A Hybrid Overlay Network for Bulk Data in Challenged Networks

Azarias Reda
University of Michigan
Ann Arbor, Michigan
azarias@umich.edu

Brian Noble
University of Michigan
Ann Arbor, Michigan
bnoble@umich.edu

ABSTRACT

Developing countries face significant challenges in network access, making even simple network tasks unpleasant. Transferring bulk data in these networks is often prohibitively difficult, rendering useful information out of reach for many people. This paper introduces phoretic networking, an approach for orchestrating bulk data transfer through a series of hosts with spare storage and diverse network connectivity. By combining natural individual mobility and available network connectivity, phoretic networking forms a hybrid overlay network for delivering bulk data while preserving scalability in the state required to do so. Our implementation of a phoretic network, *Bati*, outperforms earlier attempts in using mobility for data delivery and in-network storage for enhanced path selection. Our evaluation demonstrates substantial savings in using Bati for delivering bulk data, transferring an order of magnitude more data than the network alone, and improving the delivery rate by more than 40% compared to popular ad-hoc networks.

1. INTRODUCTION

Developing countries face significant challenges in network access. Bandwidth is the cornerstone of today's global knowledge economy, but it is scarcest where it is most needed—in the developing nations of the world which require low-cost communication to accelerate their socioeconomic development [16]. Connectivity is very expensive in these countries—A 1 Mbps connection in the US costs about 25 USD per month, whereas a similar connection in rural Zambia costs about 1,700 USD [22]. The slightly more common dial-up connections, often provided through internet kiosks, are usually shared among multiple users, with effective end user bandwidth hovering around 10-15 Kbps [25].

On the other hand, the digital universe has been rapidly expanding, and bulk data makes more than 70% of the traffic on the internet [2, 9]. Unfortunately, much of this information is not immediately usable in many developing counties due to the poor network conditions. For example, downloading a 10MB audio lecture from MIT's OpenCourseWare would take more than 2 hours to complete in a kiosk, rendering it effectively out of reach for a student who has to pay by the minute. A 100MB video lecture is a non-starter.

Phoretic networking addresses this problem by leveraging two complimentary mechanisms—the *natural mobility* of users along with the *diverse, intermittent connectivity* available to them as they move. Phoretic networking combines and expands on ideas from delay-tolerant and ad hoc networking, with an eye towards scalability in the face of highly uncertain endpoints and the movements and connectivity between them. Data can travel with moving principals using excess storage, or through clusters of well-connected machines; weak connectivity is used for control plane activity, providing efficient resource management. Our implementation of this idea, *Bati*, requires no global state, no message or acknowledgement flooding, and can tolerate imperfect predictions in individual nodes' destinations, arrival times, and arrival frequencies.

The central observation in Bati is that principals have a small number of locations that they frequently visit. Each node tracks its K most popular destinations, and the expected inter-arrival time between them. To compute the distance between two destinations, the total flow of principals between them is combined. However, Bati limits the state required to do so by remembering only the most frequent destinations in combination precisely, and approximating the remainder with a conservative estimate of infrequent destinations. This preserves the scalability of the system, devoting resources to the most likely routing alternatives.

This mobility model of "distance" is augmented by recognizing that even within the developing world, clusters of good connectivity exist and local connectivity often far exceeds remote connectivity. Routing to any destination within a cluster is equivalent to reaching all of them. In turn, data can be *staged* within specific nodes in a cluster to take advantage of mobility patterns that are not precisely rooted at the origin node, but that can be reached easily by intra-cluster means. A single metric combines the contributions of mobility and network connectivity for routing decisions.

Because delivery is uncertain, Bati allows multiple packets in flight to any particular destination. Rather than try to precisely notify all nodes of delivery, Bati employs an inverted-ACK technique. Data starts with a soft timeout, which is either checked or (in the ab-

sence of any connectivity) extended as necessary. These control-plane operations can be done even in resource-poor networks typically found in the developing world with only a nominal overhead.

Bati is implemented as a cross-platform, user level library atop an unmodified network stack. To evaluate Bati, we have developed a mobility emulation platform based on the EmuLab network emulation framework. The benefit of emulation over simulation was that we could write code that can be directly deployed in the wild, rather than geared towards a specific simulator—our platform is available to others for use in evaluating similar systems. Bati provides an order of magnitude improvement in throughput compared to the network alone, and over 40% improvement in delivery rate compared to solutions relying only on node mobility.

2. RELATED WORK

Related work comes from two principal areas—ad-hoc networking and delay/disruption tolerant networks. Ad-hoc networks are concerned with cases where central coordination is not possible, and infrastructure is not present. Nodes must self-organize to deliver messages, often communicating with some radio only when in range. Delay tolerant networks form a confederation of regional networks with in-network storage for delivering messages across boundaries. In the remainder of this section, we will look at several systems from each class and describe how our work relates to them.

