An Efficient Prediction-Based Routing in Disruption-Tolerant Networks

Quan Yuan, Ionut Cardei, and Jie Wu, *Fellow*, *IEEE*

Abstract—Routing is one of the most challenging, open problems in disruption-tolerant networks (DTNs) because of the short-lived wireless connectivity environment. To deal with this issue, researchers have investigated routing based on the prediction of future contacts, taking advantage of nodes' mobility history. However, most of the previous work focused on the prediction of whether two nodes would have a contact, without considering the time of the contact. This paper proposes predict and relay (PER), an efficient routing algorithm for DTNs, where nodes determine the probability distribution of future contact times and choose a proper next-hop in order to improve the end-to-end delivery probability. The algorithm is based on two observations: one is that nodes usually move around a set of well-visited landmark points instead of moving randomly; the other is that node mobility behavior is semi-deterministic and could be predicted once there is sufficient mobility history information. Specifically, our approach employs a time-homogeneous semi-Markov process model that describes node mobility as transitions between landmarks. Then, we extend it to handle the scenario where we consider the transition time between two landmarks. A simulation study shows that this approach improves the delivery ratio and also reduces the delivery latency compared to traditional DTN routing schemes.

Index Terms—Disruption-tolerant networks (DTNs), landmarks, time-related Markov model, prediction, routing.

1 Introduction

Wireless ad hoc networks have been traditionally modeled as connected graphs with stable end-to-end paths. However, for emerging wireless applications, such as sensor networks for wildlife tracking and MANETs operating in challenging environments [2], wireless links are short-lived and end-to-end connectivity turns out to be sporadic. Such phenomena are prevalent in disruption-tolerant networks (DTNs) [3], [4], [9], [12], where the connection between nodes in the network changes over time, and the communication suffers from frequent disruptions, making the network partially connected. The intermittent end-to-end paths and the changing topology make conventional MANET routing protocols fail, as they are designed with the assumption that the network stays connected. Routing in DTNs is an especially challenging problem because of the temporal scheduling element in a dynamic topology, which is not present in traditional MANETs. Nodes have to decide who the next hop is, but also when to forward, as they route packets to destinations in a store-and-carry way.

Researchers have proposed a number of broad methods to solve the above issue. In general, previous related works fall into three categories: *mobile resource-based*, *opportunity-based*, and *prediction-based*. In the first category [1], [18], systems employ mobile resources (data mules and mobile agents) as message ferries. These can be directed to pick up, move toward the next hop, and deliver messages to implement end-to-end store-and-carry message delivery. In opportunity-based schemes [22], [23], nodes forward messages during contacts that are unscheduled or random. For the prediction-based schemes [5], [7], internode contacts and mobility behavior are predicted, generally using prior contact history. The next hop and the contact in which a message is forwarded are selected using the predictions such that a quality of service (QoS) metric (e.g., delay or delivery ratio) is maximized. Most of the existing prediction-based routing protocols focus on the prediction of whether two nodes would have a contact in the future without considering *when* the contact occurs. We believe that lack of contact timing information undermines the contact prediction accuracy, and consequently reduces routing performance.

Based on our previous work [25], we propose predict and relay (PER), a routing method for DTNs that relies on predicting future contacts. We use a model based on a time-homogeneous semi-Markov process (TH-SMP) model to predict the probability distribution of the time of contact, and the probability that two nodes will have a contact in the future.

Our study is inspired by two observations from reality, pointed out in [10]. One observation is that nodes in a network within a social environment do not move completely at random. Instead, they usually move around a set of well-visited locations that we call landmarks [15], in this paper. Specifically, nodes show preference for a small number of landmarks and would move less often to the neighborhood of other landmarks [24]. While near a landmark that is visited by other users, a communication device may use the opportunity to establish contacts with other nodes and exchange messages. The second observation is that in some social environments, the node trajectory,

- *Q. Yuan is with the Department of Computing and New Media Technologies, University of Wisconsin-Stevens Point, Stevens Point, WI 54481. E-mail: ayuan@uwsp.edu.*
- *I. Cardei is with the Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431. E-mail: icardei@cse.fau.edu.*
- *J. Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122. E-mail: jiewu@temple.edu.*

Manuscript received 23 Jan. 2010; revised 16 July 2010; accepted 10 Aug. 2010; published online 3 May 2011.
Recommended for acceptance by P. Santi.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2010-01-0060.
Digital Object Identifier no. 10.1109/TPDS.2011.140.

in time is almost deterministic [17]. This means a node has its own mobility schedule and it generally moves between landmarks according to that schedule, subject to few random deviations. For example, a student on a campus moves between classrooms, dormitory, cafeteria, and gym. The dwell time at each landmark, and the landmark trajectory are fairly regular, with small variations. Nodes keep one schedule for a relatively long interval (e.g., a semester), so it can be assumed they operate in steady state with a few deviations.

The objective of our work is to explore the solutions to the routing problem in DTNs with a semi-Markov model. The main contributions of this paper are: 1) a landmark trajectory prediction method that uses a time-homogeneous semi-Markov process to determine the probability distribution of node arrival time at landmarks, 2) a method to determine a probabilistic *contact profile* that predicts inter-node contacts, and 3) a set of message forwarding rules that improve the message delivery ratio by controlling the selection of the contact in which a message is transmitted to the next hop. Simulation results show that our approach raises the delivery ratio using the improved contact prediction accuracy, compared with other traditional routing protocols. Furthermore, results show that PER also reduces the delivery latency in DTNs.

The remainder of this paper is organized as follows: in Section 2, we discuss the existing routing approaches in DTNs. Section 3 describes the overview of the predict and relay schemes. Section 4 presents the system model and detailed routing schemes in our protocol. Section 5 models transition time in node mobility based on our original model and proposes the extended prediction system. Section 6 provides simulation results. We conclude our work in Section 7.

2 Related Work

In the past, several routing schemes have been proposed to improve the routing performance in DTNs. This section reviews the related work in the past literature and highlights their differences. Due to limited space, we focus on results that inspired our work, or that are widely cited.

As mentioned before, there are three categories for current routing schemes [8], [19] in DTNs: *moving resources-based*, *opportunity-based*, and *prediction-based*. In the first category, systems usually employ extra moving resources, such as data mules and moving agents, as ferries for message delivery. Researchers in [18] present an architecture to collect data in sparse sensor networks, which uses data MULEs to pick up data from the sensors when in close range, buffer it, and drop off the data to wired access points. Similarly, in [1], buoys monitor the water quality in a lake, and onboard sensors relay measurements using nodes on tourist tour-boats and pleasure cruisers. Both of these approaches improve routing performance with additional mobile nodes, although controlling these resources leads to extra cost and overhead.

The opportunity-based schemes utilize neither the mobile resources nor the prediction methods for routing. Instead, messages are exchanged only when two nodes meet at the same place by chance. For example, Vahdat and Becker [22] use the epidemic routing scheme by flooding. Further, the ZebraNet [13] project applies such an approach to research on animal migrations and interspecies interactions. Data are flooded in the network and eventually reaches access points. Spray and Wait [20] protocol is a multicopy routing protocol that controls the flooding overhead by limiting the number of message copies distributed in the Spraying phase, and then relies on *direct delivery* when a message is transmitted to the final destination. It then waits until the destination meets one of them. Harras et al. [11] have improved and evaluated the controlled message flooding schemes with heuristics, for instance, on-hop limits or timeouts. Approaches falling into this category usually distribute multiple copies in the network to ensure a high reliability of delivery and a low latency. But, they also bring a high price in regards to the buffer occupancy and bandwidth.

In the prediction-based schemes, nodes' mobility is estimated based on a history of observations. A typical example is utility-routing [26], where each node maintains a utility value for every other node that is updated using the time between contacts. A node forwards a message copy only to nodes with a higher utility for the message destination. The utility value is considered as the predictor of two nodes' future likelihood of contact. In [6], Burns et al. propose a routing protocol that uses past frequencies of contacts, as well as the past contacts. LeBrun et al. [14] propose a routing algorithm for vehicular DTNs that uses the current position and trajectories of nodes to predict their future distance to the destination. In [5], researchers present MaxProp, a protocol that mainly relies on the prediction of the path likelihoods according to historical contact data. Protocol performance evaluations are conducted on 60 days' trace data from a real DTN network deployed on 30 buses. MobySpace [16] is another prediction-based generic algorithm for DTN routing that uses a high-dimensional euclidean space constructed upon nodes' mobility patterns. The frequency of visits of nodes to each possible location is recorded as the basis of the future distance calculation in the euclidean space. Most of these protocols focus on whether two nodes will have a contact without sufficiently considering *when* the contact occurs. Our approach, however employs a time-homogeneous semi-Markov process model to predict both the contacts and their time. We predict when two specified nodes have a contact based on their history information. Since time is considered, our future contact prediction is more accurate than the traditional ones.

