasionally-connected networks that may suffer from frequent partitions. Representative DTNs include sensor-based networks using scheduled intermittent connectivity, terrestrial wireless networks that cannot ordinarily maintain end-to-end connectivity, satellite networks with moderate delays and periodic connectivity, and underwater acoustic networks with moderate delays and frequent interruptions due to environmental factors. Snapshots of a DTN over a period of 6 minutes are shown in Figure 1, where nodes 1, 3, and 4 have circular trajectories and nodes 2, 5, and 6 are static nodes. The motion cycle of node 1 is 2 minutes and that of nodes 3 and 4 is 3 minutes. The network has a motion cycle of 6 minutes since the same snapshot of the network reoccurs only every 6 minutes. For a message that is in node 1 and is heading for node 6, a routing scheme will send it to node 3 immediately. Then the routing scheme allows node 3 to store the message until minute 2 when node 3 gets the opportunity to forward the message to node 6.

The Delay Tolerant Network Research Group (DTNRG) has designed a complete architecture to support various protocols in DTNs [1]. A DTN can be described abstractly using a graph. Each edge in this graph contains a set of contacts. A *contact* is a period of time during which two neighboring nodes can communicate with each other. Several types of contacts are defined in [1]. Among them are the *persistent contacts* between the persistently connected nodes and the *predicted contacts* that are intermittently available.

Their availability is predicted based on the history of previously observed contacts or some other information.

Routing in DTNs with predicted contacts is an active research area. In Figure 1, the contact between nodes 1 and 2, the one between nodes 3 and 4, and the one between nodes 5 and 6 are persistent contacts. All other contacts in the same figure are predicted contacts which repeat with frequencies of at least once every 6 minutes. Figure 2 is the graph model of the physical network in Figure 1. In this figure, a thick line represents a persistent contact, and a thin line represents a predicted contact.

Routing in DTNs poses some unique challenges compared to a conventional data network due to the uncertainty and time-varying nature of network connectivity. The additional complexity of the time dimension significantly complicates the routing decision. Centralized routing approaches based on deterministic mobility include [8] and [11]. Ferry-based routing includes those with deterministic [15], [18] and semi-deterministic [17] mobility. Flooding-based approaches for networks with general mobility include [3], [7], and [16]. Probability-based routing without a message delivery guarantee for semi-deterministic mobility models include [6], [9], and [10]. Like many of the previous works, this paper will focus on deterministic mobility where the mobility of the nodes are predetermined and, therefore, the contacts in the network are predictable.

On the other hand, many hierarchical routing approaches [2], [5], [12], and [13] have been proposed in MANETs where the (virtual) hierarchical network constructed by multilevel clustering is used as a compressed topology abstraction for routing. In this paper, we will adapt the hierarchical network, which was previously only applicable in static and connected networks, to our DTN model. The challenge is to effectively represent the time-space information in the hierarchical structure. There is no previous work in DTNs that use the hierarchical network to compress the time-space information and use it as a time-space topology abstraction in routing. Moreover, there is no previous solution that guarantees both scalability and delivery.

Since it is difficult if not impossible for a scalable and delivery-guaranteed algorithm to obtain the necessary information in order to make a routing decision in a network of completely random movement, we propose a simplified DTN model in which each node is either static or has a strict repetitive motion. We name our proposed routing protocol in this model the *DTN Hierarchical Routing* (DHR). Though our model poses strong assumptions in some circumstances, it is adaptable in many other situations, and we believe that our work will lay a groundwork for the understanding of scalable routing in DTNs.

We achieve scalability in our model by using three combined methods to compress the time-space topology information. First, we construct a hierarchical network by performing multilevel clustering. We aggregate related contact information to the links in the hierarchical network. In order to route, each node only needs to have partial global topology information (consisting of the contact information stored in some of the links in the hierarchical network) whose size is $O(\log(N))$ in the network of size N.

Two other methods are used to compress the contact information in the links in the hierarchical network. The first one is the redundant contact information removal method, which removes the contact information in a link that does not contribute to the link's underlying time-variant shortest path. The second method is to stop aggregating contact information in the links above a certain level, i.e., to remove the time dimension of the time-space information in the upper levels of the hierarchical network. Analytical and experimental results show that both of these compression methods have limited effects on routing performance.

The contribution of our work is summarized as follows:

- We present a DTN model which is realistic for some practical DTNs and with which multilevel clustering can be performed to build a hierarchical network.
- Based on this model, we build the first hierarchical network in DTNs in which the time-varying nature of the physical topology is reflected by the time-variant information (the contact information) maintained by the links in the hierarchical network.
- We develop compression algorithms which reduce the amount of information in the hierarchical links without much loss or significant impact on the routing performance.
- We devise a hierarchical routing algorithm based on our hierarchical network. Analytical studies and simulation results support that, in our DTN Hierarchical Routing (DHR) algorithm, the size of the per node information is small and scalable, and the routing performance approximates that of the optimal time-space Dijkstra's algorithm.

The rest of the paper is organized as follows. A simplified DTN model is given in Section 2. Section 3 presents a framework of multilevel clustering and hierarchical routing. Section 4 proposes the hierarchy clustering and the routing algorithm in our DTN model as well as other routing information compression methods. Section 5 presents analysis on the routing performance and the scalability of our algorithm. Simulation and results are displayed in Section 6. Section 7 discusses related works. Finally, Section 8 concludes the paper.

2. A SIMPLIFIED DTN MODEL

We model the DTN as a graph where vertices are nodes and edges are sets of (representative) contacts. Our model allows only static nodes and nodes with strict repetitive motions. A strict repetitive motion means that the node moves on a predetermined trajectory repetitively and the position of the node is a function of time in each repetition. Although this assumption might be quite strong in some situations, it is not unrealistic in many other cases since the motion of most real objects (such as buses and airplanes) have repetitive patterns, and their positions at any particular time can be roughly estimated. Example networks to which such a model can be directly adopted are satellite networks where nodes have accurate periodic connectivity, and underwater acoustic networks where events that trigger the aberrancy of the nodes from their fixed trajectory are rare.

We define the *motion cycle* T_i of a mobile node n_i having a strict repetitive motion as a period of time such that if the node is at any point p at time t, then it is at p at time $t+k\times T_i$ for any integer k. The *relative motion cycle* $T_{S'}$ of a set of mobile nodes $S' = \{n_1, n_2, \ldots, n_k\}$ having strict repetitive

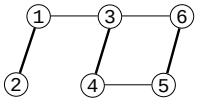


Figure 2: The simplified DTN model for the DTN in Figure 1.

Table 1: Contacts of the nodes in Figure 2.

