DTN-FLOW: Inter-Landmark Data Flow for High-Throughput Routing in DTNs

Kang Chen and Haiying Shen
Department of Electrical and Computer Engineering
Clemson University, Clemson, SC 29631
Email: {kangc, shenh}@clemson.edu

***Abstract*—In this paper, we focus on the efficient routing of data among different areas in Delay Tolerant Networks (DTNs). In current algorithms, packets are forwarded gradually through nodes with higher probability of visiting the destination node or area. However, the number of such nodes usually is limited, leading to insufficient throughput performance. To solve this problem, we propose an inter-landmark data routing algorithm, namely DTN-FLOW. It selects popular places that nodes visit frequently as landmarks and divides the entire DTN area into sub-areas represented by landmarks. Nodes transiting between landmarks relay packets among landmarks, even though they rarely visit the destinations of these packets. Specifically, the number of node transits between two landmarks is measured to represent the forwarding capacity between them, based on which routing tables are built on each landmark to guide packet routing. Each node predicts its transits based on its previous landmark visiting records using the order-k Markov predictor. In a packet routing, a landmark determines the next hop landmark based on its routing table, and forwards the packet to the node with the highest probability of transiting to the selected landmark. Thus, DTN-FLOW fully utilizes all node movements to route packets along landmark paths to their destinations. We analyzed two real DTN traces to support the design of DTN-FLOW. We also deployed a small DTN-FLOW system in our campus for performance evaluation. This deployment and trace-driven simulation demonstrate the high efficiency of DTN-FLOW in comparison with state-of-the-art DTN routing algorithms.**

I. Introduction

Delay tolerant networks (DTNs) are featured by intermittent connection and frequent network partition. Thus, DTN routing is usually realized in a carry-store-forward manner [1], which makes it possible to develop useful applications over such challenging environments. Among many applications, we are particularly interested in those that exchange data among or collect data from different areas because DTNs usually exist in areas without infrastructure networks and thereby are good mediums to realize data communication among these areas. For example, researchers have proposed to provide data communications (i.e., Internet access) to remote and rural areas [2] by relying on people or vehicles moving among rural villages and cities to carry and forward data. The concept of DTN has also been applied in animal tracking [3], which collects logged data from the digital collars attached to zebras in Kenya without infrastructure network, and in environmental monitoring, where data is collected by sensors deployed on animals in mountain areas.

Since these applications tend to transmit high volumes of data (i.e., Internet data, collected logs and monitoring data), a key hurdle in developing these applications is efficient data routing with high throughput. This challenge is non-trivial due to the DTN features mentioned previously. Epidemic routing [4], in which each node forwards its packets to all of its neighbors, can route packets to their destinations quickly. However, its broadcast nature consumes high energy and storage resource, which makes it unsuitable for the resource-limited DTNs [5]. Other DTN routing algorithms can generally be classified into three groups: probabilistic routing [6]–[9], social network based routing [10]–[15] and location based routing [16]–[21]. They exploit either past encounter records, social network properties or past moving paths to deduce a node's probability of reaching a certain node or area, and forward packets to nodes with higher probability than current packet holder. Figure 1(a) illustrates the routing process in previous routing algorithms. A packet is generated in area $A1$ for area $A9$. It is first carried by node N_1 and then is forwarded to N_2 in area $A3$ since N_2 visits $A9$ more frequently. Later, similarly, the packet is forwarded to N_3, which finally carries the packet to area $A9$. Since the number of nodes with high probability of visiting the destination usually is limited, by only relying on such nodes, previous routing algorithms fail to fully utilize all node movements, leading to degraded efficiency and overall throughput. For example, even if there are many node moving between $A2$ and $A5$, they are not utilized to forward the packet.

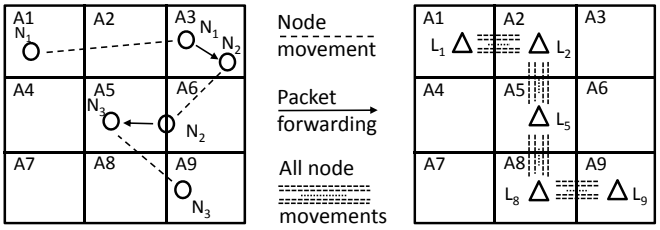


Fig. 1: Comparison between previous methods and DTN-FLOW.

To deal with this problem, we propose an inter-landmark data flow routing algorithm, called DTN-FLOW, that fully utilizes all node movements in DTNs. Figure 1(b) demonstrates DTN-FLOW. We assume that there is a popular place in each of the nine subareas in Figure 1(a). DTN-FLOW then determines landmarks from these popular places and adopts the same sub-area division as in Figure 1(a). Each sub-area is represented by one landmark. Each landmark is configured with a central station, which is an additional infrastructure with

high processing and storage capacity. Then, node movement can be regarded as transits from one landmark to another landmark. DTN-FLOW utilizes such transits to forward packets one landmark by one landmark to reach their destination areas. Nodes transiting between landmarks relay packets, even though they rarely or may even not visit the destinations of the relayed packets. We denote the landmark in each area in Figure 1(b) by L_i ($i \in [1, 9]$). For packets originated from area $A1$ targeting area $A9$, they are forwarded along landmark L_1, L_2, L_5, L_8 to finally reach L_9. Thus, DTN-FLOW fully utilizes the node transiting between different pairs of landmarks for data transmission (i.e., $A2$ and $A5$ in previous example), thereby increasing the data flow throughput.

DTN-FLOW measures the amount of nodes moving from one landmark, say L_i, to another landmark, say L_j, to represent the inter-landmark forwarding capacity from L_i to L_j. This capacity indicates the data transfer capacity between landmarks, hence is similar to the concept of "bandwidth" for physical wired or wireless links. With the measured capacity, each landmark uses the distance-vector method [22] to build its routing table that indicates the next hop landmark to reach each destination landmark. DTN-FLOW predicts node transits based on their previous landmark visiting records using the order-k Markov predictor. Then, in packet routing, each landmark determines the next hop landmark based on its routing table, and forwards the packet to the node with the highest probability of transiting to the selected landmark. Thus, DTN-FLOW fully utilizes the node transits to forward packets along landmark paths with the shortest latency to reach their destinations. The routing table is built with the Distance-vector method [22], in which each landmark informs all neighboring landmarks its routing table periodically. Upon receiving a routing table, each landmark updates its routes to other landmarks with the next landmark hop leading to a shorter delay. In DTN-FLOW, the transmission of all information (i.e., routing tables and packets) among landmarks is conducted through mobile nodes.

Therefore, the proposed DTN-FLOW is suitable for applications designed to transfer data among different areas in DTNs where distributed frequently visited locations can be identified without extreme effort. Also, since the landmark requires a lightweight additional infrastructure, these locations should be able to provide continuous energy supply. Two common scenarios that satisfy above requirements are the the data exchange between different buildings in campus and between different villages in rural areas. In both scenarios, mobile nodes (people) usually frequently visit different buildings/villages that can provide energy supply easily.

We further analyzed two real DTN traces to confirm the shortcoming of the current routing algorithms and to support the design of DTN-FLOW. We also deployed a real DTN-FLOW system in our campus using 9 mobile phones for the real-world performance evaluation. This real deployment and extensive trace-driven simulation demonstrate the high throughput and efficiency of DTN-FLOW in comparison with state-of-the-art routing methods in DTNs.

The remainder of this paper is arranged as follows. Section II and III present related work and network model and real trace analysis. Section IV introduces the detailed design of the DTN-FLOW system. In Section V, the performance of DTN-FLOW is evaluated through extensive trace-driven experiments and a real deployment. Section VI concludes this paper with remarks on our future work.

II. Related Work

A. Probabilistic Routing Methods

Probabilistic routing methods [6]–[9] use nodes' past encounter records to predict their future encounter probabilities, which is used to rank the suitability of a node to carry a packet. PROPHET [6] updates the encountering probability between two nodes when they meet and ages the probability over time. A packet is always forwarded to nodes with higher probability of meeting its destination. MaxProp [7], RAPID [8], and MaxContribution [9] extend PROPHET by further specifying forwarding and storing priorities based on the probability of successful delivery. Packets with higher priorities are forwarded first, and high priority packets replace low priory packets when a node's storage is full.

