Evaluating MobySpace Based Routing Strategies in Delay Tolerant Networks

Jérémie Leguay,*† Timur Friedman,* Vania Conan†
* Université Pierre et Marie Curie, Laboratoire LiP6–CNRS
† Thales Communications

Abstract

Because a delay tolerant network (DTN) can often be partitioned, routing is a challenge. However, routing benefits considerably if one can take advantage of knowledge concerning node mobility. This paper addresses this problem with a generic algorithm based on the use of a high-dimensional Euclidean space, that we call MobySpace, constructed upon nodes' mobility patterns. We provide here an analysis and a large scale evaluation of routing schemes using MobySpace by replaying real mobility traces. The specific MobySpace evaluated is based on the frequency of visits of nodes to each possible location. This evaluation was first presented in the proceedings of IEEE INFOCOM 2006, but for single-copy protocols only. Here, we also evaluate multi-copy routing strategies that use MobySpace as a means to control flooding. We show that routing based on MobySpace can achieve good performance compared to a number of common algorithms.

I. INTRODUCTION

This paper addresses the problem of routing in delay tolerant networks (DTNs) [1]. It evaluates schemes which are based on an earlier proposition [2] that turns the problem of DTN routing into a problem of routing in a virtual space defined by the mobility patterns of nodes. We called this virtual space MobySpace. By conducting simulations with real mobility traces, we have evaluated this concept using a single-copy routing scheme [3]. The additional contribution of this paper is the evaluation of multi-copy routing strategies that use MobySpace as a means to control flooding.

In one common DTN scenario, like the one we consider in this paper, nodes are mobile and have wireless networking capabilities. They are able to communicate with each other only when they are within transmission range. The network suffers from frequent connectivity disruptions, making the topology intermittently and partially connected. This means that there is a very low probability that an end-to-end path exists between a given pair of nodes at a given time. End-to-end paths can exist temporarily, or may sometimes never exist, with only partial paths emerging. Due to these disruptions, regular ad-hoc networking approaches to routing and transport do not work, and new solutions must be proposed.

The Delay Tolerant Network Research Group (DTNRG) [4] has proposed an architecture [5] to support messaging that may be used by delay tolerant applications in such a context. The architecture consists mainly of the addition of an overlay, called the bundle layer, above a network's transport layer. Messages transferred in DTNs are called bundles. They are transferred in an atomic fashion between nodes using a transport protocol that ensures node-to-node reliability. These messages can be of any size. Nodes are assumed to have buffers in which they can store the bundles.

The problem of routing in DTNs is not trivial. Epidemic routing [6], studied by Vahdat and Becker, is a possible solution when nothing is known about the behavior of nodes. Since it leads to buffer overloads and inefficient use of transmission media, one would prefer to limit bundle duplication and instead use routing heuristics that can take advantage of the context. The main contribution of this paper is the validation of routing schemes for DTNs that use the formalism of a high-dimensional Euclidean space based on nodes' mobility patterns, called MobySpace. We first present the evaluation of a single-copy routing scheme which uses this concept. Then, we extend this by investigating routing strategies, also based on MobySpace, which perform controlled flooding to achieve better performance in terms of delivery and delay while not impacting the network too much.

We show the feasibility of the MobySpace-based routing strategies through an instance of that virtual space in which each dimension represents the frequency with which a node can be found at a particular location. We conduct simulations by replaying mobility traces to analyse the feasibility and comparative performance of such schemes.

II. MobySpace: a Mobility Pattern Space

Two people having similar mobility patterns are more likely to meet each other, thus to be able to communicate. Based on this simple principle, our proposition [2] is to use the formalism of a Euclidean virtual space, that we call a *MobySpace*, as a tool to help nodes make routing decisions. These decisions rely on the notion that a node is a good candidate for taking custody of a bundle if it has a mobility pattern similar to that of the bundle's destination. Routing is done by forwarding bundles toward nodes that have mobility patterns that are more and more similar to the mobility pattern of the destination. Since in the MobySpace, the mobility pattern of a node provides its coordinates, called its *MobyPoint*, routing is done by forwarding bundles toward nodes that have their MobyPoint closer and closer to the MobyPoint of the destination. Note that the MobySpace is purely a virtual expression of the mobility patterns, and as such does not express the geographic coordinates of the nodes (GPS or otherwise). It cannot be used for geographic routing.

In this section, we describe manners in which mobility patterns can be characterized and the ways these patterns can be managed by the nodes, and we discuss possible limits and issues surrounding the overall concept.

A. Mobility pattern characterization

The way in which mobility patterns are characterized determines the number and the type of the dimensions of the specific MobySpace. It bears repeating that the MobySpace is not a physical space: each MobyPoint summarizes some characteristics of a node's mobility pattern. Many methods could be employed to describe a mobility pattern, but some requirements must be satisfied. We want mobility patterns to be simple to measure in order to keep them computationally inexpensive and to reduce the overhead associated with exchanging them between nodes. Furthermore, they must be relevant to routing, by helping nodes to take efficient routing decisions.

A mobility pattern could be based, for instance, upon historic information regarding contacts that the node has already had. A recent study [7] by Hui et al. has shown the interest of such mobility patterns. It highlights that contacts between people at the Infocom 2005 conference follow power-laws in terms of their duration. If we want to route a bundle from one node to another, we have an interest in taking the unevenness of the distribution into consideration. Intuitively, it could be very efficient to transmit a bundle to a relay that frequently encounters the destination. A MobySpace based on this kind of pattern would be as follows. Each possible contact is an axis, and the distance along that axis indicates an estimate of the probability of contact. Two nodes that have a similar set of contacts that they see with similar frequencies are close in this space, whereas nodes that have very different sets of contacts, or that see the same contacts but with very different frequencies, are far from each other. It seems reasonable that one would wish to pass a bundle to a node that is as close as possible to the destination in this space, because this should improve the probability that it will eventually reach the destination.

We might wish to consider an alternative space in which there is a more limited number of axes. If nodes' visits to particular locations can be tracked, then the mobility pattern of a node can be described by its visits to these locations. In this scenario, each axis represents a location, and the distance along the axis represents an estimate of the probability of finding a node at that location. We can imagine that nodes that have similar probabilities of visiting a similar set of locations are more likely to encounter each other than nodes that are very different in these respects.