Node	**Contacts** $(n_i, n_j, T_{ij}, t_s, t_d)$
1	(1, 2, -, -, -), (1, 3, 6, 0, 0.1)
2	(2, 1, -, -, -)
3	(3, 4, -, -, -), (3, 1, 6, 0, 0.1), (3, 6, 3, 2, 0.1)
4	(4, 3, -, -, -), (4, 5, 3, 2, 0.1)
5	(5, 6, -, -, -) ,(5, 4, 3, 2, 0.1)
6	(6, 5, -, -, -), (6, 3, 3, 2, 0.1)

motions is a period of time such that, if $n_1, n_2, \ldots, n_k$ are at points $p_1, p_2, \ldots, p_k$ respectively at any time t, then they are at $p_1, p_2, \ldots, p_k$ respectively at time $t + k \times T_{S'}$. The relative motion cycle T_S of a set S of nodes (including static nodes and nodes with strict repetitive motions) is equal to $T_{S'}$, if $S' \subseteq S$ ($S' \neq \emptyset$), and S' is the set of all mobile nodes in S.

For simplicity, T_S of $S = \{n_i, n_j\}$ is denoted by T_{ij}. It is obvious that if T_i and T_j are the motion cycles of n_i and n_j respectively, then T_{ij} equals the *least common multiple* (LCM) of T_i and T_j. A similar result holds when S has more than two elements. In Figure 1, for example, the motion cycles T_1 and T_3 of nodes 1 and 3 are 2 minutes and 3 minutes respectively, and thus the relative motion cycle T_{13} of nodes 1 and 3 is $LCM(2, 3) = 6$ minutes.

Once the relative motion cycle T_{ij} of nodes n_i and n_j is known, the set of contacts C occurring within any period of time equal to T_{ij} can be used to represent all the other contacts. In this paper, a predicted contact is represented by a tuple $(n_i, n_j, T_{ij}, t_{start}, t_{duration})$, and a persistent contact by $(n_i, n_j, -, -, -)$ where t_{start} and $t_{duration}$ are the starting time and the duration (which depends on speed and transmission range) of the contact. For simplicity, we construct a representative set C of contacts for each pair of nodes within the time period $(0, T_{ij})$. That is, $0 \leq t_{start} < T_{ij}$. The representative contacts for the nodes in Figure 1, whose model is shown in Figure 2, is given in Table 1. A uniform contact duration 0.1 is set to simplify the discussion.

With the representative contacts for a set V of nodes in the network, one can use the optimal time-space Dijkstra's algorithm, such as the ones in [8] and [11], to calculate the shortest path between a pair of nodes in V. Since our routing is unicast and there are usually multiple paths with the same minimal delay, we design an extended time-space Dijkstra's algorithm which first finds all paths with the minimal delay and then chooses one among those paths with the least hop-count randomly. This algorithm is used in DHR to route with local contact information in some levels of the hierarchical network. It is also used to compute the optimal routing solution on the global topology that is the basis upon which the routing performance of DHR is evaluated.

3. MULTILEVEL CLUSTERING AND HIERARCHICAL ROUTING

This section reviews a multilevel clustering and hierarchical routing algorithm (which is similar to [2], [5], [12], and [13]) in static networks on which our proposed algorithm in section 3.2 is based. Organizing a network into a hierarchical structure could make the management of routing tables scalable. Clustering offers such a structure, and it suits networks with relatively large numbers of nodes. Multilevel clustering, which is clustering applied recursively over clusterheads, is feasible and effective in large networks.

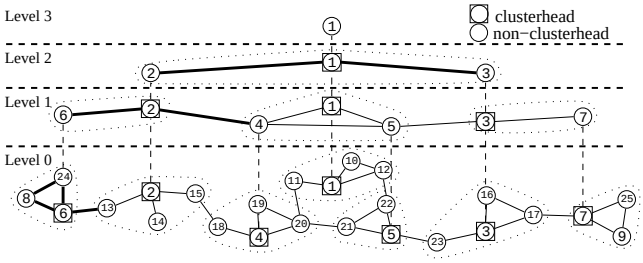


Figure 3: Lowest ID hierarchical clustering.

3.1 Multilevel clustering

Clustering is conducted first by electing clusterheads. Then, non-clusterheads choose clusters to join and become members. There are two kinds of clustering algorithms. One is the cluster algorithm [4]. In this algorithm, a node selects itself as a clusterhead if it has the highest priority among its unclustered neighbors. A non-clusterhead joins the cluster of a clusterhead that has the highest priority among the node's neighboring clusterheads. The other is the core algorithm [13] in which each node selects a node in its neighborhood including itself with the highest priority as a clusterhead and then joins that node's cluster. The main difference between these two are whether clusterheads could be neighbors (as in core) or not (as in cluster). Also, the cluster algorithm runs sequential rounds while the core algorithm needs one round to complete. In our work, we use the cluster algorithm. There are many ways to define node priorities. The lowest ID algorithm [14] is widely used. In the lowest ID cluster algorithm, the lower a node's ID, the higher is its priority. Figure 3 shows the hierarchical network as the result of the multilevel clustering using the lowest ID algorithm on the physical network shown in level 0 of the hierarchy.

The hierarchical network is a logical tree of nodes in a homogeneous network, where nodes and links above level 0 in the hierarchy are conceptual (in contrast to the physical nodes and links in level 0). A node in level $k+1$ corresponds to a cluster (or a clusterhead) in level k, and the ID of the level $k+1$ node is the same as that of the clusterhead in the level k cluster. We use subscripts to denote the level to which a node belongs. In Figure 3, we use 6_1 to denote that the node with label 6 is in level 1, which represents the cluster in level 0 consisting of 6_0, 8_0 and 24_0. A *hierarchy address* of a node is a sequence of IDs of the clusters of the node and its clusters in all levels. For instance, the hierarchy address of the physical node 8 is $(1_3, 2_2, 6_1, 8_0)$, where 6_1 is the cluster of 8_0 in level 1, 2_2 is the cluster of 6_1 (and thus of 8_0) in level 2, and 1_3 the cluster in level 3.

Each clusterhead needs to know all of its members and its adjacent clusterheads. If there is a link between any two nodes of two clusters on level k, then there is a link between the nodes representing these clusters in level $k+1$. For

instance, in Figure 3, there is a link between 6_1 and 2_1 since there is a link between 6_0 and 13_0. We call 6_0 a *gateway* and 13_0 a *remote gateway* from cluster 6_1 to cluster 2_1. Each level $k+1$ link is associated with a delay which is equal to the delay of the shortest path between the corresponding level k clusterheads. For instance, if the delay of the link between 6_0 and 13_0 is 3 and that of the link between 13_0 and 2_0 is 2, then the delay of the link between 6_1 and 2_1 is 5, since 5 is the delay of the shortest path (the only path in this example) between 6_0 and 2_0

3.2 Hierarchical routing

Hierarchical routing facilitates the hierarchical network as a topology abstraction and may not generate a shortest path. The advantage of hierarchical routing is that it is scalable (logarithmic) for localized traffic patterns, and it does not need location information.