B. Social Network Based Routing Methods

Considering that people carrying mobile devices usually belong to certain social relationships, social network based routing algorithms [10]–[15] exploits social network properties in DTNs for packet routing. MOPS [10] is a publish-subscribe system. It groups frequently encountered nodes into a cluster for efficient intra-community communication and selects nodes having frequent contacts with foreign communities for inter-community communication. BUBBLE [11] uses two layers of ranks: global and local. The global ranking is used to forward a packet to the destination community, and the local ranking helps to find the destination within the community. The concept of two-layer routing is similar to the landmark overlay in DTN-FLOW. However, BUBBLE does not split the network into sub-areas but classifies nodes into different communities. SimBet [12] adopts centrality and similarity to rank the suitability of a node to carry a packet. It is based on the concept that nodes having high centrality and similarity with the destination node tend to meet it frequently. The event dissemination system in [13] is similar to MOPS. It groups well-connected nodes into communities and selects nodes with the highest closeness centrality as brokers for inter-community dissemination. Costa *et al.* proposed a social network based publish-subscribe system [14]. It forwards messages to nodes that meet subscribers of the packet's interest category frequently and have high connectivity with other nodes. HiBop [15] defines node context by jointly considering various information, including personal interests, residence and work, and forwards packets to the nodes that have frequent encounter records with the context of the destination.

C. Location Based Routing Methods

Location based routing methods [16]–[19], [21] use previous geographical location visiting records to predict future movement and thereby decide the suitability of a node to carry a packet destined for a certain node or place. GeoDTN [16] encodes historical geographical movement information in a vector to predict the possibility of two nodes becoming neighbors. Then, packets are forwarded to nodes that are more likely to be a neighbor of the destination node. DTFR [17] first decides the area that the destination node is expected to appear based on its past location and mobility pattern, then routes the packet to the area, and finally spreads the packet across the area to reach the destination node. PGR [18] uses observed node mobility pattern to predict nodes future movement to forward packets to a certain geographical destination. GeoOpps [19] exploits the navigation system to forward packets to certain geographical destinations in vehicle networks. It calculates the minimal estimated time of delivery (METD) by considering the closest point of possible routes to the destination, and forwards packets to vehicles that lead to smaller METD. In MobyPoints [21], a node's meeting probabilities with all possible locations are encoded in vectors. Then, forwarding decisions are made based on the similarity score between the vectors of relay node and destination node. LOUVRE [23] is a similar work with DTN-FLOW that it also builds landmarks on road intersections and uses the landmark overlay for routing in vehicle networks. However, LOUVRE focuses on vehicular networks that it relies on GPS and map to determine the current landmark a node is connected to and the next landmark the node is moving toward. On the contrary, DTN-FLOW is designed for general DTNs, in which it is hard to know the next landmark a node is moving to since nodes move freely in the whole area. To solve this problem, DTN-FLOW adopts a k-order Markov predictor and a novel method to handle prediction errors, as shown in Section IV-B and IV-D.

III. Network Model and Trace Analysis

A. Network Model

1) Network Description: We assume a DTN with mobile nodes denoted by N_i. Each node has limited storage space and communication range. We select landmarks, denoted by L_i ($i = 1, 2, 3, \cdots, M$), from places that nodes visit frequently. Then, the entire DTN area is split into sub-areas based on landmarks, each of which is represented by a landmark. We configure a central station at each landmark, which has higher processing and storage capacity than mobile nodes and can cover its whole sub-area. As in other social network based DTN routing algorithms [10]–[15], DTN-FLOW also assumes the existence of social network structure in DTNs. Such social structures determine node movement, leading to re-appearing visiting patterns to these landmarks.

A transit means a node disconnects from one landmark and connects to another landmark in sequence. We denote the *transit link* from landmark L_i to landmark L_j as T_{ij}. For a transit link, say T_{ij}, we define the *bandwidth* as the average number of nodes transiting from L_i to L_j in a unit time (T), denoted by B_{ij}. For simplicity, we assume that each packet has a fixed size. Our work can be easily extended to packets with various lengths by dividing a large packet into a number of the same-size segments.

2) Packet Routing: In this paper, we focus on the routing of packets to a landmark/sub-area. Then, normal probabilistic routing algorithms can be used to route packets to a specific location or node. DTN-FLOW can also employ the Cellular IP protocol [24] to track node locations and forward the packet to the new landmark containing the node. We leave the detail of the two extensions as our future work.

3) Differences with Infrastructure Networks: Though DTN-FLOW presents similar overlay as the infrastructure network (i.e., small sub-areas covered by landmarks), they have significant differences. Firstly, DTN-FLOW does not require landmarks to be inter-connected with fixed links. Rather, landmarks rely on the mobil nodes moving between them to relay packets. Secondly, as we will show later, each landmark only functions as a special relay node in the packet routing. Therefore, landmarks do not bring about server-client structure but keep the ad hoc nature of DTNs. Unlike base stations, landmarks are just static nodes in DTNs with higher processing capacity (e.g., a desktop).

This also means that the import of landmarks into DTNs only needs to build some fixed nodes in the network, which does not incur much extra effort. Though the landmark nodes require higher capacity in storage and energy compared to normal mobile nodes, these requirements can be easily satisfied since landmarks can have continuous energy supply (i.e., solar or wind energy). It is easy to maintain landmark nodes since there is no global management responsibility or control information stored on landmarks. When a landmark malfunctions, we can simply remove it and merge its area to that of another landmark or replace it without network-level re-configuration.

4) Purpose of Landmarks: In DTN-FLOW, landmarks function as "routers" in the network. Each landmark decides the neighboring landmark to forward its received packets. Neighboring landmarks are connected by "links" that take mobile nodes as the transfer media to carry packets. Without landmarks, packets are relayed purely through mobile nodes when they meet with each other, which may suffer from the uncertainty of node mobility. Therefore, landmarks make the DTN routing more structured (i.e., along landmarks) in the network dynamism in DTNs. Moreover, as previously mentioned, landmarks do not incur too much extra cost in terms of installing and maintenance. In summary, the use of landmarks in DTN-FLOW can better utilize node mobility in DTNs for efficient packet routing with a low extra cost.

B. Trace Analysis

In order to better understand how nodes transit among different landmarks in DTNs, we analyzed two real DTN traces collected from two different scenarios: students on campus and buses in the downtown area of a college town.

1) Empirical Datasets:

Dartmouth Campus Trace (DART) [25]. DART recorded the WLAN Access Point (AP) association with digital devices carried by students in the Dartmouth campus between Nov. 2, 2003 and Feb. 28, 2004. We preprocessed the trace to fit our investigation. We regarded each building as a landmark and merged neighboring records referring to the same node (mobile device) and the same landmark. We also removed short connections ($< 200s$) and nodes with fewer records (< 500). Finally, we obtained 320 nodes and 159 landmarks.

DieselNet AP Trace (DNET) [26]. DNET collected the AP association records from 34 buses in UMass Transit from Oct. 22, 2007 to Nov. 16, 2007 in Amherst, MA. Each bus carried a Diesel Brick that constantly scanned the surrounding area for open AP connections, and a GPS to record its GPS coordinators. Since there are many APs in the outdoor testing environment, some of which are not from the experiment, we removed APs that did not appear frequently (< 50) from the trace. We mapped APs that are within certain distance ($< 1.5km$) into one landmark. Similar to the processing of the DART trace, neighboring records refereing to the same node (bus) and the same landmark were merged. Finally, we obtained 34 nodes and 18 landmarks.

The key characteristics of the two traces are summarized in Table I. We then measured the landmark visiting distribution and the transits of mobile nodes among landmarks.

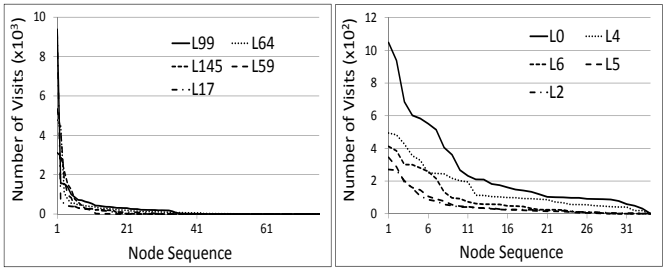
TABLE I: Characteristics of mobility traces.