3 Predict and Relay

We consider a DTN with a finite number of mobile nodes with unique IDs that move mostly between a set of landmarks. A landmark is defined as a place where nodes can communicate directly, i.e., any two nodes that are located at a landmark at the same time can establish a contact to exchange messages. Nodes at different landmarks cannot establish a contact. Landmarks are also assigned unique IDs.

As described in the introduction, the networks of social nature have nodes follow a semideterministic trajectory, with small deviations from a repetitive sequence of landmarks with constant dwell times. For illustration in this paper, we use a campus network where the landmarks are independent WLANs installed in classrooms and buildings,

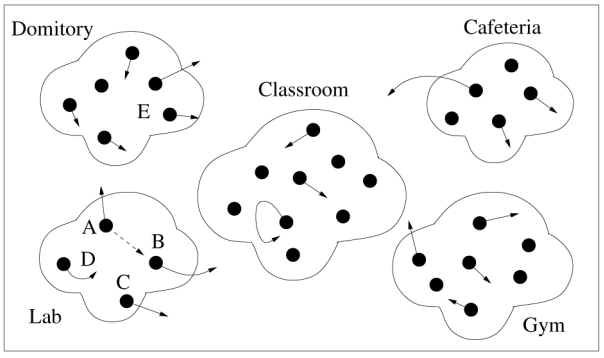


Fig. 1. An example of a landmark-based mobility model. The solid line denotes the mobility behavior of nodes. The dashed line stands for forwarding packets.

while nodes are the students and faculty with PDAs or laptops. We acknowledge that in the real world the WLANs are actually connected by a backbone. Another good example of this type of DTN is a network in a rural area where the landmarks consist of WLANs located in villages, interconnected by buses that carry messages on portable computers. A campus DTN is shown in Fig. 1. In both examples, nodes follow a scheduled route that is subject to random deviations.

In prediction-based routing schemes, history information is used to predict nodes' future mobility, which becomes the basis of the decision to forward messages toward the destination. Most of the previous prediction-based DTN routing methods predict whether two nodes would encounter each other, but consider *when* two nodes will meet insufficiently. We argue that figuring out when two nodes will meet with a probability distribution could improve the delivery ratio, as well as reduce the delivery latency.

PER is a single-copy DTN routing protocol—only a single instance of a message is forwarded toward the destination. Each message carries in its header a time-to-live (TTL) field. After the TTL expires, the message should be dropped. Messages are forwarded hop-by-hop in a succession of contacts using a greedy approach. At each forwarding step, PER selects the next hop that has the highest probability of delivery to the destination.

When a node u has to forward a message from its queue (e.g., at the beginning of a contact, or when a message is received from an application) it computes a probability metric $f(x)$ for all nodes currently in contact with u (the set N_u), and for itself, $x \in \{u\} \cup N_u$. This metric indicates the delivery performance to the destination if node u selects node x as the next hop and forwards the message to x.

The current node then selects the next-hop h as the node for which the delivery probability metric is maximized

$$h = \arg \max_{x \in \{u\} \cup N_u} f(x). \quad (1)$$

If the selected next hop is the current node ($h = u$), then the message will not be forwarded.

The formulation of the probability metric $f(x)$ is detailed in Section 4.3 and is based on predicting node mobility over a finite time horizon. In contrast with prior research that mainly focuses on estimating the probability of contact regardless of the contact time, our method uses landmark trajectory prediction to determine the probabilities of contact for each time unit. The prediction of nodes' future mobility relies on their trajectory history, that is recorded and disseminated throughout the network in an epidemic fashion. We believe this approach is feasible in a network that has reached steady state.

Fig. 1 gives an example of the PER process. Node A needs to send a message to node E. Located at the same landmark with node A are nodes B, C, D. Based on the history mobility information, A predicts that before the TTL expires, node B has a better delivery performance to node E than all other nodes in the Lab. Therefore, node A will forward the message to node B. In this scenario, node B will leave the Lab landmark later and will meet E in the $Classroom$ for delivery.

4 System Model

4.1 Assumptions

In this paper, we focus on the effectiveness of a time-based mobility prediction for DTN routing. Thus, we make some simplifying assumptions that will be addressed as part of our future work. We assume that during a contact, nodes can successfully transfer all messages needed to be forwarded. This requires reliable transport and high bandwidth or long enough contacts. These conditions can be more guaranteed when two nodes tend to dwell at the same landmark for at least several minutes in a DTN supported by 802.11a/g/n WLANs, like a campus environment.

Moreover, each landmark has a unique landmark id in the network, and nodes are aware of which landmark they are located at anytime. Also, we assume that the whole network is composed of the neighborhoods of landmarks, which means a node is always associated to a certain landmark in the network. Nodes spend zero time on the transition between landmarks.

4.2 TH-SMP Model

We model the mobility of node m with a time homogeneous semi-Markov, (X_n^m, T_n^m) with discrete time. The states are represented by the landmarks $L = 1, \ldots, l$. A node that moves between two landmarks transitions in the Markov process between the corresponding states. We assume the transition probabilities between states have the Markov memoryless property, meaning that the probability of a node m transiting from state X_i^m to state X_{i+1}^m is independent of state X_{i-1}^m. Thus, process (X_n^m) is a standard discrete-time Markov Chain. The random variable T_n^m represents the time instant of the transition $X_n^m \rightarrow X_{n+1}^m$. Random variable $T_{n+1}^m - T_n^m$ describes the landmark sojourn time, or state holding time. Note that the sojourn time does not include the time when the nodes are in transit between landmarks. These random variables are independent and identically distributed (i.i.d.), with distributions that do not change over time (time-homogeneous) and can be different from the geometric or the exponential distributions (semi-Markov).

The associated time-homogeneous semi-Markov kernel Q is defined by

$$\begin{aligned} Q_{ij}^m(t) =& P\big(X_{n+1}^m = j, T_{n+1}^m - T_n^m \le t | X_0^m, \ldots, X_n^m; \\ & T_0^m, \ldots, T_n^m\big) \\ =& P\big(X_{n+1}^m = j, T_{n+1}^m - T_n^m \le t | X_n^m = i\big). \end{aligned}$$

Suppose $P^m = |p_{ij}^m|$ is the transition probability matrix of the (X_n^m) embedded Markov chain for node m. Then, the transition probability from state i to state j is

$$p_{ij}^m = \lim_{t\to\infty} Q_{ij}^m(t) \quad i, j \in L.$$

Also, we derive the probability $S_i^m(t)$ that node m will leave the neighborhood of landmark i on or before time unit t

$$S_i^m(t) = P\big(T_{n+1}^m - T_n^m \le t | X_n^m = i\big) = \sum_{j=1}^{l} Q_{ij}^m(t). \quad (2)$$

Note that $S_i^m(t)$ also indicates the distribution of the dwell time at landmark i for node m, regardless of the next landmark.

Let $Z^m = (Z_t^m, t \in \bullet)$ be another TH-SMP that describes the state (landmark) occupied by node m at time t. The transition probabilities for process Z are $\phi_{ij}^m(t) = P(Z_t^m = j | Z_0^m = i)$. In the following, we drop the ^{m} superscript to simplify the notation. If we know that a node is currently in state i, after t time units, it will be in state j with probability $\phi_{ij}(t)$. ϕ provides the prediction of the node's location at a landmark at an arbitrary time $t > 0$ knowing its current location. The derivation of ϕ is described next.

For a fixed current state i, $\phi_{ij}(t)$ forms the probability mass function of the random variable that indicates the state at time t. Thus, $\sum_{j=1}^{l} \phi_{ij}(t) = 1$ for any initial state i and future time $t > 0$. For the border case $t = 0$, $\phi_{ij}(0) = \delta_{ij}$, where δ is the Kronecker symbol.

To determine $\phi_{ij}(t)$, we start with a special case when the process stays in state i between time 0 and t, with no transitions

$$\begin{aligned} & P(X_t = i | X_0 = i, T_1 \ge t) \\ & \quad = P(T_1 - T_0 \ge t | X_0 = i) = 1 - S_i(t). \end{aligned}$$

If the node transitions at least once between times 0 and t, we consider on the time k of the first transition from i, and on the state r to which the process moves immediately after state i. We obtain

$$\begin{aligned} & P(X_t = j | X_0 = i \text{ and at least one transition}) \\ & \quad = \sum_{r=1}^{l} \sum_{k=1}^{t-1} \dot{Q}_{ir}(k) \phi_{rj}(t-k), \end{aligned}$$

where $\dot{Q}_{ir}(k) = \frac{dQ_{ir}(k)}{dk} = Q_{ir}(k) - Q_{ir}(k-1)$ is the time derivative of Q, assuming a time step equal to the unit.

Putting it together, we obtain

$$\phi_{ij}(t) = (1 - S_i(t))\delta_{ij} + \sum_{r=1}^{l} \sum_{k=1}^{t-1} \dot{Q}_{ir}(k) \phi_{rj}(t-k). \quad (3)$$

We first note that ϕ can be calculated iteratively, as $\phi_{ij}(t)$ depends on probabilities $\phi_{lj}(t-k)$ computed in the previous steps.