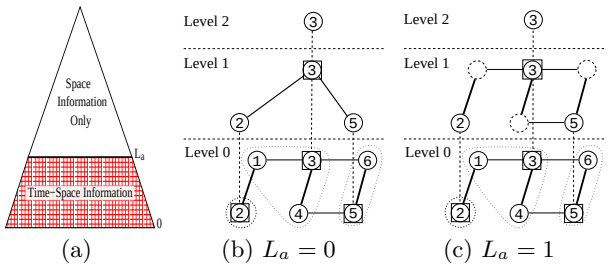
Hierarchical routing requires that the source has the hierarchy address of the destination. If only the ID of the destination is available, the source can resort to using location service [2]. Hierarchical routing is a hop-by-hop routing rather than a source routing. Before any routing, each node in the network needs to obtain the topology information of its clusters in all levels. For instance, in Figure 3, node 8 has the topology information consisting of all the thick links. Note that it includes the gateway links, such as links $(6_0, 13_0)$ and $(2_1, 4_1)$. The availability of this routing algorithm releases the clusterheads' burden of relaying every message forwarded to other clusters.

Each node makes its forwarding decision via the following steps: (1) find the lowest level k where the source s and the destination d have a common cluster, (2) define the intermediate source s' and the intermediate destination d', which are the level k clusters of s and d respectively, (3) use the optimal time-space Dijkstra's algorithm to find the next hop n' on the shortest path from s' to d' based on the level k topology information of s. (4) if $k = 0$, n' is the forwarding decision of s, otherwise go back to step 3 with a new $k = k - 1$, a new d' being the remote gateway from s' to n', and a new s' being the the node on level k (new k) which is either s or a cluster of s.

As an example, we show the process of node 20 making its forwarding decision for the destination, node 9, in Figure 3. The lowest level where nodes 20 and 9 have a common cluster head is $k = 2$, and thus $s' = 1_2$, $d' = 3_2$, and the resulting $n' = 3_2$. Since $k = 2$, we continue with $k = 1$, where $s' = 4_1$, $d' = 3_1$, and the resulting $n' = 5_1$. Again, we continue with $k = 0$, where $s' = 20_0$, $d' = 21_0$ and the resulting $n' = 21_0$. Thus, node 20's decision is to forward the message to node 21.

4. MULTILEVEL CLUSTERING AND HIERARCHICAL ROUTING IN DTNS

As mentioned in Section 3, the information in a level $k+1$ link in a static hierarchical network contains a time-invariant delay which is simply the sum of the delays of the level k links on a shortest path between the level k clusterheads. The challenge in multilevel clustering in our DTN model is that the delay information in the links is time-variant. A method needs to be proposed to aggregate the time-varying information to the links in the hierarchical network from the level that is immediately below them.



(a) (b) $L_a = 0$ (c) $L_a = 1$

Figure 4: A hierarchical network with aggregated contact information given an aggregation level.

4.1 Multilevel clustering in DTNs

Each node on level 0 of the hierarchical network corresponds to a physical node, and there is a link between two level 0 nodes if there are contacts between their physical nodes. To allow the links in each level in the hierarchical network to have time-variant delay information, we store in each link a set of contacts. Contacts are stored in the 5-element tuple format presented in Section 2. A level 0 link stores all contacts between the two end nodes. A level $k+1$ link aggregates all contacts in the level k links that form paths between the corresponding level k clusterheads, which includes those between clusterheads and gateways and those between gateways.

For example, in Figure 4(c), nodes 3_0 and 5_0 are clusterheads, and the level 1 link between 3_1 and 5_1 contains the contact information contained in all of the level 0 links between 3_0 and 4_0, between 5_0 and 6_0, between 3_0 and 6_0, and between 4_0 and 5_0.

The size of the aggregated contact information in the links increases exponentially as their level increases. To ensure scalability, the aggregation should stop at a certain level which we call the *aggregation level* L_a. The links above level L_a maintain time-invariant delays just as in the static hierarchical network, which is illustrated in Figure 4(a).

Figure 4(b) and Figure 4(c) show the results of two multilevel clusterings on the same network with different L_a (0 and 1). One difference is shown in the link between nodes 3_1 and 5_1. In Figure 4(b), we use a thick line to represent a time-invariant delay in the link since $L_a < 1$. In Figure 4(c), we move the four links from level 0 to level 1 to illustrate that all related contact information is aggregated to the link from the level immediately below.

The delay on the links above level L_a is calculated in the same way as in the static hierarchial network. To calculate the time-invariant delay on level $L_a + 1$, each link on level L_a also needs to have a time-invariant delay. We use the *weighted average delay* which is the expectation (in statistics) of the time-variant delays of the shortest paths between these two end nodes of the level L_a link. In our DTN model, the weighted average delay can be calculated with the set of shortest paths within a period of time equal to the LCM of the cycles of the contacts stored in the link.

We will now present two examples to show the calculation of the weighted average delay. First, we calculate $D(1_0, 3_0)$ in Figure 4(c), which is a hierarchical network of the DTN in Figure 1. The link between 1_0 and 3_0 contains only one contact $(1, 3, 6, 0, 0.1)$ as given in Table 1. Here we assume the transmission delay (including queuing delay and radio

transmission duration) is 0.1. During the first 0.01 minute when the contact is active, the delay is 0.01 (the transmission delay). In the remaining 6 minutes, the delay is $(6-t+0.01)$ where $(6-t)$ is the waiting time for the next contact. Thus,

$$D(1_0, 3_0) = \frac{1}{6}[\int_0^{0.01} 0.1dt + \int_{0.1}^{6} (6 - t + 0.01)dt] \approx 3.$$

Second, we calculate $D(3_1, 5_1)$ in Figure 4(c). The link between 3_1 and 5_1 contains four contacts, and the LCM of their cycles is 3 minutes. We can observe in Figure 1 that, in the 2.9 minutes, before the occurrence of contact between nodes 3 and 6, path $(3, 4, 5)$ has the minimal delay $\max(3 - t + 0.01, 0.02)$; during the contact (0.1 minute, especially the last 0.01 minute), path $(3, 6, 5)$ has the minimal delay of 0.02. Therefore,

$$D(3_1, 5_1) = \frac{1}{3}[\int_0^{0.1} 0.02dt + \int_{0.1}^{3.0} \max(3-t+0.01, 0.02)dt] \approx 1.5.$$

We show the priorities and the weighed average delays used in the multilevel clustering in Figure 4(c) in Table 2.