	DART Campus (DART)	DieselNet AP (DNET)
# Nodes	320	34
# Landmarks	159	18
Duration	119 days	20 days
# Transits	477803	25193
# Transits per day	2401	1257

2) Landmark Visiting Distribution: We first measured the number of each node's visits to each landmark. Due to page limit, we only show the visiting distribution of the 5 most visited landmarks in the two traces in Figure 2(a) and Figure 2(b), respectively. Nodes failing to visit these landmarks throughout the trace were omitted. We see that in both traces, for each of the top 5 landmarks, only a small portion of nodes visit it frequently. For example, in the DART trace, less than 15 out of 320 nodes have high visiting frequencies to landmarks. Though not shown in the figures, such a finding holds for almost all landmarks in the two traces. Thus, we obtain the first observation (O):

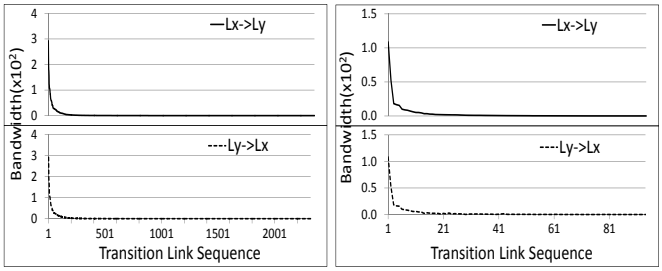
O1: *For each sub-area, only a small portion of nodes visit it frequently.*

This observation matches our daily experience that a department building in a campus usually is mainly visited by students in the department, and a bus station may only be visited by buses that stop at it. Such a finding validates our claim in the introduction section that the number of nodes frequently visiting the destination area is limited, which leads to degraded throughput performance of previous routing algorithms that only rely on such nodes for packet forwarding.



(a) DART. (b) DNET.

Fig. 2: Visiting distribution of top 5 most visited landmarks.



(a) DART. (b) DNET.

Fig. 3: Bandwidth distribution of transit links.

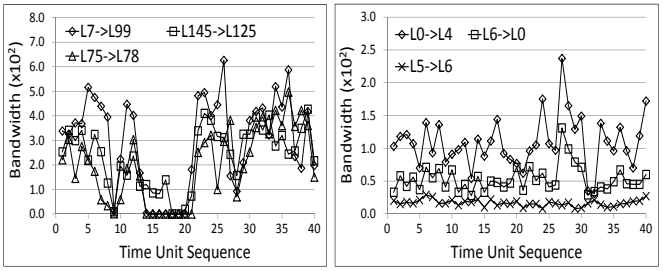
3) Transits among Landmarks: We refer to two transit links containing the same landmarks but have different directions (e.g., T_{ij} and T_{ji}) as *matching transit links*. We then measured the bandwidths of all transit links in the two traces and ordered them in decreasing order. We label two matching transit links with the same sequence number and plot them in two separated sub-figures, as shown in Figure 3(a) and Figure 3(b). Transit links with 0 bandwidth were omitted. From the two figures, we make the following two observations.

O2: *A small portion of transit links have high bandwidth.*

O3: *The matching transit links are symmetric in bandwidth.*

We also measured the bandwidth of all transit links in the two traces along time. The time unit was set to 3 days and 0.5 day in the two traces, respectively, resulting in a total of 40 time units for both traces. Due to page limit, we only present the results of the 3 highest bandwidth transit links in Figure 4(a) and Figure 4(b). Figure 4(a) shows that except two periods of time units [7, 10] and [14, 21], the measured bandwidth of each transit link fluctuates around its average value slightly over time. We checked the calendar and found that the two periods of time matches the Thanksgiving and Christmas holidays, which means that few students moved around in the campus. In Figure 4(b), we see that the measured bandwidth of each transit link is more stable around its average bandwidth than in the DART trace. This is because that 1) the DNET trace excludes holidays and weekends, and 2) bus mobility is more repetitive over time than human mobility. Also, both figures show that though there are some fluctuations, the bandwidth difference relationship of the three transit links remains the same most of the time. We then derive:

O4: *The bandwidth of a transit link measured in a certain time period can reflect its overall bandwidth.*



(a) DART. (b) DNET.

Fig. 4: The transit distribution of top 3 highest bandwidth transit links.

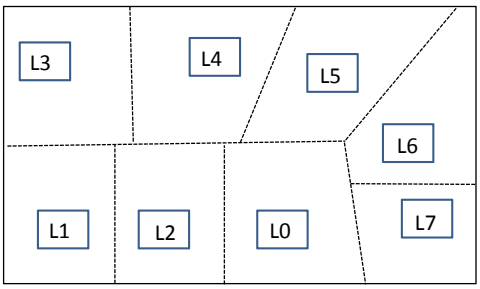


Fig. 5: Sub-area division in our campus deployment.

IV. System Design

In this section, we introduce the detailed architecture of our DTN-FLOW system based on above observations. It has four main components: (1) landmark selection and sub-area division, (2) node transit prediction, (3) routing table construction, and (4) packet routing algorithm. Component (1) provides general guidelines to select the location of landmarks and split the DTN into sub-areas. Component (2) predicts the next landmark a node is going to visit based on its previous visiting records. Such predictions are used to forward packets and exchange routing tables among landmarks. Component (3) measures the data transfer capacity between each pair of landmarks, based on which the routing table is built indicating the next hop landmark for each destination landmark and associated estimated delay. With the support of the first two components, component (4) determines the next-hop landmark and the forwarding node in packet routing.

A. Landmark Selection and Sub-area Division

The landmark selection determines the places to install landmarks. Sub-area division divides the entire to sub-areas with each area having one landmark. Both landmark selection and sub-area division are conducted by the network administrator or planner who hopes to utilize the DTN for a certain application.

1) Landmark Selection: As previously introduced, we select popular places that are frequently visited by mobile nodes as landmark places. Therefore, the network administrator first needs to identify popular places. A simple way is to collect node visiting history and take top N_v most frequently visited places as popular places. Popular places in DTNs with social network structures can be pre-determined based on node mobility pattern in the social network. For example, in the DART network, we can easily find popular buildings that students visit frequently: library, department building, and dorm. In DTNs in rural areas, villages are naturally popular places. In the DTNs using animals as mobile nodes for environment monitoring in mountain areas, places with water or food usually are often visited.

The popular places resulted from both the two ways are put into the candidate list for landmarks. There may be several popular places in a small area. Thus, not every popular place needs to be a landmark. Therefore, for every two candidate landmarks with distance less than D_v meters, the one with less visit frequent is removed from the candidate list. Above process is repeated until the distance between every two candidate landmarks is larger than D_v meters.

N_v and D_v decide the number of landmarks and consequently, the sizes of sub-areas. The network administrator can adjust their values so that there are landmarks in most areas of the network and in the areas where data is transferred. Based on our experience, this step can be completed quickly based on the understanding of node mobility patterns in an application scenario, as what we did for the traces in the trace analysis part. We leave the analysis on how N_v and D_v affects the routing performance as our future work.

2) Sub-area Division: With the landmarks, we split the entire network into sub-areas. Since the sub-area division only serves the purpose of routing among landmarks, we do not need a method to precisely define the size of each sub-area. Therefore, we follow below rules to generate sub-areas:

- Each sub-area contains only one landmark.
- The area between two landmarks are evenly split to the two sub-areas containing the two landmarks.
- There is no overlap among sub-areas.

Note that the split of area between landmarks does not affect how nodes move between landmarks. Nodes can transit among landmarks through any routes Figure 5 gives an example of the sub-area division in our campus deployment of DTN-FLOW, which is introduced later in Section V-B. With above landmark selection algorithms, a landmark may be responsible for a large area while some landmarks may only be responsible for a small area. Recall that landmarks are selected from popular places. Then, a large sub-area is caused by the fact that there is no other popular place in that area. Then, even we put extra landmarks in the area, packets cannot reach them quickly since they are not popular places and few nodes visit them frequently. Therefore, such a design (i.e., uneven sub-area division) does not degrade the routing efficiency but helps to limit the number of landmarks and save the total cost.