The evaluation and the comparison of the different kinds of mobility patterns are kept for further studies. In Secs. III and IV, we test a MobySpace based on the frequency with which nodes find themselves in certain locations

B. Mobility pattern acquisition

There are several ways a node can learn its own mobility pattern. First, a node can learn its mobility pattern by observing its environment, e.g., by studying its contacts or its frequency of visits to different locations. If the node requires information about its current position, we can assume that particular tags are attached to each location. Alternatively, we can imagine that nodes are able to interrogate an exiting infrastructure to obtain these patterns. This infrastructure would act as a passive monitoring tool for pattern calculation. The system can be accessible anywhere in a wireless or in a wired fashion or it can be located at certain places.

Similarly, there are several ways that a node can learn the mobility patterns of other nodes. These mobility patterns could be spread in an epidemic fashion. Nodes could also spread just the most significant coordinates of their mobility patterns to reduce buffer occupancy and network resource consumption (an idea that we explored in [3]). We can also imagine that nodes drop off their mobility patterns in repositories placed at strategic locations, and at the same time they update their knowledge with the content available at the repositories. We leave the study of possible solutions to future work.

C. Mobility pattern usage

As mentioned in the introduction, the mobility pattern of a node determines its coordinates in the MobySpace, i.e., the position of its associated MobyPoint. The basic idea is that bundles are forwarded to nodes having mobility patterns more and more similar to that of the destination. Formally, let U be the set of all nodes and L be the set of all locations. The MobyPoint for a node $k \in U$ is a point in an n-dimensional space, where $n = |L|$. We write $m_k = (c_{1_k}, ..., c_{n_k})$ for the MobyPoint of node k. The distance between two MobyPoints is written $d(m_i,m_j)$.

At a point in time, t, the node k will have a set of directly connected neighbors, which we write as $W_k(t) \subseteq U$. $W_k^+(t) = W_k(t) \cup \{k\}$ is the augmented neighborhood that contains k. MobySpace routing consists of either choosing one of these neighbors to receive the bundle or deciding to keep the bundle. The routing function, which we call f, chooses the neighbor that is closest to the destination, b. The decision for node k when sending a bundle to b is taken by applying the function f:

$$f(W_k^+(t),b) \quad = \quad \begin{cases} b \text{ if } b \subset W_k(t), \text{ else} \\ i \in W_k^+(t) : d(m_i,m_b) = \min_{j \in W_k^+(t)} d(m_j,m_b) \end{cases} \tag{1}$$

The choice of the distance function d used in the routing decision process is important. One straightforward choice is Euclidean distance. Examples of other distance functions can be found in [2].

D. Possible limits and issues

DTN routing in a contact space or a mobility space is based on the assumption that there will be regularities in the contacts that nodes have, or in their choices of locations to visit. There is always the possibility that we may encounter mobility patterns similar to the ones observed with random mobility models. The efficiency of the virtual space as a tool may be limited if nodes change their habits too rapidly.

The Euclidean spaces that we have discussed here are finite in terms of number of dimensions, but in practice the number of dimensions might be unbounded. This is the case, for instance, in the space we use as a case study in Sec. III. Additional mechanisms must be found to allow this.

Finally, the routing scheme presented here is based on each node forwarding just a single copy of a bundle, which may be a problem in case of node failure or nodes leaving the system for extended periods of time. We investigate in Sec. V strategies that use MobySpace to perform multi-copy routing efficiently.

III. Frequency of visit based MobySpace

To evaluate routing based on MobySpace, we use a simple kind of space that we describe in the first part of this section. The second part introduces the mobility data that we replay for the evaluation.

A. Description

The frequency of visit based MobySpace we evaluate works as follow. Over a defined time interval, each node spends some portion (possibly zero) of that time at each of the n locations. This set of quantities is a node's mobility pattern, and is described by a MobyPoint in an n dimensional MobySpace. If we consider the frequencies to be reliable estimates of future probabilities, the coordinate of a node along the axis k is its probability of visit for the location k. All MobyPoints in a given MobySpace lie in a hyperplane, since we have:

$$\text{for any point } m_i = (c_{1_i}, ..., c_{n_i}),\ \sum_{k=1}^{n} c_{k_i} = 1 \tag{2}$$

Recent studies of the mobility of students in a campus [8], [9] or of corporate users [10], equipped with PDAs or laptops able to be connected to wireless access networks, show that they follow common mobility patterns. They show that significant aspects of the behavior can be characterized by power law distributions. Specifically, the session durations and the frequencies of the places visited by users follow power laws. This means that users typically visit a few access points frequently while visiting the others rarely, and that users may stay at few locations for long periods while visiting the others for very short periods. Henderson et al. observed [9] that 50% of users studied spent 62% of their time attached to a single access point, and this proportion decreased exponentially.

B. Real mobility data used

For the purpose of this study, we sought large real mobility traces that resembled what one might find in an ambient network environment. Since there are very few traces of this kind, we chose data that tracks mobile users in a campus setting. We used the mobility data collected on the Wi-Fi campus network of Dartmouth College [9]. The Dartmouth data is the most extensive data collection available that covers a large wireless access network. The network is composed of about 550 access points (APs), the number of different wireless cards (MAC addresses) seen by the network is about $13{,}000$ and the data have been collected between the years 2001 and 2004. The network covers the college's academic buildings, the library, the sport infrastructures, the administrative buildings and the student residences. Users are equipped with devices such as PDAs, laptops, and phones that support voice over IP (VoIP). The majority of the end users are students, who make intensive use of the network, especially since many of them are required to own a laptop.

The data we analysed track users' sessions in the wireless network. These data have been pre-processed by Song et al. in their prior work [11] on mobility prediction. The traces show the time at which a node associates or dissociates from an access point. Data were collected by a central server with the Syslog [12] protocol. It could happen that a node does not send a dissociation message, or that a Syslog UDP message is lost, in which case a session is considered finished after 30 minutes of inactivity.