The weighted average delay is also used in the calculation of the node priorities in the multilevel clustering as shown below.

The multilevel clustering algorithm we use in our DTN model uses two priorities for each node: the *absolute priority* $Pr(n)$ and the *relative priority* $Pr_i(n)$.

$$Pr(n) = \sum_{i,j \in N(n), i \neq j \neq n} \frac{\frac{1}{D(i,n)+D(n,j)}}{\frac{1}{D(i,j)} + \frac{1}{D(i,n)+D(n,j)}}.$$

Here $N(n)$ is the set of the neighbors of node n and $D(i, j)$ is the weighted average delay between nodes i and j. Note that each term inside the summation has a value in $(0, \frac{1}{2}]$ (since $D(i, j) \leq D(i, n) + D(n, j)$). The criteria for clusterhead selection reflected by the absolute priority are (1) a clusterhead is on or close to the shortest path between any two neighbors (which is reflected by the value of the term inside the summation), and (2) a clusterhead has a large node degree (which is reflected by the summation).

$$Pr_i(n) = \frac{Pr(n)}{D(i, n)}.$$

The criterion for clusterhead selection shown in the relative priority is: a cluster member prefers a nearby (in terms of delay) clusterhead.

The clustering algorithm is simply presented as the following: (1) each node calculates its absolute priority, (2) a node without a neighboring clusterhead declares itself a clusterhead if it has the largest absolute priority among its neighbors that haven't joined any cluster, and (3) a node having neighboring clusterheads chooses one with the largest relative priority and joins that cluster.

4.2 Contact information compression

We use two methods to compress the aggregated contact information. The first method, which has already been presented, compresses the time-space information by removing the time-dimension information in the links above level L_a. Analytical study and simulation in later sections show that its impact on the routing performance is slight when L_a is not very small. An intelligent improvement is to decide whether to compress according to the volume of the contact information and the variance (in statistic) of the time-variant delays of the link.

(a) Level 0 average delay

$D(i,j)$	$i = 1_0$	$i = 2_0$	$i = 3_0$	$i = 4_0$	$i = 5_0$	$i = 6_0$
$j = 1_0$	-	0.1	3	-	-	-
$j = 2_0$	0.1	-	-	-	-	-
$j = 3_0$	3	-	-	0.1	-	1.5
$j = 4_0$	-	-	0.1	-	1.5	-
$j = 5_0$	-	-	-	1.5	-	0.1
$j = 6_0$	-	-	1.5	-	0.1	-

(b) Level 0 node priority

Level 0 Priority	Absolute $Pr(n)$	Relative $Pr_j(n)$					
		$j = 1_0$	2_0	3_0	4_0	5_0	6_0
$n = 1_0$	1	-	10	3.33	-	-	-
$n = 2_0$	0	0	-	-	-	-	-
$n = 3_0$	3	1	-	-	30	-	2
$n = 4_0$	1	-	-	10	-	0.67	-
$n = 5_0$	1	-	-	-	0.67	-	10
$n = 6_0$	1	-	-	0.67	-	10	-

(c) Level 1 average delay

$D(i,j)$	$i = 2_1$	$i = 3_1$	$i = 5_1$
$j = 2_1$	-	3.1	-
$j = 3_1$	3	-	1.5
$j = 5_1$	-	1.5	-

(d) Level 1 node priority

Level 1 Priority	Absolute $Pr(n)$	Relative $Pr_j(n)$		
		$j = 2_1$	$j = 3_1$	$j = 5_1$
$n = 2_1$	0	-	0	-
$n = 3_1$	1	0.33	-	0.67
$n = 5_1$	0	-	0	-

Table 2: Data in the example hierarchical clustering of Figure 4(c).

The second compression method is to remove part of the contact information from the links. For a hierarchy link on level L $(0 \leq L \leq L_a)$, a contact is meaningful to calculate the underlying shortest path of the link only if it is on a shortest path at a particular time. This method is illustrated by the following example. The physical network in this example is shown in Figure 5(a), its model is shown in Figure 5(b), and the contacts in the network model are shown in Table 3. Node 4 has a motion cycle of 2 minutes, and the motion cycles of nodes 1 and 2 are both 6 minutes.

The hierarchical clustering process runs on the network and the result of the level 0 cluster is shown in Figure 5(b). As shown in Figure 5(c), the level 0 clusterheads are 6_0 and 7_0. 4_0 and 5_0 are the gateways from 6_1 to 7_1, and 1_0 and 2_0 are the gateways from 7_1 to 6_1. Contacts on all of the possible paths between 6_0 and 7_0 through the gateways are aggregated to the level 1 link between 6_1 and 7_1.

Since all the predicted contacts shown in Table 3 have a common period of 6 minutes, we will examine the paths from 6 to 7 with minimal delay in the first 6 minutes. It can be observed from Figure 5(a) and Table 3 that the path with the minimal delay during minute 3 and minute 5 is $(6, 5, 2, 7)$ as shown in Figure 5(d), and the path with the minimal delay in the rest of the time is $(6, 5, 1, 7)$ as shown in Figure 5(e). After removing all the contacts that are not on these two paths, the result of the compression is shown in Figure 5(f).

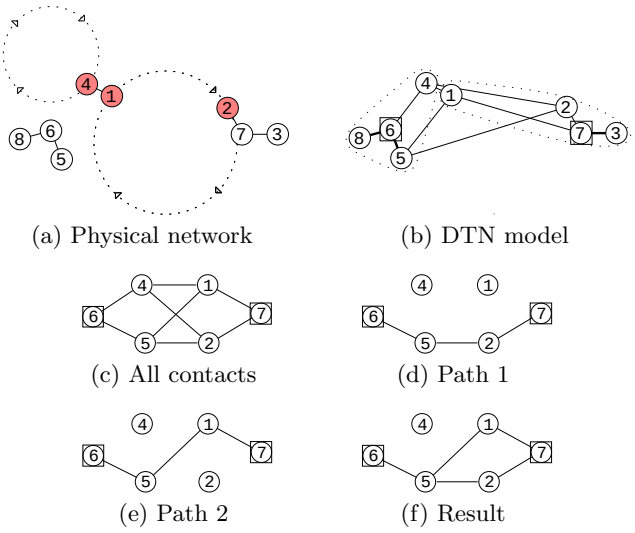


Figure 5: Another physical network at time 0 (a) and its model (b). The aggregated contact information compression process (c)-(f) on the link between 6_1 and 7_1.