3) Real World Scenarios and Limitations: Above landmark selection and sub-area division procedures require certain administration input. However, as previously introduced, this step is quite intuitive and requires slight effort. With the design of landmarks, we can see that DTN-FLOW is suitable for DTNs with distributed popular places. In a real-world DTN, popular places usually are distributed over an area. For example, the carrier of mobile devices (i.e., human or animal) usually belong to certain social structures and have skewed and repeated visiting patterns [21]. Also, different nodes have

their own frequently visited places. Therefore, the proposed DTN-FLOW is applicable to most realistic DTN scenarios.

B. Node Transit Prediction

Since DTN-FLOW relies on node transit for packet forwarding, accurate prediction of node transit is a key component. In DTN-FLOW, each node uses its previous visiting history for next transit prediction. Each node maintains a landmark visiting history table as shown in Table II. The "Start time" and "End time" denote the time when a node connects and disconnects to the central station in the corresponding landmark, respectively. Note that the "End time" of visiting previous landmark is not necessarily the same with the "Start time" of connecting to current landmark since a node may not always connect to a landmark during its movement.

TABLE II: Landmark visiting history table in a node.

Landmark ID	Start time (s)	End time (s)
8	8500	9000
1	7100	8450
7	2000	7000
$\cdots$	$\cdots$	$\cdots$

To predict node transits among landmarks, we adopt the order-k ($O(k), k = 0, 1, 2, \cdots$) Markov predictor [27], which assumes that the next transit is only related to the past k transits. A node's landmark transit history can be represented by $T_H = T_{x_1,x_2}T_{x_2,x_3}\ldots T_{x_{j-1},x_j}\ldots T_{x_{n-1},x_n}$, in which T_{x_{j-1},x_j} ($x_{j-1} \neq x_j$ and $x_{j-1}, x_j \in [1, M]$) represents a transit from $L_{x_{j-1}}$ to L_{x_j}. We let $X(n-k, n) = T_{x_{n-k},x_{n-k+1}}\cdots T_{x_{n-1},x_n}$ represent the past k consecutive transits. When $k = 0$, $X(n-k, n) = T_{x_n,x_n}$, representing the visiting of landmark L_{x_n}. Then, the probability for each possible next transit $T_{x_n,x_{n+1}}$ of a node is calculated by

$$Pr(T_{x_n,x_{n+1}}|X(n-k,n)) = \frac{Pr(X(n-k,n), T_{x_n,x_{n+1}})}{Pr(X(n-k,n))}, \quad (1)$$

where

$$Pr(X(n-k,n), T_{x_n,x_{n+1}}) = Pr(X(n-k,n+1)) \quad (2)$$

and

$$Pr(X(n-k,n)) = \frac{N(X(n-k,n))}{N(All_k)} \quad (3)$$

where $N(X(\cdot))$ and $N(All_k)$ denote the number of $X(\cdot)$ and k consecutive transits in T_H, respectively. Note $N(All_0)$ denotes the number of landmark visits of the node. Then, the transit that leads to the maximal probability based on Equ. (1) is selected as the predicted transit. For example, suppose we use an order-1 Markov on a system with 5 landmarks ($M = 5$), and the landmark transit history of a node is $T_H = T_{0,1}T_{1,3}T_{3,4}T_{4,2}T_{2,0}T_{0,1}$. Then, based on Equ. (1), the probability for each possible next landmark L_a ($a \in [1, 5]$) is calculated as $\frac{Pr(T_{0,1}T_{1,a})}{Pr(T_{0,1})}$. Suppose $a = 3$. Based on Equ. (3), $Pr(T_{0,1}T_{1,3}) = 1/5$ since $T_{0,1}T_{1,3}$ appears once in T_H and the total number of 2 consecutive transits is 5. Similarly, $Pr(T_{0,1}) = 2/6$. Then, the node's probability of meeting landmark L_3 is 0.67.

We used the order-1 Markov predictor on both DART and DNET traces. Each node makes a prediction for the next visiting landmark after each transit based on its past transits. We calculated the accuracy rate of each node, which equals the number of correct predictions divided by the number of total predictions. The minimal, 1st quantile, average, third quantile and maximal of the accuracy rates of all nodes in the two traces are shown in Figure 6. We see that in the DART trace, the accuracy rates of over 75% of nodes are higher than 64%, and the average accuracy rate of all nodes is about 77%. In the DNET trace, the accuracy rates of over 75% of nodes are higher than 59%, and the average accuracy rate of all nodes is about 66%. It is intriguing to see that the prediction accuracy in the bus network in DNET, which should have more repetitive moving patterns, is lower than that in the student network on campus in DART. We believe this is caused by the reason that we only predict one AP for the next transit while a bus may associate with one of several neighboring APs after each transit in the trace. Though certain inaccurate predictions exist, the routing efficiency can be ensured with a proper handling method that will be explained in Section IV-D. To increase the accuracy of the transit prediction, we can use the product of the prediction probability and the accuracy rate to measure the probability of meeting a specific landmark.

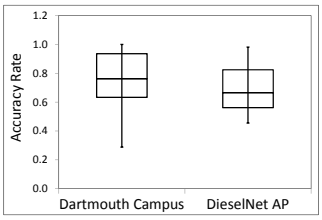


Fig. 6: The minimal, 1st quantile, average, third quantile, and maximal of the transit prediction accuracy.

C. Routing Table Construction

In DTN-FLOW, each landmark first dynamically measures the bandwidths of its transit links to all of its neighbor landmarks. The bandwidth of a transit link represents the expected delay of forwarding data through it. Based on the estimated delay, each landmark uses the distance-vector method [22] to build a routing table indicating the next landmark hop for each destination landmark that can lead to the minimum estimated delay. Each landmark periodically transfers its routing table to its neighbor landmarks, which update their routing tables accordingly. The routing table exchange is realized through mobile nodes based on the prediction of their transits. That is, landmark L_i chooses its node with the highest predicted probability of visiting L_j to forward its routing table to L_j. The detailed processes are introduced below.

1) Transit Link Bandwidth Measurement: Each landmark maintains a bandwidth table as shown in Table III to record the bandwidth from it to each of its neighbor landmarks. We let N_{ij}^t denote the number of nodes that have moved from L_i to L_j in the t-th time unit. Each landmark, L_i, periodically updates its bandwidth to landmark L_j by

$$B_{ij_{new}} = \alpha B_{ij_{old}} + (1-\alpha)N_{ij}^t \quad (4)$$

in which $B_{ij_{new}}$ and $B_{ij_{old}}$ represent the updated bandwidth and previously measured bandwidth, respectively, and α is a

weight factor.

TABLE III: Bandwidth table in a node.

Landmark ID	Measured Bandwidth	Time Unit Sequence
2	20	9
6	6	9
1	15	9
...	...	...

It is easy for landmark L_i to calculate N^t_{ji} since mobile nodes moving to L_i can report their previous landmarks to L_i. However, it is difficult for L_i to calculate N^t_{ij} because after a mobile node moves from L_i to L_j, it cannot communicate with L_i. Recall that **O3** indicates that two matching transit links are symmetric in bandwidths. In this case, L_i can regard $N^t_{ij} \approx N^t_{ji}$ and calculate $B_{ij_{new}}$ using Equ. (4).

However, the symmetric property does not always hold true. For example, transit links connecting two stations in a one way road can hardly be symmetric in bandwidth. In this asymmetric case, each landmark cannot measure the bandwidths of links from it to neighbor landmarks directly. To solve this problem, L_i relies on L_j to keep track of N_{ij}. When landmark L_j predicts that a node is going to leave it for another landmark L_i, it forwards N_{ij} to the node. When L_i receives N_{ij} from L_j, it checks whether the time unit sequence in the packet is larger than the one it uses currently. If yes, it updates its bandwidth to L_j accordingly based on Equ. (4). Otherwise, the packet is discarded.

2) Building Routing Tables: With the bandwidth table, each landmark can deduce the expected delay needed to transfer W bytes of data to each of its neighboring landmarks. Recall T denotes the time unit for B_{ij} measurement. Suppose each node has S bytes of memory, then the expected delay for forwarding a packet from L_i to L_j (D_{ij}) equals $D_{ij} = \frac{W}{B_{ij}S}T$. Then, the routing table on each landmark can be initialized with the delays to all neighboring landmarks. Each landmark, L_i, further uses the distance-vector protocol to construct the full routing table (as shown in Table IV) indicating the next hop for every destination landmark (L_d) in the network and the overall delay from L_i to L_d, denoted by $D(L_i, L_d)$.