For our study, each access point represents a location. We assume that two nodes (represented as networking cards in the data) are assumed to be able to communicate with a low range device (using Bluetooth for instance), if they are attached at the same time to the same AP. This assumption is somewhat artificial as nodes that are attached to two different APs that are close to each other might be able to communicate directly. Similarly, two nodes connected to the same AP might be out of range of each other. Nonetheless, this is the best approximation we can make with the data at hand.

The Wi-Fi scenario may be not a perfect fit for interactions between nodes in DTNs. Indeed, in opposition to always-on devices carried by humans, Wi-Fi nodes are typically turned off, transported,

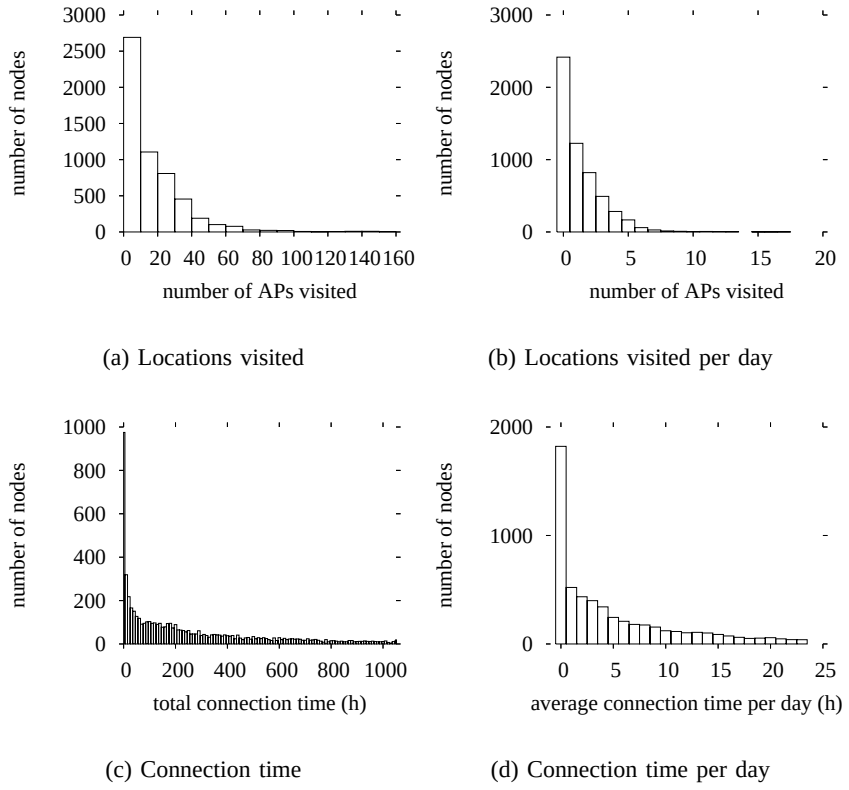
and then turned on again, thus missing potential contacts en route. However, the size, quality, and public availability of the data set make it nonetheless one of the best resources for this kind of study. Jones et al. [13] and Chaintreau et al. [14] recently used these traces in a similar way.

IV. Simulation results

This section presents the manner in which we evaluated the single-copy routing scheme presented in Sec. II-C that uses a frequency of visit based MobySpace, and the results we obtained. We first describe the properties of the 45 days of mobility data we used.

A. Mobility traces

We replayed the mobility traces inferred from Dartmouth data between January 26th 2004 and March 11th 2004. Fig. 1 shows distributions that characterize users' behavior within this period.



(a) Locations visited

(b) Locations visited per day

(c) Connection time

(d) Connection time per day

Fig. 1. Statistics on the data set with all users.

Users are mobile. They visit on average 16.66 locations in the period (see Fig. 1(a)) and 1.75 locations per day (see Fig. 1(b)). The distributions of the number of locations visited by the nodes during the period and per day follow heavy tailed distributions. This means that the majority of users have a low level of mobility while some users are very mobile. Users with a low mobility level regarding the number of locations they visit may either be users that are not very present in the data or users that stay in one place, as in students who keep their laptop connected in their room at the student residence.

Users make intensive use of the network. The mean presence time for the period is 243 hours and is 5.18 hours per day (see Fig. 1(c) and Fig. 1(d)).

B. Methodology

We have implemented a stand alone simulator to perform DTN routing evaluations. This simulator only implements the transport and network layers and it makes simple assumptions regarding lower layers, allowing infinite bandwidth between nodes and contention free access to the medium. Nodes are also supposed to have infinite buffers and to have inherent knowledge of all other nodes' mobility patterns. Because in ambient networks, nodes may have limited resources and capabilities, routing solutions should also be evaluated with limited buffers and more realistic models for the MAC and physical layers. Since our aim here is mainly to validate the idea of MobySpace routing, we leave to future work a detailed study of the modifications that would be required to accommodate resource limitations.

We compare the performance of MobySpace routing against the following:

- *Epidemic routing*: This is described by Vahdat and Becker [6]: Each time two nodes meet, they exchange their bundles. The major interest of this algorithm is that it provides the optimum path and thus the minimum bundle delay. We use it here as a lower bound. This algorithm can be also seen as the extension of Dijkstra's shortest path algorithm proposed by Jain et al. [15] that takes into account time-varying edge weights. In practice, epidemic routing suffers from high buffer occupancy and high bandwidth utilization.
- *Opportunistic routing*: A node waits to meet the destination in order to transfer its bundle. The main advantage of this method is that it involves only one transmission per bundle. Bundle delivery relies just on the mobility of nodes and their contact opportunities.
- *Random routing*: There are many ways to define a random routing algorithm. In order to design one that acts similarly to the MobySpace-based routing scheme, we attribute for each destination node j a preference list l_j, which is a randomly ordered list of all of the nodes. When a node has a bundle destined to j, it sends that bundle to the most preferred neighbor on the preference list l_j. If the most preferred neighbor has a lower preference than the current node, the bundle is not forwarded. This mechanism avoids loops by construction.
- *Hot potato routing*: When a node is at a location and the bundle's destination in not there, the node transfers the bundle to a neighbor chosen at random. We have added a rule to avoid local loops: a node can only handle a bundle one time per location visit.

We will refer to these schemes by the following names: *Epidemic*, using Epidemic routing; *Opportunistic*, using Opportunistic routing; *Random*, using Random routing; *Potato*, using Hot potato routing, and *MobySpace*, using the routing scheme that relies on the MobySpace.