Node	Contacts $(n_i, n_j, T_{ij}, t_s, t_d)$
1	(1, 4, 6, 0, 0.1), (1, 5, 6, 5, 0.1), (1, 7, 6, 2, 0.9)
2	(2, 4, 6, 4, 0.1), (2, 5, 6, 3, 0.1), (2, 7, 6, 0, 0.9)
3	(3, 7, -, -, -)
4	(4, 6, 2, 0.3, 0.1), (4, 1, 6, 0, 0.1), (4, 2, 6, 4, 0.1)
5	(5, 6, -, -, -), (5, 1, 6, 5, 0.1), (5, 2, 6, 3, 0.1)
6	(6, 4, 2, 0.3, 0.1), (6, 5, -, -, -), (6, 8, -, -, -)
7	(7, 3, -, -, -), (7, 1, 6, 2, 0.9), (7, 2, 6, 0, 0.9)
8	(8, 6, -, -, -)

Table 3: Contacts in the DTN model of Fig. 5(b).

The second compression method, which removes contacts from the links, is implemented as follows. Suppose (u, v) is a level k link, T is the LCM of the motion cycles of the contacts stored by (u, v), $\{t_i\}$ is the set of time slots which is a partition of T such that within each t_i the set of shortest paths between u and v does not change, and $\{S_{ij}\}$ is the set of contacts on the j^{th} shortest path during t_i. A boolean function E is constructed in the following steps. First, construct terms E_{ij} with the names of all contacts in $\{S_{ij}\}$ as literals. Second, construct disjunctive normal forms (a disjunction of terms) E_i with all E_{ij}s. Finally, construct E which is a conjunction of all E_is. Here, a set of contacts satisfies E_i if they form a shortest path during t_i, and a set R of contacts satisfying E can be used to calculate the delay of the shortest paths between u and v at any time. This compression algorithm reduces the size of R to reduce the storage and communication overhead while retains most of the important contacts for the links.

Again, an intelligent improvement is to choose the smallest set of contacts whose shortest paths' variance from the actual shortest paths is below a particular threshold. It should be able to further reduce the size of contact information when, for instance, 20% of the contacts form 80% of the shortest paths.

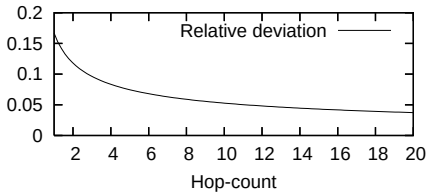


Figure 6: The inaccuracy of the time-invariant delay represented by the relative deviation $\frac{\sigma_n}{\mu_n}$.

4.3 DTN Hierarchical Routing (DHR)

Our proposed DTN Hierarchical Routing (DHR) is quite straightforward after the hierarchical network has been built. DHR is also a hop-by-hop routing. Each node makes its forwarding decision in two phases. The first phase runs only when the highest level L_c, on which the cluster of the current node and that of the destination are different, is greater than L_a. In this phase, the static hierarchical routing, as presented in Section 3.2, runs on the levels (from L_c to $L_a + 1$) where delay on the links are time-invariant. The result of this phase is an intermediate destination d' (see Section 3.2) of level L_a.

When $L_c \leq L_a$, the first phase does not run, and d' is set to the clusterhead of the destination in level L_c. The second phase, which we called *aggregation routing*, has the following steps: (1) all contact information in the links from level 0 to level L_a that are stored by the current node are aggregated to form a graph G. That is, G includes all contacts in the links in the clusters of the current node from level 0 to level L_a in the hierarchical network. (2) the optimal time-space Dijkstra's algorithm is performed on G to find a shortest path p from the current node to d'. The first hop on p is the current node's forwarding decision.

5. ANALYSIS

5.1 Routing performance

In this subsection, we will analyze the possible impact of removing the time dimension at higher levels (above L_a). We estimate the inaccuracy of using the weighted average delay to represent the delay of a time variant link with the *relative deviation* $\frac{\sigma_n}{\mu_n}$, where μ_n is the expectation of the delay of a n-hop link (i.e., the weighted average delay) and σ_n is the standard deviation of the delays of the shortest paths.

For simplicity, we only calculate $\frac{\sigma_n}{\mu_n}$ in a simplified setting where the waiting time for a contact in each hop is in a independent identical uniform distribution. Thus, $\mu_n = n \times \mu_1$ and $\sigma_n^2 = n \times \sigma_1^2$, where μ_1 and σ_1 is the expectation and the variance of the one hop delay. We have $\frac{\sigma_n}{\mu_n} = \frac{\sigma_1}{\mu_1}/\sqrt{n}$, which it plotted in Figure 6.

Since the hierarchial network provides partial time-variant shortest path information when $L_a > 0$, its performance should be bound by the results in Figure 6.

5.2 Overhead

This subsection briefly analyzes the overhead in DHR, i.e., the overhead in maintaining the hierarchy and the overhead in the routing process.

The contact information of each node is sent to its clus-

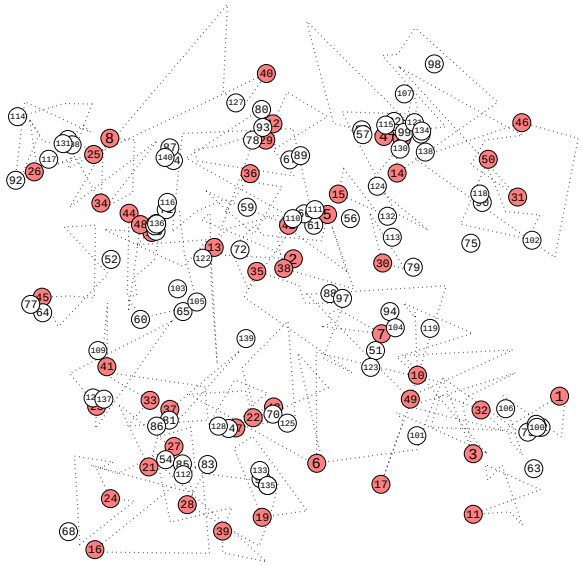


Figure 7: An example of the simulated network.

Parameter	Value
Field size	$5,000 \times 5,000(m^2)$
Transmission range	150(m)
Number of static nodes	30/60/90
Number of mobile nodes	70/60/50
Total number of nodes	100/120/140
Percentage of mobile nodes	70/50/35
Region size	600/800/1200(m)
Motion cycle	675/900/1350(s)
Aggregation level (L_a)	0-2

Table 4: Simulation parameters.

terhead which aggregates it to form a cluster topology and broadcasts the topology to the nodes in the cluster. In any broadcast algorithm, each node sends the message at most once, so the communication overhead will not exceed that of the storage overhead. Under level L_a (a constant) where the contact information is aggregated, the worst case contact information is $O(C^{L_a})$, where C is the minimal cluster size. For each cluster above L_a where links are time-invariant, cluster topology information size is $O(C)$. The number of levels in a hierarchical network is $O(\log(N))$, where N is the network size. Thus, the average communication and storage overhead for maintaining the hierarchical network is $O(\log(N) \cdot C^{L_a})$. The diameter of the network is $O(\sqrt{N})$. The routing overhead is on average $O(\sqrt{N})$.