TABLE IV: Routing table in one node.

Des. Landmark ID	Next Hop ID	Overall Delay
1	1	7
5	5	3
9	1	18
...	...	...

In the distance-vector protocol, each landmark periodically forwards its routing table and associated time unit to all neighboring landmarks through mobile nodes. When a landmark, say L_i, receives the routing table from a neighboring landmark, say L_j, it first checks whether it is newer than the previous received one. If not, the table is discarded. Otherwise, the routing table is processed one entry by one entry. For each entry, if the destination landmark, L_d, does not exist in the routing table of L_i, it is added to the routing table by setting the "Next Hop ID" as L_j and the "Overall Delay" as $D_{ij} + D(L_j, L_d)$. If L_d already exists, it checks whether $D(L_i, L_d) \leq D_{ij} + D(L_j, L_d)$. If yes, no change is needed. Otherwise, the "Next Hop ID" is replaced as L_j and the "Overall Delay" is updated with $D_{ij} + D(L_j, L_d)$. This process repeats periodically, and each landmark finally learns the next hop to reach each other destination landmark with the minimum overall delay in its routing table.

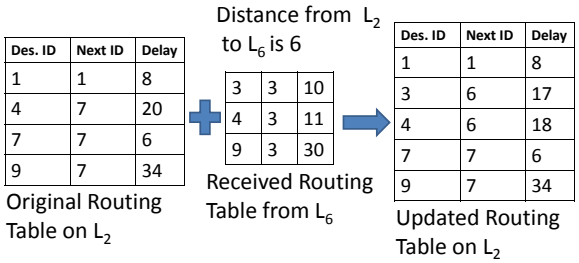


Fig. 7: Demonstration of the routing table update.

For example, suppose the routing table on L_2 contains four entries: (1, 1, 8), (4, 7, 20), (7, 7, 6) and (9, 7, 34). It receives a routing table from L_6 with 3 entries: (3, 3, 10), (9, 3, 30), (4, 3, 11), and D_{26}=7. Figure 7 summaries the process of routing table update. In detail, since the routing table of L_2 has no entry for landmark L_3, it is inserted directly with updated next hop and delay: (3, 6, 17). Though L_9 already exists in the routing table of L_2, D_{29}=34 is less than that of relaying through L_6 (i.e., 37), so no change is needed. L_4 already exists in the routing table, and D_{24}=20 is larger than that of relaying through L_6 (i.e., 18), so the "Next Hop ID" is changed to 6 and "Overall Delay" is set to 18. The final entries in the routing table of L_2 are (1, 1, 8), (3, 6, 17), (4, 6, 18), (7, 7, 6), and (9, 7, 34).

D. Packet Forwarding Algorithm

During the packet forwarding, based on a packet's destination, a landmark refers to its routing table to select the next-hop landmark, and forwards the packet to the mobile node that has the highest predicted probability to transit to the selected landmark. Thus, if node transit prediction is correct, each transit can reduce the expected delay by the maximum. However, as mentioned in Section IV-B, node transit prediction may not always be accurate, which means a node may carry a packet to another landmark rather than the one indicated in the routing table. Also, there may exist nodes that are moving to the packet's destination node directly, which can be utilized to enhance the routing performance. We first introduce our approaches to handle the two issues and then summarize the final routing algorithm.

1) Handling Prediction Inaccuracy: To handle the inaccurate transit prediction, DTN-FLOW follows the principle that every forwarding must reduce the routing latency. Thus, when a node moves from L_i to a landmark L_k other than the predicted one L_j, the node checks whether the new landmark still reduces the expected delay to the destination L_d, that is, whether $D(L_k, L_d) < D(L_i, L_d)$. If yes, the node still forwards the packet to landmark L_k for further forwarding. Otherwise, the node holds the packet, waiting for next landmark that has shorter delay to the destination or a node that is predicted to visit L_j soon. This design aims to

ensure that each transit, though may not be optimal due to node transit prediction inaccuracy, can always improve the probability of successful delivery.

2) Exploiting Direct Delivery Opportunities: Since nodes move opportunistically in a DTN, it is possible that a landmark can discover nodes that are predicted to visit the destination landmarks of some packets. Therefore, when a landmark receives a packet, it first checks whether any connected nodes are predicted to transit to its destination landmark. If yes, the packet is forwarded to the node directly. In case the node fails to forward the packet to its destination landmark, the node uses the scheme described in Section IV-D1 to decide whether to forward the packet to the new landmark.

3) Routing Algorithm: We present the steps of the routing algorithm as following.

(1) When a node generates a packet for an area, it forwards the packet to the first landmark it meets.
(2) When a landmark, say L_i, generates or receives a packet, it first checks whether any nodes are predicted to move to the destination landmark of the packet. If yes, the packet is forwarded to the node with the highest predicted probability and the expected overall delay, which is used by the carrier node to determine whether to forward the packet to an encountered landmark not in prediction.
(3) Otherwise, L_i checks its routing table to find the next-hop landmark for the packet and inserts the landmark ID and the expected overall delay into the packet.
(4) L_i then checks all connected nodes and forwards the packet to the node that has available memory and has the highest predicted probability to transit to the next-hop landmark indicated by the routing table.
(5) When a node moves to the area of a landmark, say L_j, it forwards L_j all packets that target L_j or have less overall delay from L_j to the destination than L_i. After this, it predicts its next transit based on the order-k Markov predictor and informs this to L_j.

V. Performance Evaluation

We first conducted trace-driven experiments with both the DART and the DNET traces and then deployed a small DTN-FLOW system in our campus. We introduce the results of the experiments in the following.

A. Trace-driven Experiments

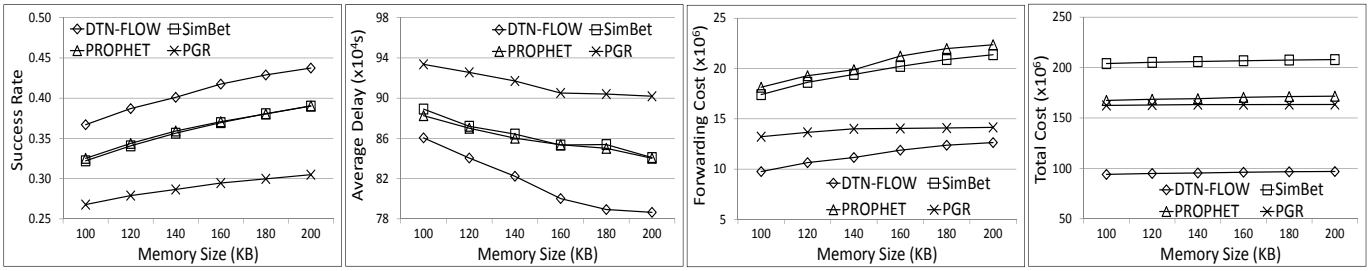
1) Experiment Settings: We used the first 1/4 part of the two traces as the initialization phase, in which nodes construct routing tables. Then, packets were generated at the rate of R_p packets per landmark per day. R_p was set to 40 unless otherwise specified. Each landmark randomly selects another landmark as the destination landmark for its generated packet. We set the TTL (Time to Live) of packets to 20 days in the DART trace and 4 days in the DNET trace. A packet was dropped after TTL. We set a large TTL in order to better evaluate the utilization of node movements. The time unit T for bandwidth evaluation and routing table update was set to 3 days. The size of each packet was set to 1KB, and the size of each node's memory was set to 150KB unless otherwise specified. The memory in the landmark was not limited. We used the order-1 Markov predictor in the experiments. We set the confidence interval to 95%.