Specifically, since we consider the time discrete in our model, (3) is rewritten as follows:

$$\begin{aligned} \phi_{ij}(k) &= P(Z_k = j | Z_0 = i) \\ &= d_{ij}(k) + \sum_{r=1}^{l} \sum_{\tau=1}^{k} v_{ir}(\tau) \phi_{rj}(k-\tau), \end{aligned} \quad (4)$$

where $d_{ij}(k) = (1 - S_i(k))\delta_{ij}$, $v_{ij}(k) = \dot{Q}_{ij}(k)$, and $k \in \bullet$. Furthermore, $v_{ij}(k)$ can be approximated by the following equation:

$$v_{ij}(k) = \begin{cases} Q_{ij}(1), & \text{for } k = 1, \\ Q_{ij}(k) - Q_{ij}(k-1), & \text{for } k > 1. \end{cases}$$

Using the assumption that the landmark dwell time random variables are independent from the embedded state transition process (X_{ij}), we derive

$$\begin{aligned} Q_{ij}(k) &= P(X_{n+1} = j, T_{n+1} - T_n \le k | X_n = i) \\ &= P(X_{n+1} = j | X_n = i) \\ &\quad \cdot P(T_{n+1} - T_n \le k | X_{n+1} = j, X_n = i) \\ &= p_{ij} S_{ij}(k). \end{aligned}$$

$S_{ij}^m(k)$ is the probability that node m will move from landmark i to landmark j at, or before time k. The time parameter k can be used to represent a relative time offset. Based on the Markov property of the underlying processes, if the state i of a node is known at a time k_0, than at time $k > k_0$, the probability of that node being in state j is $\phi_{ij}(k - k_0)$.

With sojourn time probability distributions S_{ij}^m and the transition probability matrix P^m, we can predict the future landmark location of node m based on its current location using probability distributions $\phi_{ij}^m(k)$. Section 4.4 describes how to derive these probabilities.

4.3 Contact Probabilities

In this section, we propose additional metrics to be used for studying various probabilistic delivery probability metrics $f(x)$ during the packet forwarding defined in Section 3.

Distributions $\phi_{ij}^m(k)$ give the probability that the future location at time k of a node m will be j considering that at time 0 the location was landmark i. Assuming that trajectories of nodes are independent of each other and that the most recent known state of node a is s_a (at time k_a), and for node b is s_b (at time k_b, with $k_a < k, k_b < k$), the probability of contact between a and b at a landmark i at time k is

$$C_{ab}^i(k) = \phi_{s_a i}^a(k - k_a) \cdot \phi_{s_b i}^b(k - k_b) \text{ for } k > 0.$$

Then, the probability that a and b are in contact at a time k at any landmark is

$$C_{ab}(k) = \sum_{i \in L} C_{ab}^i(k) \quad \text{for } k > 0. \quad (5)$$

Note that $C_{ab}(k)$ does not define a proper probability mass function, as $0 \le \sum_k C_{ab}(k) \overset{\le}{>} 1$.

For our study of probabilistic delivery probability metrics $f(x)$, we define the probability that two nodes begin their first contact at time k. Note that when we talk about nodes a and b beginning their first contact at time k, it means that they had no contacts in any prior time units in a considered interval. Assuming that node trajectories are

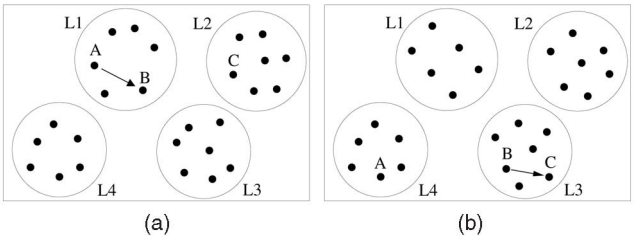


Fig. 2. An example illustrating the process of PER. (a) Network topology at Time t. (b) Network topology at Time $t + \Delta t$.

independent, the probability of the first contact at time k is defined as

$$R_{ab}(k) = C_{ab}(k) \prod_{t=0}^{k-1} (1 - C_{ab}(t)) \quad \text{for } k > 0. \tag{6}$$

Denote the maximum message acceptable delivery delay by D, which means packets are required to reach the destination in time D. Moreover, Let n_c be the chosen neighbor for evaluating the delivery probability metrics, and d is the destination. The prediction metric functions are defined as follows:

Function 1. This prediction metric function is defined in terms of the maximum probability of contact in time [1, D], which is defined as

$$f_1 = \max_k C_{n_c d}(k), \quad 1 \le k \le D. \tag{7}$$

Function 2. We define Function 2 based on the maximum average probability of contact in time [1, D], which is

$$f_2 = \sum_{k=1}^{D} C_{n_c d}(k). \tag{8}$$

Function 3. For this prediction metric function, we mainly focus on the first contact probability. Thus, maximum probability of the first contact before the deadline is the basis of Function 3, which is

$$f_3 = \sum_{k=1}^{D} R_{n_c d}(k). \tag{9}$$

Relay node selection is done based on the above prediction functions. For each message that is taken from the queue during a contact, the prediction metric is computed using only one of the three prediction metric functions for each neighbor (another node that is in contact with the current node). In a greedy approach, the node with the highest metric value is picked as the relay node to forward the packet. Intuitively, the chosen neighbor should have the largest contact probability with the destination in the future D time steps. We refer to the PER algorithms using Functions 1, 2, and 3 as PER1, PER2, and PER3, respectively. Once the relay node is determined, the message is forwarded to it. If the selected node is the current node itself, then the message will be kept in the queue for a later transmission. The corresponding algorithm is shown in Algorithm 1.

Algorithm 1. Predict and Relay Algorithm

1. Node exchanges and updates the history mobility information of other nodes with its neighbors.
2. Node uses one of the three prediction functions (PER1, PER2, and PER3) to pick a neighbor as the next hop.
3. Node forwards the packet to the chosen node in 2.

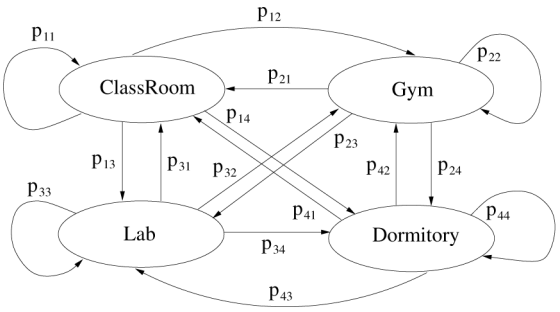


Fig. 3. The Markov model of node m's mobility.

Fig. 2 is an example that illustrates the PER algorithm, where node A sends a packet to node C at time t. Note that at time t, node A is at landmark $L1$ and node C is at landmark $L2$. With the history mobility information, A first uses one of the three prediction metric functions, for example, f_1, to find the neighbor node B which is most likely to meet the destination node C in the future. Then A relays the packet to B. After Δt, B carries the packet and moves to landmark $L3$, where it meets node C. Finally, C gets the packet from B.

4.4 Deriving Mobility Parameters

4.4.1 Two Parameters

To calculate the prediction function f, PER needs to compute two parameters: the transition probability matrix P^m and the sojourn time probability distribution matrix, $S^m_{ij}(k)$ for each node m. We retrieve these two parameters from node mobility history.

P^m is the transition probability matrix of the embedded Markov chain for node m. Fig. 3 shows an example transition probability matrix for node m that visits four landmarks: Classroom, Gym, Dormitory, and Lab. At any of those landmarks, the node could pick to stay for a while, or move to another landmark according to its preferred probability. For example, if the node is at the gym, it can then

1. move to the Lab with probability p^m_{43};
2. or stay in the gym with the probability p^m_{44};
3. or go to the dormitory with probability p^m_{42}; and
4. or head for the Classroom with probability p^m_{41}.

Those mobility probabilities constitute the transition probability matrix P^m. Note that each node has its own transition probability matrix that reflects its trajectory preference.

We define the P^m probabilities as follows:

Definition 1. *The probability p^m_{ij} that node m moves from landmark i to landmark j is defined as the observed transition frequency*

$$p^m_{ij} = num^m_{ij} / num^m_i,$$

where num^m_i stands for the number of transitions from landmark i without considering the next landmark, and num^m_{ij}

is the number of transitions from landmark i to landmark j. Obviously, $num_{ij}^m \leq num_i^m$ and $p_{ij}^m \leq 1$. By keeping track of num_i^m and num_{ij}^m, each node could generate and refine its own P matrix over time.