We considered the whole set of 536 locations that were visited over the course of the 45 days of data. The virtual space used for routing thus has 536 dimensions. Due to the difficulty of running simulations with the totality of the $5{,}545$ nodes, especially with Epidemic, for which computation explodes with the number of nodes and the number of bundles generated, we used a sampling method. We have defined two kinds of users: *active*, which generate traffic, and *inactive*, which only participate in the routing effort. Every active node establishes a connection towards 5 other nodes. An active node sends one bundle per connection. For active users, we chose only the ones that appear at least one time in the first week of the simulations in order to be able to study bundle propagation over an extended period. In each run, we sampled 300 users with 100 of them generating traffic. The simulator used a time step of 1s.

Note that since we do not have the knowledge of the social relationship that link users in the data set we used, we generate source destination pairs randomly. However, in a real scenario, we envision that it would be drastically different. People might exchange data with those that are close to them in a sociologically way which simplify the transportation of data. As a consequence, the performance obtained later in this paper may be significantly lower compared to the one we might have in a real case.

We performed 5 runs for each scenario. Simulation results reported in the following tables are mean results with confidence intervals at the 90% confidence level, obtained using the Student t distribution.

C. Results

We evaluate the routing algorithms with respect to their transport layer performance. We consider a good algorithm to be one that yields a low average bundle delay, the highest bundle delivery ratio and a low average route length. We consider two different kinds of scenarios. One with only randomly sampled users and one with only the most active.

With randomly sampled users

In this scenario, we picked 300 users completely at random and we replayed their traces while simulating DTN routing. Table I shows the simulation results. It shows for each of the implemented algorithms the mean bundle delay in number of days, the mean delivery ratio, which corresponds to the number of bundles received over the number of bundles sent, and the mean route length in number of hops. The average delay and the mean route length are computed only over bundles that were delivered.

	delivery ratio (%)	**delay** (days)	**route length** (hops)
Epidemic	82.0 ± 2.7	12.5 ± 0.9	7.10 ± 0.2
Opportunistic	4.9 ± 0.6	15.9 ± 2.5	1.0 ± 0.0
Random	7.2 ± 0.5	16.6 ± 2.6	3.12 ± 0.2
Potato	10.7 ± 1.7	19.1 ± 1.6	72.7 ± 16.5
MobySpace	14.9 ± 2.9	18.9 ± 1.0	3.8 ± 0.2

TABLE I

RESULTS WITH RANDOMLY SAMPLED USERS.

The first thing we can observe is the fact that within the 45 days of simulation there is still a certain number of bundles that are not delivered with Epidemic. The mobility of the 300 nodes or their level of presence were not sufficient to ensure all the deliveries. Our sample included just 5% of the entire set of nodes. By deploying this system on more nodes, the delivery ratio would rise closer to 100%. Furthermore, we did not select nodes based on their mobility characteristics. Some of the nodes may have poor mobility.

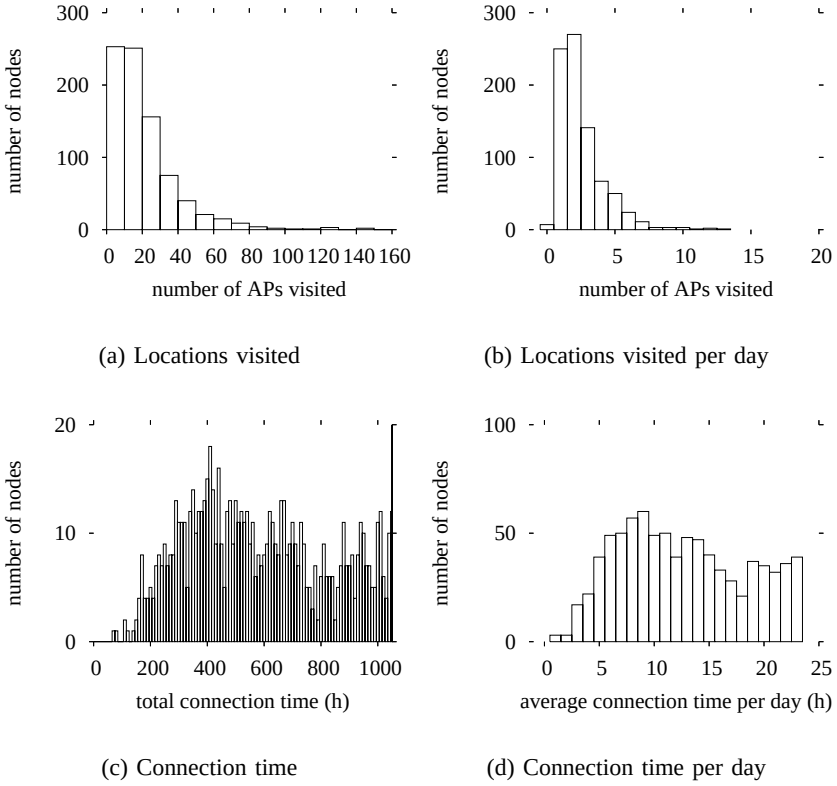
Table I shows that MobySpace delivers twice as many bundles as Random but still far less than Epidemic, which does not miss any opportunities. Random delivers somewhat more bundles than Opportunistic because the bundles are more mobile. This phenomenon is even true for Potato, which outperforms Random but delivers fewer bundles than MobySpace. At first glance, the average bundle delay of MobySpace seems poor. We believe this average is influenced by the fact that more bundles are delivered compared to the other schemes, except Epidemic. The additional bundles delivered by MobySpace might be more difficult to route than the others, leading to higher delays. The investigation of this issue is kept for future work. However, the average bundle delay is an interesting indicator of the performance an algorithm can achieve. Looking now at the average route lengths, we see that in all the cases, except Potato, they are lower than for Epidemic. MobySpace engenders routes that are about half as long as those created by Epidemic. With MobySpace, bundles are transmitted from a node to another because of their mobility patterns, not simply because of the opportunities of contact. Potato engenders routes that are extremely long because, at each contact, bundles switch from one node to another. Potato may not be suitable for a real system because of bandwidth and energy consumption issues.