We compared DTN-FLOW with three state-of-the-art routing algorithms: SimBet [28], PROPHET [6], and PGR [18]. They were originally proposed for node-to-node routing in DTNs. We adapted them to fit landmark-to-landmark routing to make them comparable to DTN-FLOW. We use SimBet to represent the social network based routing methods. It combines centrality and similarity to calculate the suitability of a node to carry packets to a given destination landmark. Centrality is calculated based on the ability to connect landmarks, and similarity is derived from the past co-existence records about the frequency that the node visits the landmark. We use PROPHET to represent the probabilistic routing methods. It simply employs the encountering records to calculate the future meeting probability to guide the packet forwarding. We modified the two methods to fit our test scenario by referring the suitability and probability to a certain landmark. We use PGR to represent the location based routing methods. It uses observed node mobility routes, described as a sequence of locations, to check whether the destination landmark is on a node's route. It then forwards a packet to the node that is likely to move to the destination landmark along its route. We measured following metrics.

- *Success rate*: The percentage of packets that successfully arrive at their destination landmarks.
- *Average delay*: The average time per packet for successfully delivered packets to reach their destination landmarks.
- *Forwarding cost*: The number of packet forwarding operations occurred during the experiment.
- *Overall cost*: The total number of packet and routing information forwarding operations during the experiment. Forwarding a routing table or a meeting probability table with m entries is counted as m.

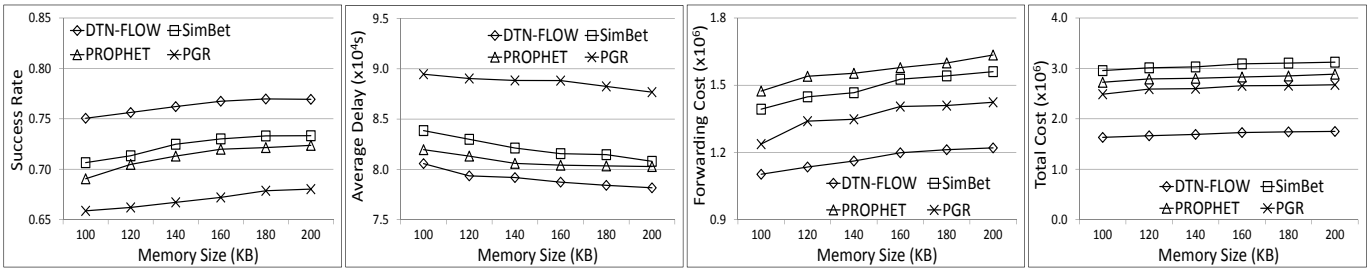
2) Performance with Different Memory Sizes: We first evaluated the performance of the four methods when the size of memory in each node was varied from 100KB to 200KB with a 20KB increase in each step.

Success Rate: Figure 8(a) and Figure 9(a) present the success rates of the four methods with the DART and the DNET traces, respectively. We see that when the memory in each node increases, the hit rates always follow DTN-FLOW$>$SimBet$\approx$PROPHET$>$PGR. DTN-FLOW has the highest hit rate because it fully utilizes node movements to forward packets one landmark by one landmark to their destination landmarks, even though some nodes rarely or may not visit these destinations. On the contrary, other methods only rely on nodes that visit destinations frequently for packet forwarding. Limited number of such nodes prevent them from achieving high success rate. PGR tries to predict the entire route of a node including multiple locations for packet forwarding. However, the accuracy of such a prediction usually is low. Recall in Figure 6, the average accuracy rate is already below 80% for the prediction of only one location. The



(a) Success rate. (b) Average delay. (c) Forwarding cost. (d) Total cost.

Fig. 8: Performance with different memory sizes using the DART trace.



(a) Success rate. (b) Average delay. (c) Forwarding cost. (d) Total cost.

Fig. 9: Performance with different memory sizes using the DNET trace.

prediction of continuous multiple locations would be lower. Therefore, PGR has the lowest success rate.

PROPHET relies on previous encountering frequency to calculate the probability of a node to meet a certain landmark. SimBet exploits social properties, namely centrality and similarity, to rank a node's suitability to carry packets to a certain landmark. Thus, these two methods gradually forward packets to their destination landmarks, leading to a relatively high success rate. We also see that SimBet has slightly higher success rate than PROPHET in one trace and has similar success rate with PROPHET in another trace. This is because in addition to meeting probability, SimBet also considers centrality that measures the activeness of a node to visit different landmarks. Since a node with high centrality may not have high visiting frequency to the destination landmark, the improvement on the success rate is very small.

We also see that when the memory size increases, the success rates of all the four methods increase. This is because each node can carry more packets and provide more forwarding services. In summary, the experimental results verify the high throughput of DTN-FLOW in transferring data among landmarks with difference memory sizes on each node.

Average Delay: Figure 8(b) and Figure 9(b) show the average delays of successfully delivered packets in the four methods with the DART and the DNET traces, respectively. We see that the average delays always follow DTN-FLOW$<$SimBet$\approx$PROPHET$<$PGR when memory size increases. DTN-FLOW has the lowest average delay because the designed routing tables in landmarks guide packets to be forwarded along the fastest paths to their destinations. For PGR, as explained previously, it is difficult to accurately predict long paths with multiple locations, thus leading to inaccurate forwarder selection and the long delay.

In SimBet and PROPHET, packets may be generated in or carried to areas where very few nodes move to their destinations regularly. Therefore, packets have to wait for a certain period of time before meeting nodes that visit their destinations frequently, leading to a moderate average delay. Moreover, since SimBet also considers the centrality. Nodes with high centrality (i.e., connecting many landmarks) cannot meet the destination landmarks as frequently as those that have high meeting probability with the destination landmarks directly. Therefore, it generates a slightly higher average delay than PROPHET.

We also see that when the memory size on each node increases, the average delays of all methods decrease. Larger memory size enables a node to carry more packets from one landmark, reducing the packets' waiting time in landmarks. As a result, each packet can be forwarded more quickly, leading to a lower delay. These experimental results show the high efficiency of DTN-FLOW in transferring data among landmarks with difference sizes of memory in each node.

Forwarding Cost: Figure 8(c) and Figure 9(c) plot the forwarding costs of the four methods with the DART and the DNET traces, respectively. We find that the forwarding costs follow DTN-FLOW$<$PGR$<$SimBet$<$PROPHET with both traces. DTN-FLOW refers to the routing table to forward packets along fastest landmark paths to reach their destinations, which usually takes several forwarding operations.

PGR has the second lowest forwarding cost. This is because based on mobility routes, nodes tend to show similar ability to visit a certain destination. Therefore, a packet holder cannot easily find another node that has higher probability of meeting the destination node. Then, packets are not forwarded frequently. However, the low forwarding cost in PGR also results in a low efficiency.

Both SimBet and PROPHET use a metric to rank the suitability of nodes for carrying packets and forward packets to high rank nodes. Then, even when a packet holder meets a node with a slightly higher suitability, it forwards the packet

to the node. As a result, SimBet and PROPHET have higher forwarding cost than PGR and DTN-FLOW. Also, PROPHET has marginal higher forwarding cost than SimBet. This is because higher-centrality nodes usually are a few, leading to fewer forwarding. On the contrary, PROPHET forwards packets greedily by only considering meeting frequency.

We further find that when the memory size on each node increases, the forwarding costs of the four methods increase. This is because when each node has more memory, it can carry more packets and exchange more packets with encountered landmarks, resulting in more forwarding cost.

Total Cost: Figure 8(d) and Figure 9(d) plot the total costs of the four methods with the DART and the DNET traces, respectively. We see that with both traces, the total costs follow DTN-FLOW$<$PGR$<$PROPHET$<$SimBet. Recall that the total cost includes packet forwarding cost and maintenance cost, which is incurred by routing information forwarding. In DTN-FLOW, the maintenance cost comes from routing table updates. When a node connects to a new landmark, it forwards the routing table of its previously connected landmark to the new landmark and receives the routing table of the new landmark. In PGR, PROPHET, and SimBet, two encountering nodes exchange their calculated suitability/rank for each destination landmark and then decide whether to forward packets to the other node. Since a node's probability of meeting a landmark is lower than that of meeting another node, maintenance cost in DTN-FLOW is less frequent than that in other methods. Therefore, DTN-FLOW produces the lowest maintenance cost, and hence the lowest total cost.

Moreover, SimBet has higher maintenance cost than PGR and PROPHET since in addition to the similarity information, nodes in SimBet also need to exchange one more piece of centrality information. Comparing Figure 8(d) and Figure 9(d) with Figure 8(c) and Figure 9(c), we notice the maintenance cost is much higher than the forwarding cost. Therefore, SimBet has the highest total cost. PGR and PROPHET have roughly the same maintenance cost since two encountering nodes in PGR and PROPHET exchange almost the same amount of information.