We calculate the sojourn time probability distribution $S_{ij}^m(k)$, $k \in \bullet$ as follows:

Definition 2. *The sojourn time probability distribution at landmark i when followed by a transition to landmark j, $S_{ij}^m(k)$, is defined as*

$$S_{ij}^m(k) = P\big(t_{ij}^m < k\big),$$

where t_{ij}^m is the sojourn time or state holding time at landmark i when j is the next visited landmark. We assume that when the network reaches steady state, the mobility history provides a representative sample from which the sojourn time distribution can be drawn. Therefore, the probabilities $P(t_{ij}^m < k)$ are computed with the equation

$$P\big(t_{ij}^m < k\big) = \sum_{n=0}^{k-1} P\big(t_{ij}^m = n\big). \tag{10}$$

In Markov processes, the sojourn time is usually considered to have an exponential distribution. Our use of a semi-Markov model eliminates this constraint and is more reflective of real world processes.

Computing probabilities $P(t_{ij}^m < k)$ is relatively simple. For example, node m can measure all times t_{ij}^m whenever it moves from landmark i to landmark j. In that way, the distribution of the sojourn time probability is a discrete distribution. For instance, assume that we have six measurements for t_{ij}^m, which are 2, 4, 4, 5, 4, 6. Then, the $P(t_{ij}^m < 5)$ is 2/3.

Next, we describe how a node can determine the sojourn time distributions for all other nodes in the network.

4.4.2 History Information Exchange

To predict node mobility in the PER algorithms, every node needs to know other nodes' mobility history information. Specifically, the history mobility information is defined as a 5-tuple $\langle nodeID, P, S, T_{rec}, LandmarkID_{cur}\rangle$, where P is the transition probability matrix, S is the sojourn time probability distribution matrix, T_{rec} is the recording time when the record is generated, and $LandmarkID_{cur}$ is the recorded landmark where the node is located when the record is generated. Whenever two nodes become neighbors, they will exchange the history mobility records they have. A node adds the record of its new neighbor into its local database, and updates the history information with the new data by comparing the parameter T_{rec}. Note that it is possible that nodes only have a partial view of the whole network. However, when a node calculates $\phi_{ij}(k)$ for the neighbor and the destination, it is very possible that T_{rec} from their records can be different, which means the start time for computation is different. Therefore, the equations for prediction in the above are no longer correct. To improve the accuracy of predictions under this scenario, we utilize the following modifications.

When node A wants to send a packet to node B, node A first looks up B's history mobility information locally, which is $\langle nodeID_B, P_B, S_B, T_{recB}, LandmarkID_{curB}\rangle$. If A needs to know where node B is at time t, the following equation is used

$$\phi_{LandmarkID_{curB}j}(t - T_{recB}) = P(Z_t = j | Z_{T_{recB}} = LandmarkID_{curB}) \quad for\ j \in L,$$

where Z_t is the landmark where the node is at time t.

Similarly, node A can adjust the calculation of $\phi_{ij}(k)$ for a neighbor. The only difference is that it finds the neighbor's latest record, and replaces the start time with the recording time T_{rec} in the record, as well as the start landmark with the recorded landmark $LandmarkID_{cur}$ in the record. Note that the predicted time window for those two nodes may be different in this way, to make sure that the mobility behavior at the same future time spot is predicted.

The reader will notice that the size per node of the P matrix ($|L|^2$), and especially the S matrix could be large—$|L|^2 H$ for S, where H is the prediction window. Nevertheless, we expect that in the real world these matrices will be very sparse due to the typical routine found in DTNs with social nature, where the node trajectory is almost deterministic with small deviations, such as the network of students in a campus or in public transportation.

5 PER with Transition States

In the previous chapters, we have assumed the DTN model where nodes spend zero time units on traveling between landmarks. However, in practical situations, a routing protocol should be able to consider a node's transition time between landmarks. For example, it does take time for a student to move from gym to classroom. Such transition time lessens the precision of our prediction model, and consequently, degrades the practical routing performance of PER protocols. In this section, we extend our prior prediction model by introducing the transition state concept. We argue that although the change is subtle, our new model is more applicable, and it is not trivial from the design side because of the following issues. First of all, the formal and detailed mechanism to model the node behaviors and predict node contacts with the transition states is needed. Second, because of the transition states, memory usage is enlarged to collect and store history information for predictions. Thus, a memory usage optimization is required to improve system performance.

5.1 Prediction Model Extension

The main idea to extend our prediction model is that we introduce the transition states, and refine the prediction model. A transition state is identified by its starting landmark and ending landmark, which indicates that a node is traveling from its starting landmark to its ending landmark. Note that we assume that nodes can contact only when they are at the same landmark. If there are l landmarks in the network, the number of transition states is $l^2 - l$, and there are l^2 states for a node in total, including the l landmark states.

Before presenting the refined prediction model, we first need to index the states for a node. In our previous model, we can simply model nodes' states based on the manually assigned landmark IDs. l landmarks indicate that a node has l states, however, as we mentioned, there are $l^2 - l$

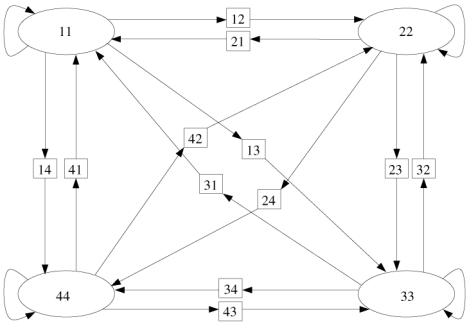


Fig. 4. The Markov model of a node's states in a network with four landmarks. The square indicates a transition state between two landmarks, and the ellipse is the state at a landmark.

transition states, and indexing those transition states with the unique manually assigned IDs turns out to be inefficient and unpractical. To solve this issue, we present the following indexing system for a node's states. Let us use the $ii(1 \le i \le l)$ to denote the l landmark states first. For a transition state, if its starting landmark is ii and the ending landmark is jj, the transition state is represented by ij. Then, we utilize $1, 2, \ldots, l^2$ to index the landmark states and the travel states in the order of $11, 12, \ldots, 1l, 21, 22, \ldots, 2l, \ldots, ll$. For instance, the landmark state denoted by 33 is indexed as $2 \times l + 1$, and the travel state denoted by 23 is indexed as $l + 3$. With the above indexing system, mobility behaviors of a node can be modeled as a Markov process, as shown in Fig. 4. In this way, the transition states can automatically be indexed only with the given landmark IDs.

With the above states indexing system, we refine our prediction model for transition states as follows: we still use the same notations in our previous work. Obviously, we have $Q_{ij}(k) = p_{ij}S_{ij}(k)$. Note that $S_i(k)$, the probability that the node will leave state i in time k also indicates the distribution of the residence time at state i. Specifically, when state i is a landmark state, $S_i(k)$ means the residence time distribution at state i. If state i is a transition state, $S_i(k)$ is the traveling time distribution between two landmarks. Thus, we have,

$$S_i(k) = P(T_{n+1} - T_n \le k | X_n = i) = \sum_{j=1}^{l^2} Q_{ij}(k). \tag{11}$$

We can then derive the probability $\phi_{ij}(k)$, which is the probability that a node is at state i at time k if it was at state j at time 0 as follows:

$$\begin{aligned}\phi_{ij}(k) &= (1 - S_i(k))\delta_{ij} + \sum_{r=1}^{l^2} \int_0^k \phi_{rj}(k-\tau) dQ_{ir}(\tau) \\ &= \left(1 - \sum_{j=1}^{l^2} Q_{ij}(k)\right)\delta_{ij} + \sum_{r=1}^{l^2}\sum_{\tau=1}^{k} \dot{Q}_{ir}(\tau)\phi_{rj}(k-\tau).\end{aligned}$$

Since δ_{ij} is the Kronecker δ, and $\dot{Q}_{ij}(k)$ can be approximated with $Q_{ij}(k)$, $\phi_{ij}(k)$ could be determined by the mobility history information p_{ij} and $S_{ij}(k)$. Then, we can utilize the $\phi_{ij}(k)$ as the basis of our routing protocol for the DTN model with transition states.

Since nodes can only talk to each other when they are at the same landmark, contacts occur at the state with index $i \times l (1 \le i \le l)$ based on our states indexing system. Consequently, we need to modify the contact probability definition as follows: the contact probability that node a and b are in contact at a time k at any landmark is,

$$C^*_{ab}(k) = \sum_{i=1}^{l} C^{(il)}_{ab}(k) \quad \text{for } k > 0. \tag{12}$$

The probability of the first contact at time k is defined as,

$$R^*_{ab}(k) = C^*_{ab}(k) \prod_{t=0}^{k-1} (1 - C^*_{ab}(t)) \quad \text{for } k > 0. \tag{13}$$

The three probability metric functions are similar. The only difference is replacing the contact probability C with C^*, and the first contact probability R with R^*.