With the most active users

We also evaluate routing in a scenario with only the most active users, to see the effect of activity on performance. Such a scenario might also be more typical of an ambient network environment. Several metrics can characterize the level of activity. We use the regularity of the users' presence in the network,

as measured by the number of active days. The number of users in our data that are active all 45 days is 835. We consider these users as a pool from which we sample for each simulation run.

As in Fig. 1, but here only for the most active users, Fig. 2 shows distributions that characterize the users' behavior. We can see that this subset of users is more active than the other. The mean presence time for the period is 609.3 hours in total and 13.13 hours per day (see Fig. 2(c) and Fig. 2(d)), as opposed to 243 hours and 5.18 hours with all the users. Users visited on average 20.65 locations in the period (see Fig. 2(a)) and 2.96 locations per day (see Fig. 2(b)), as opposed to 16.66 and 1.75 with all the users.



(a) Locations visited

(b) Locations visited per day

(c) Connection time

(d) Connection time per day

Fig. 2. Statistics on the most active users data set.

	delivery ratio (%)	**delay** (days)	**route length** (hops)
Epidemic	97.8 ± 1.0	3.1 ± 0.4	8.6 ± 0.2
Opportunistic	10.4 ± 1.4	19.6 ± 1.9	1.0 ± 0.0
Random	13.7 ± 1.7	18.4 ± 1.6	3.5 ± 0.2
Potato	37.6 ± 1.0	20.0 ± 0.3	321.0 ± 30.0
MobySpace	46.6 ± 1.1	20.2 ± 2.0	5.3 ± 0.2

TABLE II

RESULTS WITH THE MOST ACTIVE USERS.

Table II shows the simulation results. Considering only the most active users, more bundles are delivered by the algorithms. MobySpace attains a delivery ratio of 46.6% instead of 14.9%. Note that, with respect to what we observed, the delivery ratio of MobySpace would have been higher if more nodes had participated in the scenario. However, we were limited by computational issues. The average bundle delay achieved is very low for Epidemic compared to the other algorithms. Route lengths are shorter than Epidemic's for Opportunistic, Random, and MobySpace, whereas the average length is higher for Potato compared

to the previous scenario with randomly sampled users.

The difference between the two scenarios can be seen most clearly by looking at the delivery ratio of Epidemic. As shown by the delivery ratio of 82% obtained by Epidemic when samples are selected among all the users, the level of presence and the mobility of nodes were often not sufficient to achieve proper routing under any circumstances. Otherwise, Epidemic should have delivered 100% of the bundles. Either some of the source-destination pairs were never linked by a path over time, or certain sources and destinations were simply not sufficiently present in the data. On the other hand, when selecting only the most active users, Epidemic achieves a delivery ratio of 97.8%. The small portion of bundles not delivered comes from the fact that, in a few cases, we sampled nodes that had no interactions with the others.

These results confirm that the MobySpace evaluated in this paper enhances routing as compared to various generic approaches in an ambient network formed by users carrying personal devices in a campus setting. MobySpace achieves a high delivery ratio compared to simple algorithms like Opportunistic, Random, or Potato.

V. Controlled flooding strategies

The previous section has shown that MobySpace routing in single-copy mode far outperforms the other single-copy protocols but still delivers half as many bundles as Epidemic. The question arises: can an intermediate scheme provide many of the benefits without the same overhead as Epidemic? We investigate here what would be the performance if MobySpace is used to guide a controlled flooding scheme. We describe the methodology we used for this study and we show, with simulation results, that low overhead can be achieved using MobySpace while having a delivery ratio roughly similar to that of Epidemic.

A. Methodology

Similarly to the previous evaluations, we compare MobySpace-based controlled flooding solutions to other well-known multi-copy strategies. We first present these well-known schemes, since the ones that use MobySpace are variants:

- *Spray and Wait*: Unless it meets the destination, the source transmits a copy of the bundle it carries to the N first nodes met. These nodes are used as relays but only transmit the bundle to the destination if encountered.
- *TTL based*: The source uses a simple Epidemic scheme but with a TTL equal to T in order to only reach relays that are at most T hops away.
- *Probabilistic flooding*: The source floods its bundle like in Epidemic. However, each relay only transmits N copies to the first nodes met that do not have already it, with a probability P. Otherwise, relays act as *passive* relays like in Spray and Wait.

We will refer in the rest of the paper to these algorithms as respectively *Spray*, *F_TTL*, *Proba*. The routing strategies based on MobySpace we evaluated are the following:

- *MobySpace Spray to Closer and Wait*: This algorithm is similar to Spray, but here, copies of a bundle are distributed only to the first N nodes met that are closer to the destination in the MobySpace. This design choice has been motivated by the fact that we want to increase the utility of each transmission.
- *MobySpace Spray to Closest and Wait*: This scheme acts similarly to *MobySpace Spray to Closer and Wait*, but copies are distributed one after another to nodes that are closer and closer to the destination in the MobySpace.
- *MobySpace Spray to Closer and Route*: Once distributed to relays in the same fashion as *Spray to Closer and Wait*, bundles are normally routed with the basic MobySpace single-copy scheme toward the destination.
- *MobySpace Spray to Closest and Route*: Once distributed to relays in the same fashion as *Spray to Closest and Wait*, bundles are normally routed with the basic MobySpace single-copy scheme toward the destination.

- *MobySpace Probabilistic flooding*: The difference with Proba is that when bundles are sprayed at relays, copies of bundles are distributed only to nodes being closer to the destination in the MobySpace than the current relay.
- *MobySpace Epidemic*: In this scheme, bundles are flooded like in the former Epidemic but are only transmitted from one node to another if the node is closer to the destination in the MobySpace.

We refer to these schemes as *MSpray*, *MSprayT*, *MSprayRoute*, *MSprayTRoute*, *Mproba*, *MEpidemic*.

B. Simulation results

We performed simulations exactly as in Sec. IV. We used the same seeds in the random number generator and the same subset of most active users that we identified in the Dartmouth data.

While in the evaluation of single-copy schemes, the overhead of protocols (except for Epidemic) was directly linked to the route lengths they induced, this is no longer the case with controlled flooding schemes. As a consequence, we use a new metric in this section, which is the total number of transmissions that occurred before bundle delivery (or non delivery for those that never reached their destination).