6. SIMULATION & RESULTS

We develop a stand-alone, discrete event simulator to evaluate our protocol. This simulator only implements the network layers and it makes simple assumptions regarding lower layers. For instance, it assumes infinite bandwidth and nodes having infinite buffers. We compare the performance of DHR against the optimal time-space Dijkstra's algorithm in terms of delay and hop-count of the resulting paths.

6.1 Simulation settings

In our simulation, we generate connected (in the DTN sense) networks containing both mobile nodes and static nodes in a $5,000 \times 5,000m^2$ field. We generate three groups of 30 networks with different ratios of mobile nodes and static nodes as shown in Table 4. For instance, the first group contains 70 mobile nodes and 30 static nodes (mobile nodes account for 70% of the total nodes). All nodes have a uniform transmission range of 150m. An example of the simulation network is shown in Figure 7, where the trajectories of the mobile nodes are shown by dotted lines.

The trajectory of the a mobile node is generated as follows; (1) a square region of a given size is placed at a random position in the network, (2) 3 to 7 points are sprayed at random positions inside the region, (3) a trajectory is formed by starting form the first point, traveling each points once and finally coming back to the first point. Here we use the nearest neighbor algorithm in the traveling salesman problem to generate a short trajectory. The length of the sides of a square region is 600m, 800m, or 1200m. The nodes' motion cycles corresponding to these square regions are 675s, 900s or 1350s. Thus, the relative motion cycle of the network is 2,700s. The placement of the static nodes in the networks are random. Table 4 shows the critical simulation settings. We only use time-space connected networks in our simulations.

For a source-destination pair, we refer to the routes resulting from the optimal time-space Dijkstra's algorithm as the *Dijkstra routes*, and that of the DHR as *DHR routes*. In different experiments, we let L_a range from 0 to 2. After a hierarchical network is built, we route messages from every node in the network to 30 other randomly selected nodes. The delay and hop-count for all the source-destination pairs are then averaged and grouped by the hop-count of their Dijkstra route. The results are also averaged over the 30 networks for each group with the same mobile/static nodes ratio.

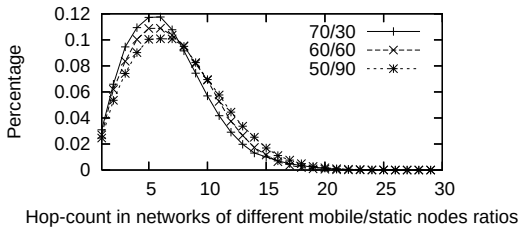
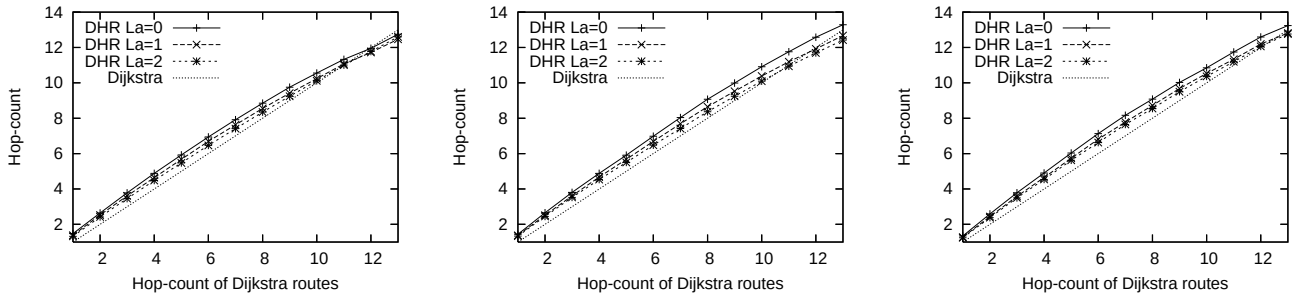


Figure 8: Percentage of routes of different lengths.

6.2 Simulation results and discussions

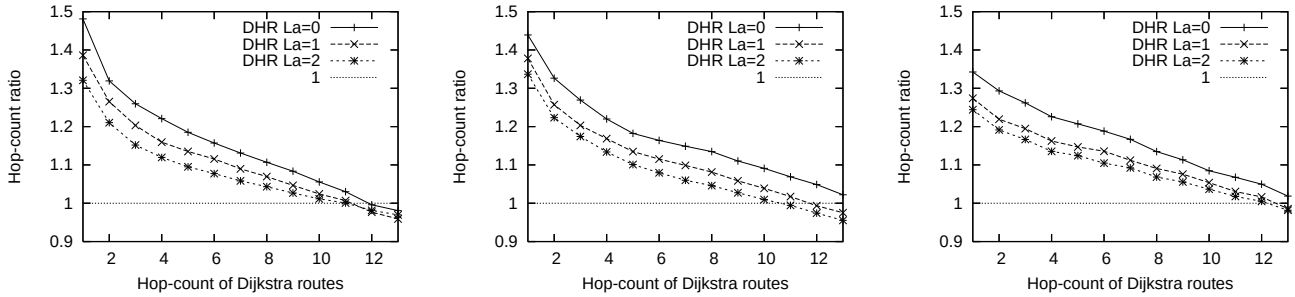
First, we show the results of DHR routing using only the compression which removes contacts above level L_a. Figure 8 shows the distribution of the hop-count of the Dijkstra routes in the three network groups. Thus we only show the simulation results where the hop-count is between 1 hop and 13 hops because only the routing performance within this range is accurate (with large volume of data) and representative (with more than 90% of the total paths).

Figure 9 shows the lengths of DHR routes in terms of hop-count. The x-axis is the length of the Dijkstra routes of the same source-destination pairs. DHR routes with different L_as are all very close to the Dijkstra routes in hop-count and a larger L_a gives a smaller hop-count. Note that Dijkstra routes are not always shorter than DHR routes, since they are first optimized in delay and then in hop-count.



(a) 70 mobile & 30 static nodes (b) 60 mobile & 60 static nodes (c) 50 mobile & 90 static nodes

Figure 9: Hop-count of DHR (with different L_as) routes compared with that of the Dijkstra routes in networks of different numbers of mobile nodes and static nodes.



(a) 70 mobile & 30 static nodes (b) 60 mobile & 60 static nodes (c) 50 mobile & 90 static nodes

Figure 10: The ratio of the hop-count of the DHR (with different L_as) routes to that of the Dijkstra routes in networks of different numbers of mobile nodes and static nodes.