5.2 Memory Optimization

Note that $\sum_{r=1}^{l^2} p_{ir} = 1$, and each node has its own transition probability matrix. As mentioned before, p_{ij} can be calculated as the ratio of the number of transitions from state i to state j to the number of transitions from state i without considering the next state. However, considering that there are l^2 states in the network, counting the number of transitions for each state is very resource consuming. In addition, we need to establish a l^4 matrix to hold the transition probability for every single node in the network. We note that the state transition probability matrix P is a sparse matrix and most of its elements are 0. For example, because each travel state has only one possible successor state, which is its ending landmark, moving from a travel state to other states is impossible. Therefore, we define a condensed matrix and a mapping function to take the place of P, in order to save the overhead and memory. In that way, we can only maintain the condensed matrix for storing history information, and map it to P when we need to calculate the contact probabilities for packet forwarding.

We define the landmark transition probability matrix $\bar{P} = |\bar{p}_{ef}| (1 \le e, f \le l)$ as the target condensed matrix. In $\bar{P}$, an element is the probability that a node moves from one landmark to another regardless of travel states. For example, $\bar{p}_{24}$ indicates the probability that a node moves from landmarks 2 to 4. There are $l \times l$ elements in matrix $\bar{P}$, and $\sum_{f=1}^{l} \bar{p}_{ef} = 1 (1 \le e, f \le l)$. In addition, the landmark index e in $\bar{P}$ represents the landmark with the state index $e \times l$ in P. Observing that a node at a travel state can only head for its ending landmark, and a node at a landmark state can only go to its successor travel state, we define the following mapping function between matrix $\bar{P}$ and P.

Definition 3. *The mapping function between $\bar{P}$ and P is defined as follows:*

$$p_{ij} = \begin{cases} \bar{p}_{ef}, & if\ i = e \times l + e, \\ & and\ j = e \times l + f, \\ 1, & if\ j = (i\%l) \times l + i\%l, \\ & \quad and\ i \ne j, \\ 0, & other\ i, j. \end{cases} \tag{14}$$

The method of generating $\bar{P}$ is similar to the one we use for P. With the mapping function and the landmark transition probability matrix $\bar{P}$, we could reduce the size of the stored transition probability matrix from l^4 to l^2.

For other matrices used in our prediction model, like S, Q, and Φ, we expect in the real world these matrices to be very sparse due to the typical routine found in semideterministic DTNs with a social nature, such as the network of students on a campus or in public transportation. Thus, basic schemes to store a sparse matrix, such as coordinate format and compressed sparse row format, could be utilized to optimize the memory usage in the system.

6 Performance Evaluation

In this section, we would like to evaluate our three PER algorithms, and contrast their performance against several simple single-copy routing algorithms and epidemic routing with custom simulations and real trace analysis. Our main objective is to investigate whether the three PER algorithms can increase the delivery ratio in comparison to other single-copy routing algorithms. Also, we want to see how the three PER algorithms provide better end-to-end delivery latency.

Routing protocols. We have implemented and compared the following routing schemes. Note that the names in the parentheses are used to refer to the routing schemes in result plots. We mainly focus on the single-copy schemes in our simulation study.

Epidemic routing ("Epidemic"): a node would spread the message it has to any nodes it encounters that have not seen it yet. Eventually, every node in the network will obtain a copy of that message. This protocol relies on multicopy delivery, such that messages can reach the destination on multiple paths. We implement this approach to investigate the optimal end-to-end delivery latency between two nodes.

Utility-based routing ("Utility"): each node maintains a utility value for every other node in the network, based on a timer indicating the time elapsed since the two nodes last encountered each other. Here, the smaller the elapsed time is, the bigger the utility value will be. A node would forward the message only to the neighbor which has the larger utility for the destination.

Random selection ("Random"): a node would randomly pick a neighbor as a relay node to forward the message until the message reaches its destination.

Direct delivery ("Direct"): the source does not forward the message to anyone unless it encounters the destination.

PER ("PERX"): a node would distribute a message with the schemes described in this paper. Since we have three different criteria to indicate whether to forward messages for a node, we employ PER1, PER2, and PER3 to refer to our three schemes, respectively.

Message generation. We use the poisson distribution to model message generation in the network. In detail, we regulate that the average message arrival rate in the network is 10. For each message, we randomly select a pair of nodes as source and destination, respectively.

6.1 Simulation Settings

We have used a custom packet-based simulator implemented in Java to evaluate and compare the performance of the different routing protocols. Discrete time is used in our simulation.

In our simulation, we establish a landmark based DTN model, where there are several predefined landmarks in the network. Nodes usually revolve around these landmarks. That is, nodes would stay in the neighborhood of a landmark, or move to the neighborhood of other landmarks with their own preferred probability. Two nodes can only communicate when they are associated with the same landmark. In our case, we simulate scenarios with 12 landmarks and 30 nodes. Initially, nodes are uniformly distributed among the landmarks. Moreover, we assume that every node has a trajectory deviation probability p, which is used to simulate nodes' semideterministic mobility behavior. That is, every node has a probability $1-p$ to visit a landmark from where it is currently, and visit any other $|L|-1$ landmarks with probability $p/(|L|-1)$. We also regulate the probability that a node moves from landmark i to landmark i is 0, $P_{ii}=0$. p is a simulation parameter that varies from 0 to 0.5 with a step of 0.1.

We define the maximum sojourn time that a node moves from a landmark to another as $2+w$, where w is called the *sojourn time width window* and varies from 0 to 20 time units. The sojourn time of a node is uniformly picked in $[2, 2+w]$. In all scenarios considered, each message is assigned a TTL value of 40 time units. The prediction time window for the three PER algorithms is fixed to 60 time units. The choice of the time unit really depends on the network scenario and applications. If the granularity of the time unit is too big, say 1 hour, then we may miss the nodes' location change information. On the other hand, if the granularity is too small, say 1 sec, then we may introduce unaffordable memory usage when doing the prediction calculation. For example, in a campus network, 5 minutes could be a reasonable option as the time unit. In our simulation, we specify the time unit as 5 minutes. To apply our PER algorithms, a node needs to generate its transition probability matrix P and sojourn time probability distribution $S_{ij}(k)$ first. Note that at the beginning, the acquired P and $S_{ij}(k)$ are not stable, since the collected mobility history information to generate those two parameters is not sufficient. Therefore, to better evaluate the system performance, we run the simulation for a "warm-up period" to reach steady state, and collect sufficient history mobility information to generate stable P and $S_{ij}(k)$. After that, the simulator runs 2,048 time units for each scenario to collect data. We then run each scenario ten times to report the average. Considering the average message arrival rate is 10, we believe that in 2,048 time units a sufficient number of messages can be generated for retrieving the stable average data tendency.

The common goals of any DTN routing protocol is to maximize the delivery ratio, and to minimize the delivery latency. Therefore, to evaluate the performance, we use the following metrics: 1) *Delivery ratio* is defined as the ratio of the number of successfully delivered messages to the number of all messages generated in the network; and 2) *Delivery latency* is the average end-to-end delivery latency between a pair of source and destination nodes in the network.

6.2 Results in the Simplified DTN Model

First, we compare the delivery ratio and the delivery latency of the three PER algorithms with other approaches under

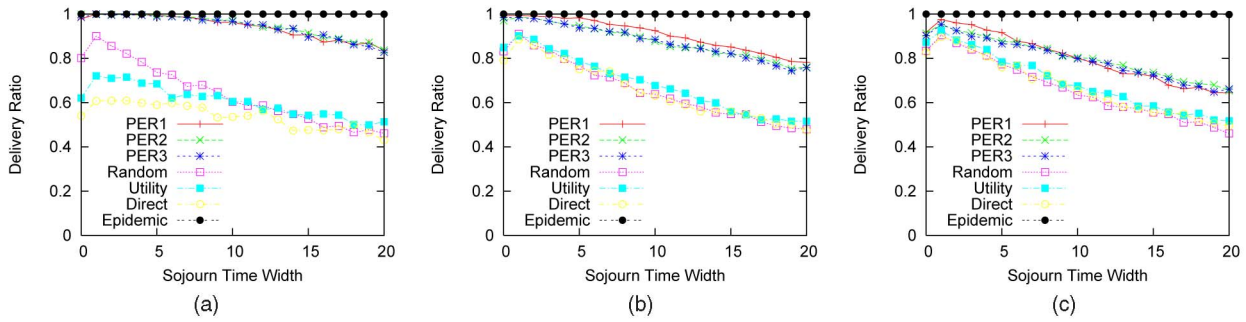


Fig. 5. Comparison of delivery ratio under different trajectory deviation probability p. (a) Trajectory deviation, $p = 0.0$. (b) Trajectory deviation, $p = 0.2$. (c) Trajectory deviation, $p = 0.5$.

three different p scenarios with 0.0, 0.2, and 0.5, respectively. In the $p = 0.0$ case, nodes' mobility is very deterministic. Therefore, the nodes' future mobility prediction becomes accurate. For $p = 0.5$, the network's entropy is higher as nodes move more randomly and the trajectory prediction becomes less accurate. Case $p = 0.2$ is on the middle level. We just want to evaluate how the three PER algorithms perform compared to other schemes under those three scenarios. Fig. 5 plots the delivery ratio under the three different p scenarios with w varying. We see that all three PER algorithms gain larger delivery ratios than utility-based routing, random selection routing, and direct delivery approach. This is because, the three PER algorithms have better mobility prediction when forwarding messages, which could increase the possibility to contact the destination. Also, as w increases, the delivery ratios of all the routing schemes go down. A common reason is that because as w rises, message delivery delay increases, even beyond the predefined TTL, which downgrades the delivery ratio. But for the three PER algorithms, another reason is that when w increases, the accuracy of $S_{ij}(k)$ falls, which then affects the accuracy of the future mobility prediction. Besides, we find that, in $p = 0.0$, PER algorithms perform much better than other routing schemes while in $p = 0.5$, the advantage is reduced and all routing schemes work very similar. In other words, the more deterministic the nodes' mobility behavior is, the better PER performs.