Table III presents the simulation results. This table is divided into three parts: (1) results for the main protocols we evaluated in Sec. IV, (2) results for the usual multi-copy algorithms and (3) results for MobySpace-driven controlled flooding solutions.

Epidemic and Opportunistic show the two extremes. The first one delivers 97.8% of bundles with an average delay of 3.1 days, an average route length of 8.6 and $74{,}674.0$ transmissions. The second one only delivers 10.4% of bundles with an average delay of 19.6 days, an average route length of 1.0 and 52.2 transmissions. As seen previously, the delivery ratio for MobySpace is 46.6% which is right in between these two extremes but with an overhead closer to Opportunistic with $2{,}291.6$ transmissions.

Looking at the well-know multi-copy schemes we evaluated, we observe that the higher the number of copies sent into the network, the better the delivery ratio and the higher the overhead. None of these solutions outperforms MobySpace in delivery ratio; they all lead to a higher overhead for an equivalent delivery ratio. For instance, Spray with $N = 5$ achieves 50.0% delivery but with an overhead of $4{,}618.2$ which is more than twice as much as MobySpace.

In some of the cases, MobySpace-based controlled flooding solutions improve delivery ratio while leading to lower overhead. For instance, with 5 copies distributed Spray achieves 35.5% delivery with $2{,}554.8$ transmissions while MSpray obtains 42.2% delivery with only $2{,}344.8$ transmissions.

In other cases, especially when the number of copies distributed is high (e.g., Spray and MSpray with $N = 50{,}100$, or Proba and MProba with $N = 5$, $P = 0.6$), the MobySpace-based solutions show a lower delivery ratio but lead to a significantly reduced overhead. MSpray with $N = 5$ leads to half as much overhead while delivering only 6.3% fewer bundles. This is due to a lack of opportunism of MobySpace based schemes in their forwarding decisions. The average delay suffers also from this lack. The delay for MProba with $N = 5$ and $P = 0.6$ is 8.1 days higher than Proba with the same parameters.

MSprayRoute shows encouraging performance. With only 5 copies distributed, it achieves 80.8% of delivery with an overhead of only $7{,}816.4$. The two variants MSprayT and MSprayRouteT have lower overheads compared to their homologous MSPray and MSprayRoute but with, as expected, lower delivery ratios. Note also that they were only feasible with $N = 5$ and $N = 10$ because they were not able to distribute 30 copies. There were only a few opportunities for sources to transfer bundles to nodes that are closer and closer to the destination in the MobySpace.

MFlooding has one of the best results, it delivers 92.8% of bundles with an average delay of 9.9 days while only using $15{,}140.2$ transmissions (80% less than Epidemic).

Fig. 3 highlights the trade-off that exists between the proportion of undelivered bundles and the overhead for each class of protocols we evaluated. We see that the trade-off takes a concave form, going from the upper left part with Opportunistic to the lower right part with Epidemic. Since our goal is to minimize both the overhead and the number of bundles not delivered, this plot shows that MobySpace-based solutions

	N	P	T	delivery ratio (%)	delay (days)	Overhead (transmissions)	route length (hops)
Epidemic				97.8 ±1.0	3.1 ±0.4	74,674.0 ±1378.3	8.6 ±0.2
MobySpace				46.6 ±1.1	20.2 ±2.0	2,291.6 ±140.1	5.3 ±0.2
Opportunistic				10.4 ±1.4	19.6 ±1.9	52.2 ±6.9	1.0 ±0.0
Spray	5			35.5 ±2.0	18.4 ±1.4	2,554.8 ±37.8	1.8 ±0.0
	10			50.0 ±3.6	17.6 ±1.0	4,618.2 ±98.6	1.9 ±0.0
	30			67.9 ±3.4	17.8 ±0.7	8,525.2 ±107.5	1.9 ±0.0
	50			69.4 ±3.0	18.1 ±0.7	9,248.4 ±166.2	1.9 ±0.0
	100			69.6 ±3.0	18.1 ±0.8	9,419.6 ±158.3	1.9 ±0.0
F_TTL			2	69.6 ±3.0	18.1 ±0.8	9,421.2 ±159.2	1.9 ±0.0
			3	93.6 ±1.8	12.2 ±1.0	25,926.8 ±874.6	2.9 ±0.0
			4	97.3 ±1.3	7.9 ±0.7	39,452.8 ±1,426.7	3.7 ±0.0
			6	97.8 ±1.0	4.6 ±0.5	55,391.0 ±1,413.7	5.2 ±0.0
Proba	2	0.1		23.6 ±2.3	19.4 ±1.3	1,334.4 ±46.7	1.8 ±0.1
	2	0.2		26.2 ±1.9	19.5 ±1.8	1,700.2 ±51.1	2.1 ±0.1
	2	0.3		31.3 ±2.0	18.4 ±1.5	2,436.0 ±98.3	2.6 ±0.1
	5	0.1		44.9 ±2.3	17.5 ±1.0	4,378.6 ±82.4	2.4 ±0.0
	5	0.2		61.6 ±2.7	15.5 ±1.0	10,911.6 ±386.0	3.6 ±0.0
	5	0.3		80.2 ±3.0	12.3 ±0.8	26,808.6 ±1,562.7	5.6 ±0.2
	5	0.6		96.2 ±1.5	5.5 ±0.5	58,492.4 ±732.2	7.8 ±0.1
	10	0.5		97.1 ±1.3	6.1 ±0.7	55,635.8 ±1209.4	6.7 ±0.1
MSpray	5			42.2 ±2.5	18.5 ±0.7	2,344.8 ±58.4	1.9 ±0.0
	10			53.6 ±2.8	18.4 ±0.7	3,785.2 ±98.0	1.9 ±0.0
	30			63.0 ±3.9	19.2 ±0.8	5,380.0 ±188.1	1.9 ±0.0
	50			63.1 ±3.9	19.2 ±0.8	5,442.8 ±194.1	1.9 ±0.0
	100			62.1 ±1.8	19.3 ±0.7	5,363.0 ±134.0	1.9 ±0.0
MSprayT	5			43.6 ±1.9	19.9 ±0.7	1,743.0 ±64.9	1.9 ±0.0
	10			44.9 ±1.9	20.1 ±0.7	1,814.0 ±77.9	1.9 ±0.0
MSprayRoute	5			80.8 ±4.1	15.4 ±1.0	7,816.4 ±269.6	4.9 ±0.0
	10			85.5 ±3.3	13.7 ±1.1	11,853.6 ±424.4	4.7 ±0.0
	30			88.6 ±2.7	13.2 ±1.0	17,047.4 ±763.8	4.5 ±0.1
	50			88.7 ±2.7	13.2 ±1.0	17,921.2 ±784.0	4.5 ±0.1
	100			88.7 ±2.7	13.2 ±1.0	18,044.0 ±826.5	4.5 ±0.1
MSprayTRoute	5			71.7 ±3.4	17.4 ±0.9	4,507.8 ±211.8	4.3 ±0.1
	10			72.4 ±3.5	17.4 ±0.9	4,547.6 ±218.7	4.3 ±0.1
MProba	2	0.1		31.0 ±2.6	19.1 ±1.2	1,291.0 ±36.5	2.0 ±0.1
	2	0.2		34.6 ±3.2	19.2 ±1.0	1,526.0 ±36.3	2.2 ±0.1
	2	0.3		40.8 ±2.5	19.0 ±1.2	1,908.4 ±50.0	2.5 ±0.1
	5	0.1		51.3 ±3.1	18.6 ±0.8	3,235.0 ±82.7	2.3 ±0.0
	5	0.2		61.5 ±2.9	17.6 ±1.0	4,473.0 ±106.9	2.7 ±0.1
	5	0.3		70.4 ±2.0	16.9 ±1.0	5,819.8 ±192.1	3.0 ±0.1
	5	0.6		87.6 ±2.4	13.6 ±1.0	10,154.0 ±465.7	3.9 ±0.1
	10	0.5		87.5 ±2.5	13.4 ±1.0	11,590.6 ±743.0	3.6 ±0.0
MFlooding				92.8 ±1.6	9.9 ±1.2	15,140.2 ±981.9	4.4 ±0.1