We also see that when the memory size on each node increases, the total costs of all methods remain stable. This is because the maintenance cost is much higher than the forwarding cost and is only determined by the number of encounters among nodes or between nodes and landmarks, which is irrelevant to the memory size in each node. The results on forwarding cost and total cost verify the high efficiency of DTN-FLOW in terms of cost with different memory sizes on each node.

3) Performance with Different Packet Rates: We also evaluated the performance of the four methods with different packet generation rates. We varied the packet rate from 20 to 60 with 10 increase in each step.

Success Rate: Figure 10(a) and Figure 11(a) show the success rates of the four methods using the DART and the DNET traces, respectively. We see that the success rates follow DTN-FLOW$>$SimBet$\approx$PROPHET$>$PGR. Such results match those in Figure 8(a) and Figure 9(a) for the same reasons.

We also see that when the packet rate increases, the success rates of the four methods decrease. The forwarding opportunities in the system are determined by node memory and encountering opportunities, which are independent with the number of packets. When the number of packets increases, the number of packets that can be delivered successfully does not increase accordingly, leading to a degraded success rate. The high success rate of DTN-FLOW with different packet rates verifies the high throughput performance of DTN-FLOW.

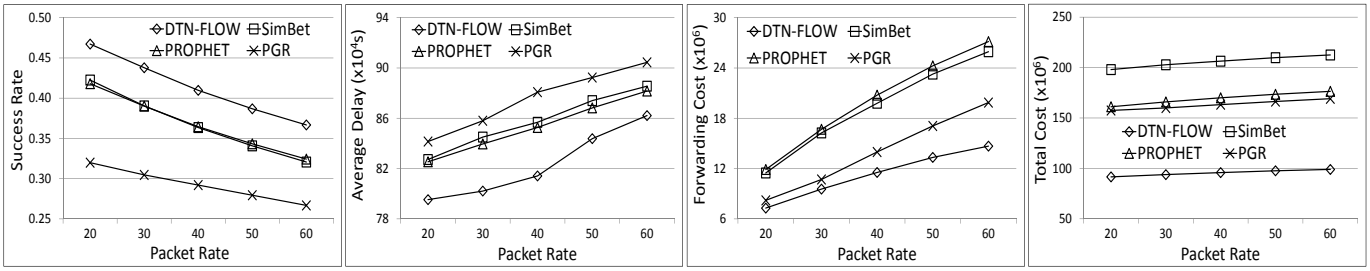
Average Delay: Figure 10(b) and Figure 11(b) illustrate the average delays of the four methods using the DART and the DNET traces, respectively. We see that the average delays follow DTN-FLOW$<$SimBet$\approx$PROPHET$<$PGR. This relationship remains the same as in Figure 8(b) and Figure 9(b) for the same reasons. Moreover, we find that when the packet rate increases, the average delays of the four methods increase. This is caused by the limited forwarding opportunities in the system. When there are more packets in the system, the average time a packet needs to wait before being forwarded increases, resulting in higher total delay. DTN-FLOW always generates the lowest average delay at all packets rates, which demonstrates the high efficiency of DTN-FLOW in terms of routing delay.

Forwarding Cost: Figure 10(c) and Figure 11(c) show the forwarding costs of the four methods using the DART and the DNET traces, respectively. We see that the forwarding costs follow DTN-FLOW$<$PGR$<$SimBet$<$PROPHET. Again, this relationship is the same as in Figure 8(c) and Figure 9(c) due to the same reasons. We also see that the forwarding costs of the four methods increase when the packet rate increases. When there are more packets generated in the system, more forwarding opportunities are utilized, resulting in more packets forwarding operations. When the packet rate is large enough and all forwarding opportunities are utilized sufficiently, the forwarding cost would remain stable. This is why the forwarding costs in Figure 11(c) remain relative stable when the packet rate is larger than 40.

Total Cost: Figure 10(d) and Figure 11(d) show the total costs of the four methods using the DART and the DNET traces, respectively. We see that their total costs again follow DTN-FLOW$<$PGR$<$PROPHET$<$SimBet. This result matches that in Figure 8(d) and Figure 9(d) for the same reasons. We also find that the total costs of the four methods remain quite stable when the packet rates increase. This is because that the maintenance costs of the four methods, which are irrelevant to the packet rate, dominate the total costs. Such results further confirm the high efficiency of DTN-FLOW in terms of cost with difference packet rates.

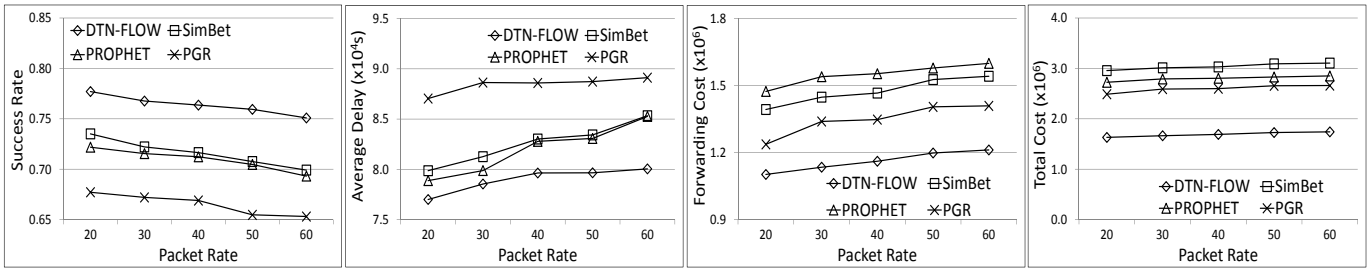
Combining all above results obtained with various memory sizes and packet rates, we conclude that DTN-FLOW has superior performance in achieving high throughput, low average delay, and low cost data transmission between landmarks than previous routing algorithms in DTNs.

B. Real Deployment



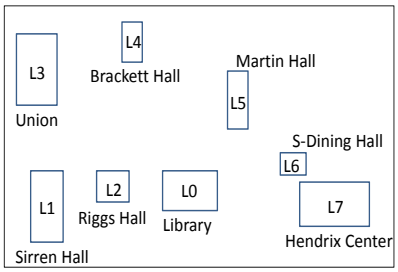
(a) Success rate. (b) Average delay. (c) Forwarding cost. (d) Total cost.

Fig. 10: Performance with different packet rates using the DART trace.



(a) Success rate. (b) Average delay. (c) Forwarding cost. (d) Total cost.

Fig. 11: Performance with different packet rates using the DNET trace.

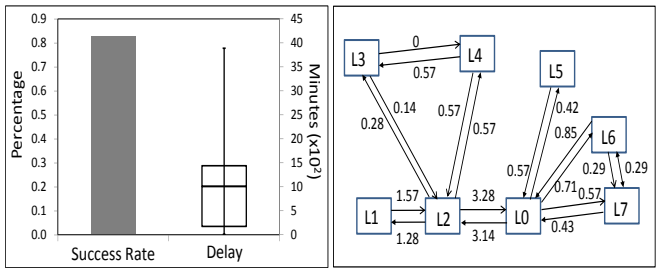


(a) Map for landmark locations.

Item	Value
# Landmarks	8
# Nodes	9
# Transits	147
# Packets	2100
Duration	4 days
Packet size	1 KB
Node memory	50 KB

(b) Configuration summary

Fig. 12: Landmark map and configurations in the real deployment.



(a) Success rate and delay distribution. (b) Bandwidths of transit links

Fig. 13: Experimental results in real deployment.

1) Settings: We deployed DTN-FLOW in our campus for real-world evaluation of its performance. We selected 8 buildings as landmarks and labeled them as L_0 to L_7. Their location relationships are shown in Figure 12(a). Among the 8 landmarks, L_0 is the library, L_1, L_2, L_4, and L_5 are department buildings, and L_3, L_6, and L_7 are the student center and dining halls. Each of 9 students from 4 departments carried a Windows Mobile phone daily. Each Windows Mobile phone checks its GPS coordinator periodically to judge whether it is within the area of any landmarks. If yes, it communicates with the landmark through Wifi.