Figure 10 presents the same data as Figure 9 from a different angle where the hop-counts of the DHR routes are divided by that of the Dijkstra routes. The hop-count of the DHR routes becomes smaller than in the Dijkstra routes when the hop-count of the Dijkstra routes is 13.

Figure 11 compares the DHR routes with the Dijkstra routes in terms of delay. DHR always perform worse than the optimal time-space Dijkstra's algorithm, and DHR with a larger L_a performs better. The figure shows that the DHR routes of the three groups of networks (different in static/mobile nodes ratio) have similar average delays.

Figure 12 shows the delay ratio of the DHR routes to the Dijkstra routes. The ratio is greater than and closer to 1.0 as hop-count increases. DHR paths have less than 20% additional delay than Dijkstra paths in most cases when $L_a = 1$ or $L_a = 2$, while they have more than 40% additional delay in all cases when $L_a = 0$. The performance of DHR becomes better when the source-destination distance increases, which is consistent with our previous analytical results shown in Figure 6.

We also conduct simulation applying the contact information removal method. Due to space limitation, only the delay ratios are shown. Figure 13 shows the delay ratio of DHR with and without contact information removal. Figure 14 shows the per node storage overhead in the hierarchical network. The overhead is calculated in terms of the ratio of the number of contacts stored in each node to the total number of contacts in the network. In Section 5, we have discussed that the communication overhead is proportional to the storage overhead.

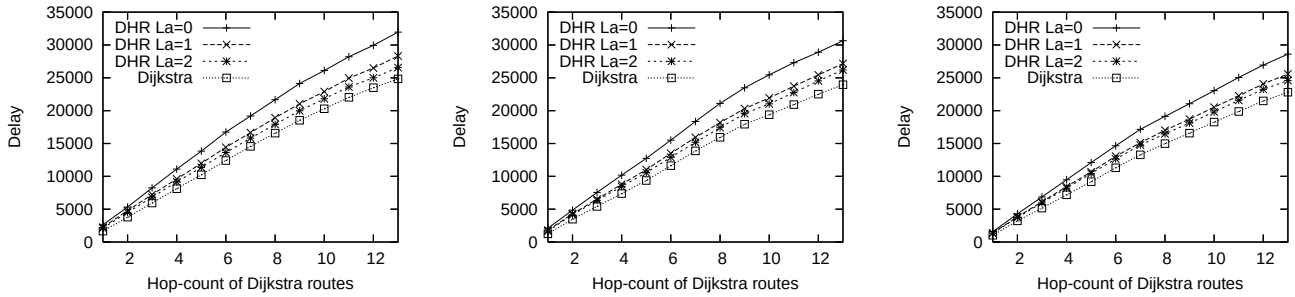
It is shown in Figure 14 that the per node contact information is only 1-3% of the total information size in the network when using contact information removal, which is a substantial saving when compared to the 10% when not using it. This provides a trade-off between scalability and routing performance.

6.3 Summary of simulation

To sum up, the simulation results show that our scalable algorithm, DHR, has a satisfactory performance that is comparable to the optimal result from the optimal time-space Dijkstra's algorithm for networks combined with mobile nodes and static nodes in different ratios. DHR with a larger aggregation level and less contact information compressed performs better than with a smaller aggregation level in terms of both hop-count and delay. Routing performance improves as the source and destination distance in terms of hop-count increases, which is consistent with our theoretical analysis as shown in Figure 6. While having good routing performance, the storage and communication overhead of running a DHR are far smaller than the overhead of running the optimal time-space Dijkstra's algorithm.

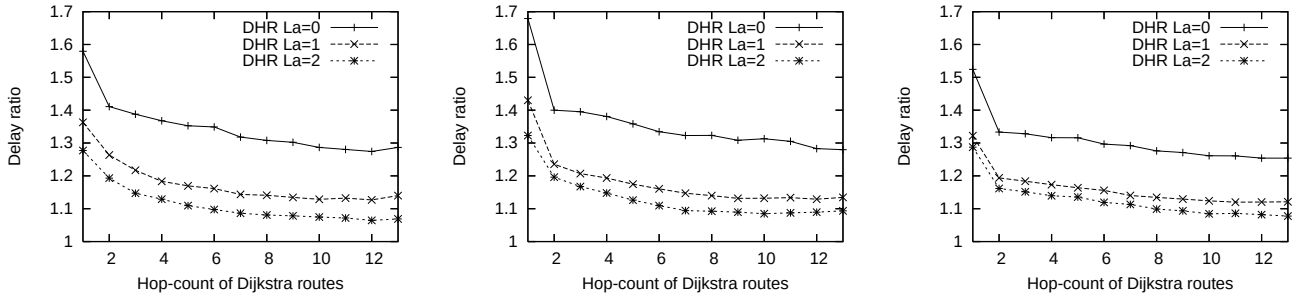
7. RELATED WORKS

In [8], Jain, Fall, and Patra exhaustively formulated the DTN routing problem under different degrees of knowledge about the network. Specifically, the Dijkstra's algorithm is used to calculate the shortest path with contacts being predictable, and also a linear programming approach which calculates the optimal routes across the network with ad-



(a) 70 mobile & 30 static nodes (b) 60 mobile & 60 static nodes (c) 50 mobile & 90 static nodes

Figure 11: The delay of the DHR (with different L_as) routes compared with that of the Dijkstra route in networks of different numbers of mobile nodes and static nodes.



(a) 70 mobile & 30 static nodes (b) 60 mobile & 60 static nodes (c) 50 mobile & 90 static nodes

Figure 12: The ratio of the delay of the DHR (with different L_as) routes to that of the Dijkstra routes in networks of different numbers of mobile nodes and static nodes.

ditional knowledge of the global traffic demand. In [11], Merugu, Ammar, and Zegura proposed a time-space routing algorithm that is similar in spirit to the Dijkstra's algorithm in [8].

Among the approaches in deterministic routing, [15] and [18] exploited deterministic trajectory of mobile nodes to help deliver data, improve data delivery performance, and reduce energy consumption in nodes. In [17], Wu, Yang, and Dai used semi-deterministic trajectory of mobile node to achieve deterministic results of several routing schemes. However, such a trajectory is selected from a set of predefined hierarchical structured routes.