Fig. 6 records the delivery latency under the three different p scenarios where w changes. Although the three PER algorithms are designed to increase the delivery ratio in routing, their performance in terms of delivery latency is better too, compared to other routing protocols. We can see that in all three scenarios, the PER algorithms have less delivery latency, even though, as p increases, the delivery latency of the three PER algorithms becomes closer to other protocols. Thus, we believe that accurate predictions during messages' forwarding process could also help reduce the delivery latency. In addition, as w goes up, the delivery latency of the three PER algorithms is raised. The reason is obvious. Because nodes become less mobile, it will cost more time to deliver messages with nodes' mobility.

We then compare the performance of the three PER algorithms with other approaches, under three different w with 0, 10, and 20, respectively. As explained before, w controls the width of the uniform distribution from which the node sojourn time is sampled. The intent behind this experiment is to check how those routing protocols perform under different nodes' mobility. Note that a larger w implies that nodes are less mobile. Thus, $w = 0$ means nodes transfer frequently among the neighborhoods of the landmarks in the network while for $w = 20$, nodes' transition between nodes occurs less often, on average each $(2 + 20)/2 = 11$ time units. In Fig. 7, we show the delivery ratio under the three different w scenarios where p varies. In all three scenarios, the three PER algorithms present better delivery ratio than other protocols. Additionally, as p increases, which implies that node mobility becomes less deterministic, the delivery ratio of the three PER algorithms falls down. But, p does not influence the delivery ratio of other routing protocols. This trend indicates that when the node mobility is less random applying the PER protocols yields better results.

Fig. 8 summarizes the end-to-end delivery latency of all routing protocols under the three different w scenarios. We

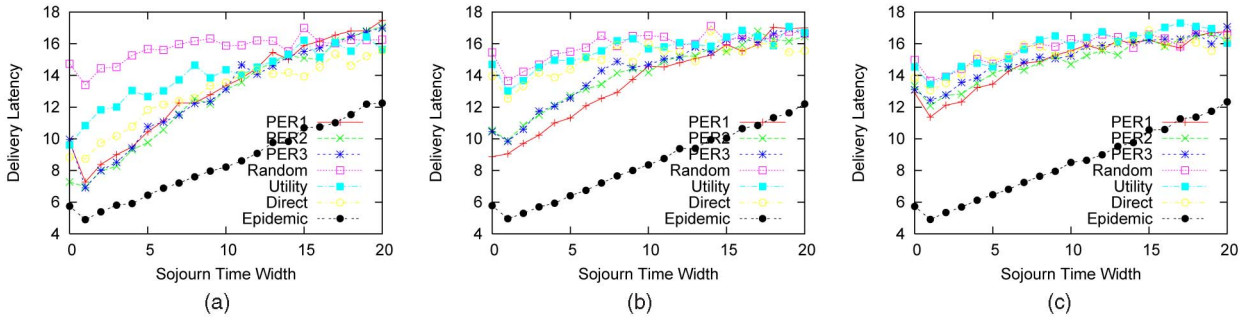


Fig. 6. Comparison of delivery latency under different trajectory deviation probability p. (a) Trajectory deviation, $p = 0.0$. (b) Trajectory deviation, $p = 0.2$. (c) Trajectory deviation, $p = 0.5$.

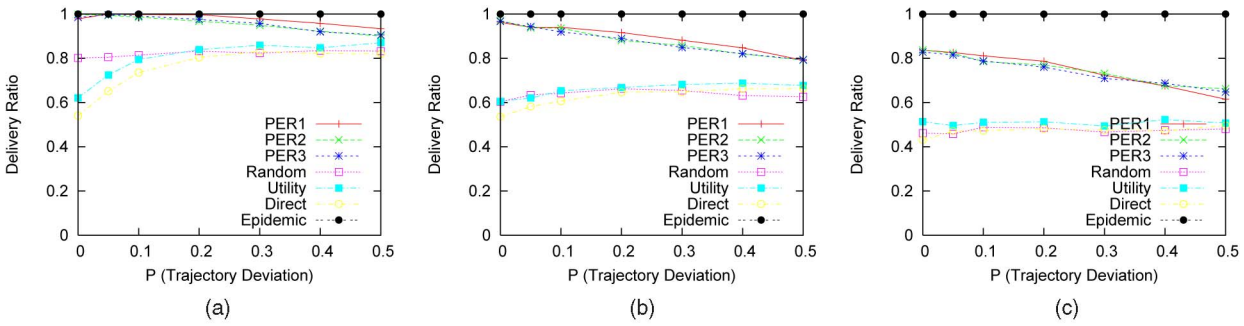


Fig. 7. Comparison of delivery ratio of routing protocols under different sojourn time window width w. (a) Sojourn time width, $w = 0$. (b) Sojourn time width, $w = 10$. (c) Sojourn time width, $w = 20$.

find that the higher the node mobility is, and the less random the transition times are (lower w), the lower the latency is for PER protocols. When the width of the transition time window grows ($w = 20$), the PER prediction accuracy falls, and the time spent by messages in queues grows. As a consequence, the latency of PER protocols grows comparable to that of the other protocols. Overall, PER algorithms are more effective when the randomness of the node trajectories is low, as seen in scenarios with reduced trajectory deviation probability (p) and sojourn time window width (w). In addition, similar to the trend in the delivery ratio figures, p does not drastically influence the delivery latency of other routing protocols. However, as p grows, which indicates that mobility becomes less deterministic, we see that the delivery latency of our three PER algorithms goes down.

6.3 Stanford Trace Analysis

The Stanford trace [21] collects a 12-week log of a local-area wireless network installed throughout the Gates Computer Science Building of Stanford University. The building is L-shaped (the longer edge is called the a-wing, and the shorter the b-wing). It has four main floors with offices and labs, a basement with classrooms and labs, and a fifth floor with a lounge and a few offices. Each of the main floors has two access points, one for each wing. Additionally, the library, which spans both the second and third floors, also has an access point. The basement has two access points, one near the classrooms and one for the Interactive Room, a special research project in the department. The smaller fifth floor only has one access point. The wireless user community roughly consists of 74 users including students, staff, and faculty. The authors collected three separate types of traces during a 12-week period, encompassing the 1999 Fall quarter. Here, we use the SNMP trace. That is, every two minutes, the router queries, via Ethernet, all twelve access points for the MAC addresses of the hosts currently using that access point as a bridge to the wired network. Once we know which access point a MAC address uses for network access, we know the approximate location (floor and wing) of the device with that MAC address. We pair these MAC addresses with the link level addresses saved in the packet headers to determine the approximate locations of the hosts in the tcpdump trace. Since the trace only records nodes' location information at a specified time stamp, and retrieving the transition time between landmarks is not sufficiently accurate, here we analyze the Stanford trace based on the simplified DTN model which does not consider the transition time.

Because of the memory constraint, we select seven landmarks and 30 nodes from the trace. Our data processing includes the following steps. 1) To introduce more mobility in the network without losing nodes' location change information, we use 4,000 original time slots as a time unit in our trace analysis. Since original Stanford data records nodes' behaviors from time stamp 51,281 to 7,275,539, our total simulation time turns to be 2,000 time units correspondingly. 2) We divide the access points into seven landmarks, which are five floors, library, and basement. Nodes can talk to each other only when they are at the same landmark. 3) We find that in the original trace there are some nodes barely moving. Such nodes are simply removed from our analysis because our

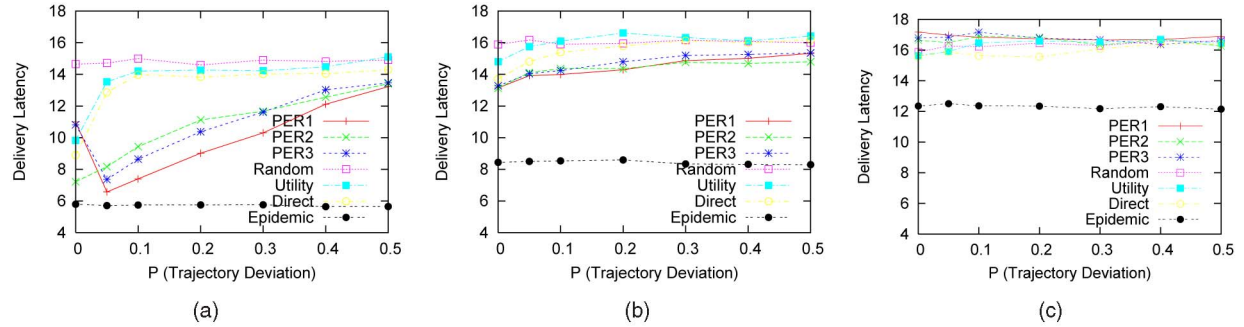


Fig. 8. Comparison of delivery latency of routing protocols under different sojourn time window width w. (a) Sojourn time width, $w = 0$. (b) Sojourn time width, $w = 10$. (c) Sojourn time width, $w = 20$.