TABLE III
SIMULATION RESULTS FOR CONTROLLED FLOODING SCHEMES.

such as MSprayRoute with $N = 5$ and MFlooding, which are in the bend of the curve, tend to reach our main objective.

These results have shown that allowing multiple copies to be sent, while maintaining MobySpace's primary objective, which is to get closer at each transmission to the destination in the MobySpace, is a real benefit for the performance of DTN routing schemes.

However, as we have seen, the kind of knowledge about node mobility that we used for routing does not allow us, on the data we used, to achieve the same performance as Epidemic, in delay especially. More knowledge, or knowledge that better characterizes node mobility, would certainly improve routing performance. The trade off is then to find the most relevant information and its most efficient use

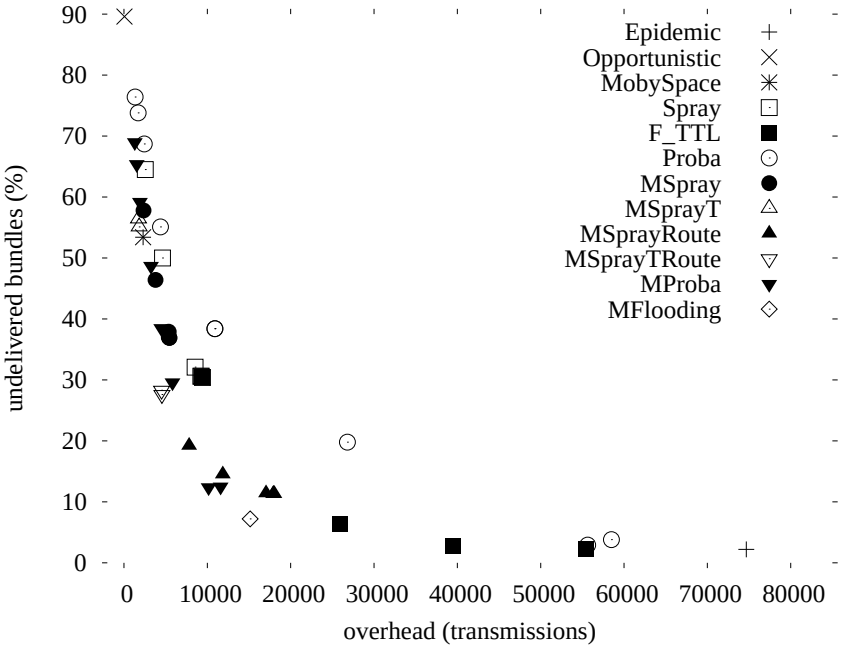


Fig. 3. Trade-off between delivery and overhead.

for routing, without impacting the network too much in terms, for instance, of computation power required by nodes or amount of information shared among nodes. Furthermore, the difference between the results obtained with MobySpace and that of Epidemic might be explained by the fact that a large number of interactions are simply not predictable because of the complexity hidden in node connectivity patterns. MobySpace-based solutions are not able to take advantage of these unpredicted interactions, while Epidemic is.

VI. Related work

The case of only opportunistic contacts has been analyzed by Vahdat and Becker [6] using the epidemic routing scheme that consists of flooding. In order to control flooding in DTN, Spyropoulos et al. have introduced the Spray and Wait [16] protocol that distributes a number of copies to relays and then waits until the destination meets one of them. Harras et al. [17] have evaluated simple controlled message flooding schemes with heuristics based, for instance, on hop limits or timeouts. They also introduce a mechanism based on bundle erasure. Once a message arrives at the destination after basic flooding, the remaining copies in the buffers of other nodes are erased with the proposed scheme. Wang et al. [18] reincode the messages with erasure codes and distribute their different parts over a large number of relays, so that the original messages can be reconstituted even if not all bundles are received. Chen et al [19] have proposed a solution that combines the strength of the previous erasure coding scheme and the advantages of aggressive forwarding as in epidemic routing. Widmer et al. [20] have explored network coding techniques. All these approaches distribute multiple copies of bundles, they ensure a high reliability of delivery, and a low latency, but they imply high buffer occupancy and high bandwidth consumption.