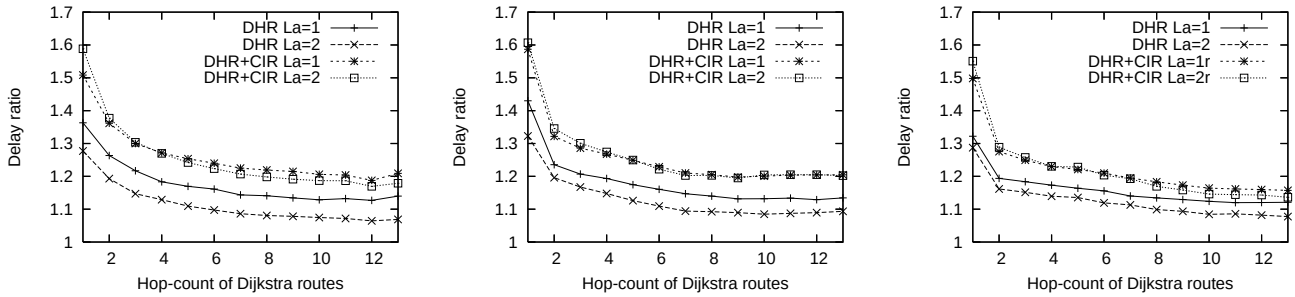
Epidemic routing [16] is a random movement and flooding-based algorithm, where all nodes are mobile and have infinite buffers. When a node has a message to send, it propagates the message to all nodes through contacts. Epidemic routing is extended in Gossip [7] where each message is flooded to the neighbor nodes with probability p. In *age matters* [3], Dubois-Ferriere, Grossglauser and Vetterli required that nodes keep a record of their most recent encounter times with all other nodes. Instead of searching for the destination, the source node searches for any intermediate node that encountered the destination more recently than did the source node itself.

In probability-based routing [10], each node maintains the delivery predictability (based on historical contacts) to all known destinations and uses them to make routing decisions. In [9], Leguay, Friedman, and Conan presented a routing scheme for DTNs which uses a high-dimensional Euclidean space constructed upon nodes' motion patterns. The axis in the Euclidean space could be contacts or particular locations in the network. This algorithm might not deliver messages in some situations, and requires global knowledge, such as the locations of the access points. Similarly, Solar [6] used a set of hubs for probability-based routing in semi-deterministic mobility modeling of DTNs.

Much work has been done on multilevel clustering and hierarchical routing in MANETs. In the multilevel clustering approaches such as DART [5], L+ [2], MMWN [13], and WHIRL [12], certain nodes are elected as clusterheads. These clusterheads in turn select higher level clusterheads, up to some desired level. A node's address is defined as a sequence of clusterhead identifiers, one per level, allowing the size of routing tables to be logarithmic in the size of the network. One problem with explicit clusterheads is that routing through clusterheads creates traffic bottlenecks. In L+, this issue is partially solved by allowing nearby nodes to route packets instead of the clusterhead.

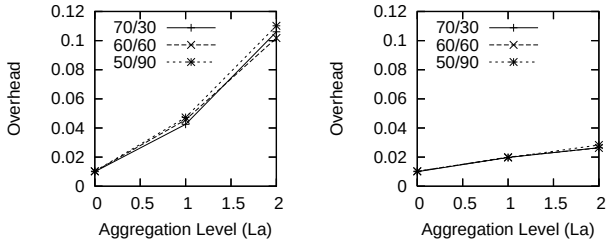
8. CONCLUSION

In this paper, we have proposed DTN Hierarchical Routing (DHR) in a simplified DTN model where nodes have strict repetitive motions. We constructed a hierarchical network which provides a compressed time-space topology abstraction of the network in our DTN model. Two aggregated contact information compression algorithms are presented for better scalability. Simulation results showed that the performance of DHR approximates that of the optimal time-space Dijkstra's algorithm in terms of delay and hop-count in networks of different ratio of mobile and static nodes.



(a) 70 mobile & 30 static nodes (b) 60 mobile & 60 static nodes (c) 50 mobile & 90 static nodes

Figure 13: The ratio of the delay of the DHR (with and without contact information removal) routes to that of the Dijkstra routes in networks of different number of mobile nodes and static nodes.



(a) Without redundant contact info removal. (b) With redundant contact info removal.

Figure 14: Storage/communication overhead.

Simulation results are consistent with theoretical analysis in that the performance of DHR increases as the source-destination distance increases, and the routing performance degrades slightly as more contact information in the hierarchical network is compressed.

Our future work will focus on improving the contact information compression algorithms. Two of such improvements have been suggested in Section 4.2. Relaxing the constraints in our DTN model and making necessary changes to the hierarchical clustering and routing algorithms are also very important future work to increase the applicability of DHR to more general DTN models.

9. 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] B. Chen and R. Morris. L+: Scalable landmark routing and address lookup fo multi-hop wireless network. In *MIT LCS Technical Report 837*, March 2002.

[3] 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.

[4] A. Ephremides, J. E. Wieselthier, and D. J. Baker. A design concept for reliable mobile radio networks with frequency hoping signaling. *Proc. of IEEE*, 75(1):56–73, January 1987.

[5] J. Eriksson, M. Faloutsos, and S. Krishnamurthy. Scalable ad hoc routing: The case for dynamic addressing. In *Proc. of IEEE INFOCOM*, 2004.

[6] J. Ghosh, S. J. Philip, and C. Qiao. Sociological orbit aware location approximation and routing (SOLAR) in MANET. In *Proc. of ACM MobiHoc*, 2005.

[7] J. Haas, J. Y. Halpern, and L. Li. Gossip-based ad hoc routing. In *Proc. of IEEE INFOCOM*, 2002.

[8] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. In *Proc. of ACM SIGCOMM*, 2004.

[9] J. Leguay, T. Friedman, and V. Conan. DTN routing in a mobility pattern space. In *Proc. of ACM SIGCOMM Workshop on Delay-Tolerant Networking*, 2005.

[10] A. Lindgren, A. Doria, and O. Schelen. Probabilistic routing in intermittently connected networks. *Lecture Notes in Computer Science*, 3126:239–254, Aug 2004.

[11] 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.

[12] G. Pei, M. Gerla, X. Hong, and C. Chiang. A wireless hierarchical routing protocol with group mobility. In *Proc. of WCNC*, 1999.

[13] R. Ramanathan and M. Steenstrup. Hierarchically organized, multihop mobile wireless networks for quality of service support. *Mobile Networks and Applications*, 3(1):101–119, 1998.

[14] P. Sinha, R. Sivakumar, and V. Bharghavan. Enhancing ad hoc routing with dynamic virtual infrastructures. In *Proc. of IEEE INFOCOM*, 2001.

[15] M. M. B. Tariq, M. Ammar, and E. Zegura. Message ferry route design for sparse ad hoc networks with mobile nodes. In *Proc. of ACM MobiHoc*, 2005.

[16] A. Vahdate and D. Becker. Epidemic routing for partially-connected ad hoc networks. In *Technical Report, Duke University*, 2002.

[17] J. Wu, S. Yang, and F. Dai. Logarithmic store-carry-forward routing in mobile ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 18(6), June 2007.

[18] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proc. of ACM MobiHoc*, 2004.