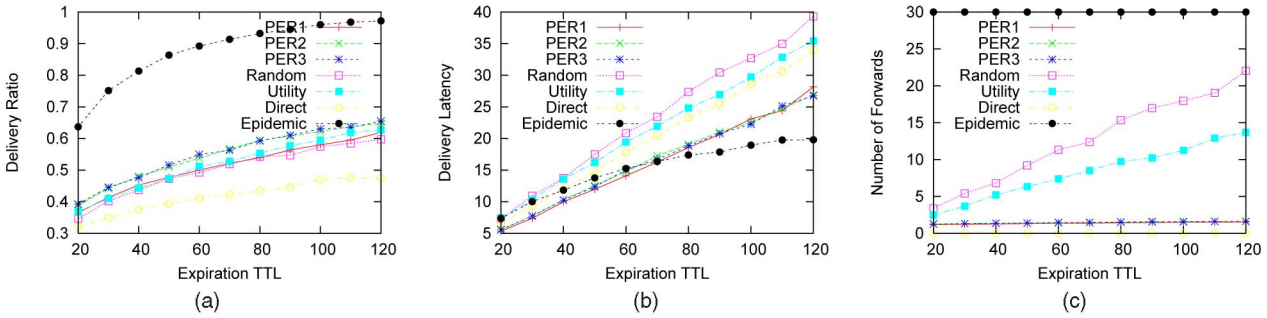


Fig. 9. Comparison of routing protocols under Stanford trace with different TTL. (a) Delivery ratio. (b) Delivery latency. (c) Number of forwards.

objective is to anticipate nodes' future movements, and predicting still nodes' mobility is trivial. Also, for some nodes, there are holes in the trace. A trace hole for a node means that the node disappears in the trace for quite a long time because of some restrictions in the trace collection. To make the trace more consistent, we solve the trace hole by repeating the node's history movement during its disappearing time. For each simulation, nodes were uniformly selected to be the source or destination of a packet. The message packet generation is modeled by the poisson distribution, with the average arrival rate 10 time units. All packets have an expiration TTL ($20 \leq$ TTL ≤ 120), which represents the delay requirement. The prediction window is 60 time units. Each simulation was repeated 20 times with different random seeds for statistical confidence. Besides delivery ratio and delivery latency, we also look at the total number of forwards in our analysis, which reflects the overhead in terms of the number of times a packet forward occurred in the DTN.

Fig. 9a, 9b, and 9c show the results of delivery ratio, delivery latency, and number of forwards for different protocols, respectively. In Fig. 9a, the delivery ratio climbs as the TTL is raised, and the delivery ratio of the epidemic routing represents the upper bound. We see in this trace case, PER2 and PER3 outperform other single-copy routing protocols on delivery ratio, while PER1 does not have an obvious advantage compared to others. The reason could be that the prediction metric function 1 only focuses on the maximum contact probability on a time unit, while functions 2 and 3 consider the contact probability of a period, which is more reliable in reality. Our PER protocols and utility-based routing have better delivery ratio than the random selection due to their own forwarding indicator. In Fig. 9b, latency increases as the delay requirement of the packet lessens because more long-lived packets are successfully delivered. All three of our PER protocols have less delivery latency then other single-copy protocols due to the efficient probability prediction. We also notice that when TTL ≤ 70, the delivery latency of the epidemic routing is not the smallest. The reason is that epidemic routing delivers more packets with a larger delivery ratio, and these packets bring larger average delivery latency. When the TTL grows larger than 70, there is less of an impact, and the epidemic routing dominates with the smallest delivery latency.

In Fig. 9c, the total number of forwards of our three PER protocols are less than 2, which indicates that the PER protocols can pick the relatively suitable neighbor to relay message packets by using predictions, and save the forwarding times as a result. Since the prediction window is fixed at 60 time units, more predictions and forwardings are conducted when the TTL climbs. Thus, when the TTL varies from 20 to 120, the number of forwards of our three PER protocols increases from 1.1 to 1.6, correspondingly. Epidemic routing simply floods packets in the network, which will drain the bandwidth and storage of the network. Such a feature makes it impractical in reality, even though it can maximize the delivery ratio and minimize the delivery latency. Because random selection just simply forwards packets without any indicator, its number of forwards is larger than any other single-copy routing protocol.

6.4 Results in the DTN Model with Transition States

We use the DTN model with transition states as presented in Section 5, where the extended PER (denoted by PER+) is employed for routing in this simulation. We set landmark trajectory deviation p as 0.2. The time that nodes spend on the transition state when moving between two landmarks is uniformly distributed in $[3, 5]$. We define the maximum landmark state residence time as $3 + w'$, where w' is called the *landmark state residence time width*. Note that an extreme case is that the landmark state residence time is far larger than the transition time, where the transition time can be ignored. Such a case is already taken care of by our simplified DTN model. Considering that, here we vary w' only from 4 to 8 time units. The landmark state residence time of a node at a specified landmark is uniformly selected in $[3, 3 + w']$. Note that large w' brings more mobility uncertainty, and indicates that nodes may spend more time at landmarks. In our simulation, we have seven landmarks in the simulated network. For performance evaluation, we consider three different network scale scenarios, with 4, 12, and 30 nodes, respectively. The 4 nodes scenario is a sparse network case, while the 30 nodes case indicates more mobility because of the increasing node density. The scenario with 12 nodes represents the middle level. Other configurations are the same as described in the simplified DTN model. Due to the limited space, we only implement PER1 and PER1+. Moreover, we use the epidemic routing algorithms to compare the performance of PER1 and PER1+. The routing performance in terms of delivery ratio and latency will be evaluated.

Fig. 10 plots the delivery ratio under different w' for the three different network scenarios. We see that PER1+ gains a larger delivery ratio than PER1 approach. This is because, our routing algorithm has better prediction of future

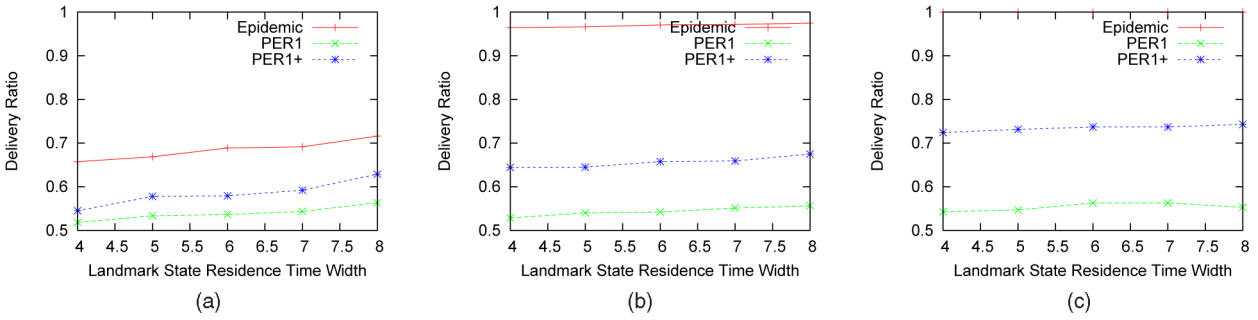


Fig. 10. Comparison of delivery ratio under different landmark state residence time width. (a) Number of $\mathrm{nodes} = 4$. (b) Number of $\mathrm{nodes} = 12$. (c) Number of $\mathrm{nodes} = 30$.