In the test, each landmark generates 75 packets evenly in the daytime of each day. We simulated a scenario in which L_0 (Library) needs to collect information from other buildings. Then, all packets were targeted to L_0. The TTL of each packet was set to 3 days. We set the size of each packet to 1KB and the memory on each node to 50KB. The time unit T was set to 720 minutes. The deployment configuration is summarized in Figure 12(b).

2) Experimental Results: Figure 13(a) demonstrates the success rate and the minimal, first quantile, average, third quantile, and maximal of the delays of successfully delivered packets. We see that more than 82% of packets were successfully delivered to the destination. Also, more than 75% of packets were delivered within 1400 minutes, and the average delay is about 1000 minutes. Note that the entire deployment only employed 9 mobile nodes with 147 transits to forward packets. A larger deployment with more nodes would increase the success rate and reduce the delay. These experimental results demonstrate the high efficiency of the DTN-FLOW in transferring data among landmarks.

We also obtained the bandwidth of each transit link at the end of the deployment, as shown in Figure 13(b). We omit transit links with bandwidth lower than 0.14 to show the major routing paths. We again find that for most pairs of landmarks, the two transit links connecting them are symmetric on bandwidth. This further confirms our previous observation (**O3**) in Figure 3(a) and Figure 3(b). The bandwidth on different transit links are also within our expectation. For example, the links between L_0 and L_2 have very high bandwidth. This is because most students attended the test are from departments located in L_2 and L_0, and they usually study in the library (L_0) and go to classes in both department buildings (L_1 and L_2). Also, the route between L_1 and L_0 must go through L_2. Such results justify that the DTN-FLOW can accurately measure the

amount of transits among landmarks.

We further recorded the routing table on each landmark. Due to page limit, we only show those of L_2, L_4, and L_6 in Table V. We see that the routing tables match the fastest path based on transit link bandwidths shown in Figure 13(b). For L_2, it needs to go through L_0 to reach L_0, L_5, L_6, L_7. For L_4, it relies on L_3 and L_2 to reach other landmarks. For L_6, except for L_7, it has to go through L_0 to reach other landmarks. Such results verify that the routing table update in DTN-FLOW, which relies on mobile nodes, is reliable and can reflect the suitable paths to each destination.

TABLE V: Routing tables in L_2, L_4, and L_6.

Landmark ID	Destination Landmark	Next-hop
L_2	L_0, L_5, L_6, L_7	L_0
	L_1	L_1
	L_3	L_3
	L_4	L_4
L_4	L_0, L_1, L_3, L_6, L_7	L_3
	L_2, L_5	L_2
L_6	$L_0, L_1, L_2, L_3, L_4, L_5, L_6$	L_0
	L_7	L_7

VI. Conclusion

In this paper, we propose DTN-FLOW, an efficient routing algorithm to transfer data among landmarks for high throughput in DTNs. DTN-FLOW splits the entire DTN area into sub-areas with different landmarks, and uses node transits between landmarks to forward packets one landmark by one landmark to reach their destinations. Specifically, DTN-FLOW consists of four components: landmark selection and sub-area division, node transit prediction, routing table construction, and packet routing algorithm. The first component selects landmarks from places that are frequently visited by nodes and split the network work into sub-areas. The second component predicts node transits among landmarks based previous movements using the order-k Markov predictor. The third component measures the transmission capability between each pair of landmarks, which is used to build routing tables indicating the next landmark hop to each destination. The routing table exchange among landmarks for routing table update is realized through mobile nodes. In the fourth component, each landmark decides the next-hop landmark for each packet by checking its routing table and forwards the packet to the node with the highest probability of meeting the landmark. We have analyzed two real traces to verify the shortcoming of previous DTN routing algorithms and hence confirm the motivation and design of DTN-FLOW. We also deployed a small DTN-FLOW system in our campus. The experimental results from the real deployment and extensive trace-drive simulation prove the high efficiency and throughput of DTN-FLOW compared with different previous routing algorithms in DTNs. In the future, we plan to investigate how to combine node-to-node communication to further enhance the packet routing efficiency.

Acknowledgements

This research was supported in part by U.S. NSF grants CNS-1254006, CNS-1249603, CNS-1049947, CNS-1156875, CNS-0917056 and CNS-1057530, CNS-1025652, CNS-0938189, CSR-2008826, CSR-2008827, Microsoft Research Faculty Fellowship 8300751, and U.S. Department of Energy's Oak Ridge National Laboratory including the Extreme Scale Systems Center located at ORNL and DoD 4000111689.

References

[1] S. Jain, K. R. Fall, and R. K. Patra, "Routing in a delay tolerant network," in *Proc. of SIGCOMM*, 2004.
[2] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. of SIGCOMM*, 2003.
[3] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with ZebraNet," in *Proc. of ASPLOS-X*, 2002.
[4] A. Vahdat and D. Becker, "Epidemic routing for partially-connected ad hoc networks," Duke University, Tech. Rep., 2000.
[5] Y. Tseng, S. Ni, and E. Shih, "Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network." in *Proc. of ICDCS*, 2001.
[6] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks." *Mobile Computing and Communications Review*, vol. 7, no. 3, 2003.
[7] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "MaxProp: Routing for vehicle-based disruption-tolerant networks," in *Proc. of INFOCOM*, 2006.
[8] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "DTN routing as a resource allocation problem." in *Proc. of SIGCOMM*, 2007.
[9] K. Lee, Y. Yi, J. Jeong, H. Won, I. Rhee, and S. Chong, "Max-Contribution: On optimal resource allocation in delay tolerant networks." in *Proc. of INFOCOM*, 2010.
[10] F. Li and J. Wu, "MOPS: Providing content-based service in disruption-tolerant networks," in *Proc. of ICDCS*, 2009.
[11] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: social-based forwarding in delay tolerant networks," in *Proc. of MobiHoc*, 2008.
[12] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant manets," in *Proc. of MobiHoc*, 2007.
[13] E. Yoneki, P. Hui, S. Chan, and J. Crowcroft, "A socio-aware overlay for publish/subscribe communication in delay tolerant networks," in *Proc. of MSWiM*, 2007, pp. 225–234.
[14] P. Costa, C. Mascolo, M. Musolesi, and G. P. Picco, "Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks," *IEEE JSAC*, vol. 26, no. 5, pp. 748–760, 2008.
[15] B. Chiara and *et al.*, "Hibop: A history based routing protocol for opportunistic networks," in *Proc. of WoWMoM*, 2007.
[16] J. Link, D. Schmitz, and K. Wehrle, "GeoDTN: Geographic routing in disruption tolerant networks," in *Proc. of GLOBECOM*, 2011.
[17] A. Sidera and S. Toumpis, "DTFR: A geographic routing protocol for wireless delay tolerant networks," in *Proc. of Med-Hoc-Net*, 2011.
[18] J. Kurhinen and J. Janatuinen, "Geographical routing for delay tolerant encounter networks," in *Proc. of ISCC*, 2007.
[19] I. Leontiadis and C. Mascolo, "GeOpps: Geographical opportunistic routing for vehicular networks," in *Proc. of WOWMOM*, 2007.
[20] J. Lebrun, C. nee Chuah, D. Ghosal, and M. Zhang, "Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks," in *Proc. of VTC*, 2005.
[21] J. Leguay, T. Friedman, and V. Conan, "DTN routing in a mobility pattern space," in *Proc. of WDTN*, 2005.
[22] C. Hedrick, "RFC1058 - Routing Information Protocol," 1988.
[23] K. C. Lee, M. Le, J. H0Ł1rri, and M. Gerla, "Louvre: Landmark overlays for urban vehicular routing environments." in *Proc. of VTC Fall*, 2008.
[24] A. G. Valko, "Cellular IP: A new approach to internet host mobility," *ACM Computer Comminication*, 1999.
[25] T. Henderson, D. Kotz, and I. Abyzov, "The changing usage of a mature campus-wide wireless network," in *Proc. of MOBICOM*, 2004.
[26] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "Enhancing interactive web applications in hybrid networks," in *Proc. of MOBICOM*, 2008.
[27] L. Song, D. Kotz, R. Jain, and X. He, "Evaluating location predictors with extensive wi-fi mobility data," in *Proc. of INFOCOM*, 2004.
[28] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant MANETs," in *MobiHoc*, 2007.