A large amount of work concerning routing in DTNs has also been performed with predicted contacts, such as the algorithm of Lindgren et al. [21], which relies on nodes having a community mobility pattern. Nodes mainly remain inside their community and sometimes visit the others. As a consequence, a node may transfer a bundle to a node that belongs to the same community as the destination. This algorithm has been designed as a possible solution to provide Internet connectivity to the Saami [22] population who live in Swedish Lapland with a yearly cycle dictated by the natural behavior of reindeer. In a similar manner, Burns et al. [23] propose a routing algorithm that uses past frequencies of contacts. Also making use of past contacts, Davis et al. [24] improved the basic epidemic scheme with the introduction of adaptive

dropping policies. Recently, Musolesi et al. [25] have introduced a generic method that uses Kalman filters to combine and evaluate the multiple dimensions of the context in which nodes are in order to take routing decisions. The context is made of measurements that nodes perform periodically, which can be related to connectivity, but not necessarily. This mechanism allows network architects to define their own hierarchy among the different context attributes.

VII. Conclusions and future work

The main contribution of this paper has been the validation of a generic routing scheme that uses the formalism of a high-dimensional Euclidean space constructed upon mobility patterns, the MobySpace. We have shown, through the replay of real mobility traces, that it can applied to DTNs and that it can bring benefits in terms of enhanced bundle delivery and reduced communication costs. We have evaluated the use of MobySpace not only for single-copy routing, but also as a means to drive and improve existing basic controlled flooding solutions.

Future work along these lines might include studies concerning the impact of the structure of the Euclidean space, i.e., the number and type of dimensions, and the similarity function. Different kinds of Euclidean space can be investigated by considering schemes like the one described in Sec. III that takes for each dimension the frequency of contacts between a certain pair of nodes or the one that captures cyclic frequential properties during nodes' visits to locations.

Work also remains to be done on the stability of mobility patterns over time and their ability to be learned by nodes. The patterns may contain long term and short term dependencies, as pointed out by Ghosh et al. [26]. Nodes can have different mobility patterns that are each stable. For instance, they can have one for the week-ends, one for the vacations, and one for working weeks.

Additionally, further validations is needed on real data and in different environments. MobySpace can be tested on traces coming from larger cell networks, like GSM networks. We might also want to evaluate MobySpace in different social contexts where nodes have specific mobility patterns.

Acknowledgments

We gratefully acknowledge David Kotz for enabling our use of wireless trace data from the CRAWDAD archive at Dartmouth College. We thank Marc Giusti and Pierre Lafon at the STIX laboratory (École Polytechnique / CNRS) for access to the machines we used for the simulations. This work was supported by LiP6 and Thales Communications through their joint research laboratory, Euronetlab, and the CIFRE grant 135/2004 provided by the ANRT (Association Nationale de la Recherche Technique).

References

[1] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. SIGCOMM*, 2003.
[2] J. Leguay, T. Friedman, and V. Conan, "DTN routing in a mobility pattern space," in *Proc. WDTN*, 2005.
[3] J. Leguay, T. Friedman, and V. Conan, "Evaluating mobility pattern space routing for DTNs," in *Proc. INFOCOM*, 2006.
[4] "Delay Tolerant Network Research Group (DTNRG)," `http://www.dtnrg.org`.
[5] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay tolerant network architecture, IRTF draft, draft-irtf-dtnrg-arch-02.txt," July 2004.
[6] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Tech. Rep. CS-200006, Duke University, April 2000.
[7] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in *Proc. WDTN*, 2005.
[8] M. McNett and G. M. Voelker, "Access and mobility of wireless PDA users," Tech. Rep., UC San Diego, 2004.
[9] T. Henderson, D. Kotz, and I. Abyzov, "The changing usage of a mature campus-wide wireless network," in *Proc. Mobicom*, 2004.
[10] M. Balazinska and P. Castro, "Characterizing Mobility and Network Usage in a Corporate Wireless Local-Area Network," in *Proc. MobiSys*, 2003.
[11] L. Song, D. Kotz, R. Jain, and X. He, "Evaluating location predictors with extensive Wi-Fi mobility data," in *Proc. Infocom*, 2004.
[12] C. Lonvick, "The BSD syslog protocol," RFC 3164. IETF, August 2001.
[13] E. P. C. Jones, L. Li, and P. A. S. Ward, "Practical routing in delay-tolerant networks," in *Proc. WDTN*, 2005.
[14] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on the performance of opportunistic forwarding algorithms," in *Proc. INFOCOM*, 2006.

[15] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *Proc. SIGCOMM*, 2004.
[16] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks," in *Proc. WDTN*, 2005.
[17] K. Harras, K. Almeroth, and E. Belding-Royer, "Delay tolerant mobile networks (DTMNs): Controlled flooding schemes in sparse mobile networks," in *Proc. Netwoking*, 2005.
[18] Y. Wang, S. Jain, M. Martonosi, and K. Fall, "Erasure coding based routing for opportunistic networks," in *Proc. WDTN*, 2005.
[19] L. Chen, C. Yu, T. Sun, and H. Chu Y. Chen, "A hybrid routing approach for opportunistic networks," in *Proc. CHANTS*, 2006.
[20] J. Widmer and J. Le Boudec, "Network coding for efficient communication in extreme networks," in *Proc. WDTN*, 2005.
[21] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," in *Proc. SAPIR*, 2004.
[22] A. Doria, M. Uden, and D. P. Pandley, "Providing connectivity to the Saami nomadic community," in *Proc. Developement by Design Conference*, 2002.
[23] B. Burns, O. Brock, and B. N. Levine, "MV routing and capacity building in disruption tolerant networks," in *Proc. Infocom*, 2005.
[24] J. A. Davis, A. H. Fagg, and B. N. Levine, "Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks," in *Proc. ISWC*, 2001.
[25] M. Musolesi, S. Hailes, and C. Mascolo, "Adaptive routing for intermittently connected mobile ad hoc networks," in *Proc. WOWMOM*, 2005.
[26] J. Ghosh, M. J. Beal, H. Q. Ngo, and C. Qiao, "On profiling mobility and predicting locations of campus-wide wireless network users," Tech. Rep. CSE-2005-27, State University of New York at Buffalo, December 2005.