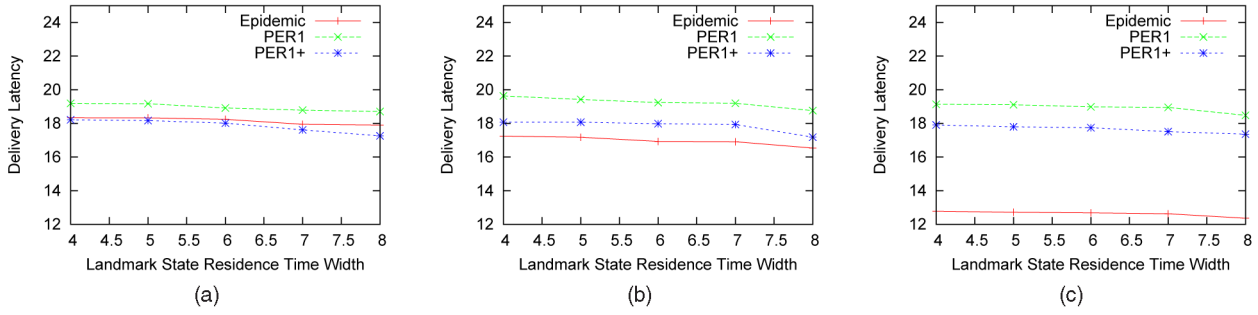


Fig. 11. Comparison of delivery latency under different landmark state residence time width. (a) Number of $\mathrm{nodes} = 4$. (b) Number of $\mathrm{nodes} = 12$. (c) Number of $\mathrm{nodes} = 30$.

contacts. Different from the simplified DTN model results, as w' increases, the delivery ratios of almost all the routing schemes are going up. The reason is that as nodes stay longer at landmark state while the traveling time is fixed, there would be more contact opportunities. Besides, we find that in the sparse network with 4 nodes, even the epidemic routing cannot guarantee 90 percent delivery ratio. When the number of nodes increases to 12 and 30, epidemic routing can achieve almost 100 percent delivery ratio, while the delivery ratio of our protocols is also positively raised. This is because more nodes bring more mobility in the network, which eventually increases the message delivery opportunities.

Fig. 11 summarizes the end-to-end delivery latency comparison under different w'. We find that PER1+ has better deliver latency than PER1. Specifically, in the sparse network scenario, the delivery latency of our algorithm can even beat the epidemic routing at some point. But we cannot say that our algorithm has lower average delay than the epidemic routing solution, because the delivery ratio of the epidemic routing is much better in the above case, meaning more large latency messages are delivered successfully, which may eventually raise the average delivery latency. As the number of nodes increases to 30, the delivery latency of epidemic routing dramatically decreases because of the sufficient contacts in the network. Additionally, when w' grows, nodes have more contact opportunities, and more suitable message carriers could be reached and employed to forward packets. Thus, the protocol PER1+ can save more time on delivering message packets to the destination, and as a result the delivery latency is slightly reduced. Overall, PER+ is more efficient than PER according to the results, especially under the sparse network scenario.

7 Conclusions

In this paper, we propose the Predict and Relay scheme, an efficient routing scheme in DTNs. We introduce a time-homogeneous semi-Markov process model to predict the future contacts of two specified nodes at a specified time. With this model, a node can select a proper neighbor as the next hop to forward the message. This paper defines three different prediction functions to assist in choosing the proper neighbor for message delivery. Simulation results show that our approach raises the delivery ratio, as well as reduces the delivery latency, compared to other traditional DTN routing protocols.

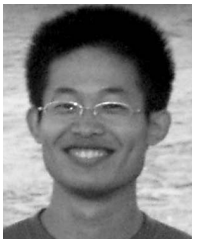
Acknowledgments

This work was supported by NSF grants CNS 0847664, CNS 0626240, CCF 0830289, and CNS 0948184.

References

[1] "Sensor Networking with Delay Tolerance (SeNDT)," http://down.dsg.cs.tcd.ie/sendt/, 2011.
[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Comm. Magazine*, vol. 40, no. 8, pp. 102-114, Aug. 2002.
[3] A. Balasubramanian, B.N. Levine, and A. Venkataramani, "DTN Routing as a Resource Allocation Problem," *Proc. SIGCOMM*, 2007.
[4] N. Banerjee, M. Corner, and B. Levine, "An Energy-Efficient Architecture for DTN Throwboxes," *Proc. IEEE INFOCOM*, 2007.
[5] J. Burgess, B. Gallagher, D. Jensen, and B.N. Levine, "Maxprop: Routing for Vehicle-Based Disruption-Tolerant Networks," *Proc. IEEE INFOCOM*, 2006.
[6] B. Burns, O. Brock, and B.N. Levine, "Mv Routing and Capacity Building in Disruption Tolerant Networks," *Proc. IEEE INFOCOM*, 2005.

[7] I. Cardei, C. Liu, J. Wu, and Q. Yuan, "DTN Routing with Probabilistic Trajectory Prediction," *Proc. Int'l Conf. Wireless Algorithms, Systems and Applications (WASA '08)*, 2008.
[8] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, "Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Encounter Ages," *Proc. Fourth ACM Int'l Symp. Mobile Ad Hoc Networking and Computing (MobiHoc '03)*, 2003.
[9] K. Fall, "A Delay-Tolerant Network Architecture for Challenged Internets," *Proc. SIGCOMM*, 2003.
[10] J. Ghosh, S.J. Philip, and C. Qiao, "Sociological Orbit Aware Location Approximation and Routing (Solar) in DTN," Technical Report 2005-27, State Univ. of New York at Buffalo, Apr. 2005.
[11] K. Harras, K. Almeroth, and E. Belding-Royer, "Delay Tolerant Mobile Networks (DTMNs): Controlled Flooding Schemes in Sparse Mobile Networks," *Proc. IFIP Netwoking*, 2005.
[12] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble Rap: Social Based Forwarding in Delay Tolerant Networks," *Proc. Ninth ACM Int'l Symp. Mobile Ad Hoc Networking and Computing (MobiHoc '08)*, 2008.
[13] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, "Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with Zebranet," *Proc. 10th Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS-X)*, 2002.
[14] J. LeBrun, C. Chuah, and D. Ghosal, "Knowledge Based Opportunistic Forwarding in Vehicular Wireless Ad Hoc Networks," *Proc. IEEE 61st Vehicular Technology Conf. (VTC)*, 2005.
[15] K. Lee, M. Le, J. Haerri, and M. Gerla, "Louvre: Landmark Overlays for Urban Vehicular Routing Environments," *Proc. IEEE 68th Vehicular Technology Conf. (VTC)*, 2008.
[16] J. Leguay, T. Friedman, and V. Conan, "Evaluating Mobility Pattern Space Routing," *Proc. IEEE INFOCOM*, 2006.
[17] C. Liu and J. Wu, "Routing in a Cyclic Mobispace," *Proc. Ninth ACM Int'l Symp. Mobile Ad Hoc Networking and Computing (MobiHoc '08)*, 2008.
[18] R. Shah, S. Jain, S. Roy, and W. Brunette, "Data Mules: Modeling a Three-Tier Architecture for Sparse Sensor Networks," Technical Report IRS-TR-03-001, Intel Research Seattle, 2003.
[19] T. Small and Z.J. Haas, "Resource and Performance Tradeoffs in Delay Tolerant Wireless Networks," *Proc. ACM SIGCOMM Workshop Delay-Tolerant Networking (WDTN '05)*, 2005.
[20] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks," *Proc. ACM SIGCOMM Workshop Delay-Tolerant Networking (WDTN '05)*, 2005.
[21] D. Tang and M. Baker, "CRAWDAD Data Set Stanford/Gates (v. 2003-10-16)," http://crawdad.cs.dartmouth.edu/stanford/gates, Oct. 2003.
[22] A. Vahdat and D. Becker, "Epidemic Routing for Partially Connected Ad Hoc Networks," Technical Report CS-200006, Duke Univ., 2000.
[23] J. Wu, M. Lu, and F. Li, "Utility-Based Opportunistic Routing in Multi-Hop Wireless Networks," *Proc. 28th Int'l Conf. Distributed Computing Systems (ICDCS '08)*, 2008.
[24] J. Yoon, B. Noble, M. Liu, and M. Kim, "Building Realistic Mobility Models from Coarse-Grained Traces," *Proc. Fourth Int'l Conf. Mobile Systems, Applications and Services (MobiSys '06)*, June 2006.
[25] Q. Yuan, I. Cardei, and J. Wu, "Predict and Relay: An Efficient Routing in Disruption-Tolerant Networks," *Proc. 10th ACM Int'l Symp. Mobile Ad Hoc Networking and Computing (MobiHoc '09)*, 2009.
[26] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, "Performance Modeling of Epidemic Routing," *Proc. IFIP Networking*, 2006.



Quan Yuan received the BS and MS degrees in computer science in 2003 and 2006, from Huazhong University of Science and Technology, Wuhan, China. He is an assistant professor at the Department of Computing and New Media Technologies, University of Wisconsin-Stevens Point. His research interests include mobile computing, routing protocols, peer-to-peer computing, and Parallel and Distributed Systems.

Ionut Cardei is an assistant professor at the Department of Computer Science and Engineering, Florida Atlantic University. His research interests include the area of Wireless networking, Component-based software design automation, and Quality of Service and resource management in computer networks.

Jie Wu is a professor and the chairman at the Department of Computer and Information Sciences, Temple University. He has published more than 200 papers in various journal and conference proceedings. His research interests include the area of mobile computing, routing protocols, fault-tolerant computing, and interconnection networks. He is a fellow of the IEEE.