Earlier approaches in ad-hoc networks used epidemic style forwarding among mobile nodes [10, 32], which was inefficient in resource utilization. Subsequent approaches [3, 4, 5, 19, 21] improve on epidemic routing by prioritizing how likely packets are going to be delivered from each node, taking into account how often nodes meet each other. Generally, these systems do not focus on connectivity beyond direct wireless links. Some require a form of centralized resource management [13], or flood the network for global information and acknowledgment [4]. In addition, they require the routing computation and state kept at each node to grow linearly with the system, presenting difficulties in scaling to a wide-area system in the developing world. Bati builds on mobility lessons learned from these and similar systems, and integrates them into a hybrid network that can use a diverse set of links for delivering messages between nodes, without requiring linear routing state at each node. Bati uses available weak network links for opportunistic transfer of routing information, and good connectivity for route shortcuts and last-mile delivery of messages.

Delay tolerant networks provide an in-network store and forward architecture to identify the shortest path between two nodes in a poorly connected network. The key in accomplishing this goal is finding what nodes to forward messages to, and what link to use. Initial approaches used manually filled routing tables with default links for DTN routers [8]. Other iterations use centralized registers for locating nodes [29], or oracles that can answer system level questions [15], such as the average wait time for a periodic link. Using this information, they assign costs to the links available between nodes, and run a modified version of Dijkstra's shortest path algorithm to determine next hops. While it is unclear how to provide these times under natural mobility, perhaps a more significant difference is that these approaches often assume the endpoints of links are known, and it is only a matter of time until a link becomes available. In Bati's target domain, mobility is unstructured, with uncertain endpoints.

Further improvements in DTN routing looked at reducing the reliance on oracles [17]. The Minimum Estimated Expected Delay (MEED) algorithm improves on the Minimum Expected Delay (MED) algorithm that required on an oracle for discovering the expected delay of links between nodes. MEED estimates this value based on the history of the link. However, this still assumes that link destinations are known ahead of time. As a result, the kinds of mobility most utilized in these systems are fixed schedule principals such as buses [12, 29]. Our system focuses on uncertainty in mobility along with diversity in network links for delivering bulk data. Furthermore, since MEED needs to know the topology of the network, given by the estimated delay of every link between every pair of nodes, it does epidemic routing of link state messages. This might be unduly expensive in low-bandwidth, challenged networks.

DTLSR, a delay tolerant routing protocol for developing regions [7], focuses on network-based routing of packets in challenged networks. The main insight in this work is that there is an underlying stability in the network topology in developing regions, and it can be exploited for making routing decisions. Considering this, they suggest including even failed network links in their MEED style shortest path calculation with a cost proportional to how long a link has failed. DTLSR is a network-only protocol, ignoring principal mobility, and requires link-state announcements.

Our work improves on these systems by integrating the nuanced use of a diverse set of network links with unstructured natural mobility, constructing a hybrid overlay network that can be used in two complementary ways. It can deliver bulk data between peers within challenged environments, or to and from well connected ends. Bati runs on an unmodified network stack, and probabilistically learns principal mobility in the system.

3. DESIGN

For an outside observer who is not aware of an individual's motivation and schedule, human mobility can easily appear to be random and unpredictable. Yet, despite our deep-rooted desire for spontaneity and change, our daily mobility can be characterized by a deep-rooted regularity. The success of a phoretic networking archi-

tecture hinges on marrying the strengths of a diverse set of network links with a sufficiently accurate model for natural human mobility. Luckily, scientific analysis shows true randomness in human behavior to be a rather infrequent phenomenon [30]. As to the diverse set of available network links, we show a broad classification of capacity can go a long way in establishing an efficient division of *purpose*.

Bati implements these principles in building a hybrid overlay network for transferring bulk data in challenged environments. First, lets establish some definitions for terms used in the paper. *Nodes* are stationary machines within the system. Data pieces, called *envelopes*, make up the units of bulk data delivered through Bati. These envelopes could either be carried by individuals that move among nodes, called *principals*, or dispatched using an available network connectivity. A principal could be associated with a *home node*, which serves as its current address for delivering envelopes sent to it. Principals carry mobile devices, called *data capsules*, with some specified storage capacity. Nodes and data capsules might have a potentially diverse set of network links available to them. When a principal sends an envelope through Bati, there are three important constraints: shorter delivery time is desirable, good network connectivity is not always an option and storage on a data capsule is limited.

We use an individual mobility model to represent how principals move among nodes, and incrementally update a distributed, system-level transition model that represents how close nodes are connected with each other as a result of mobility. All available network connectivity is also utilized in delivering data. Some links might be best suited for shipping data, while others might be more appropriate in orchestrating roles. The following sections describe how Bati parses mobility, builds an efficient and distributed transition model, and exploits link diversity in delivering data.

3.1 Individual Mobility

People are creatures of habit and often have certain repeated patterns that can be learned probabilistically. Several projects use Markov models for mobility prediction [6, 31]. While a second order Markov model has been shown sufficient for predicting a principal's short term mobility, it often falls short when considering more than a few steps in the future [24]. In particular, we are more interested in where a principal is going to be in a day—or a week—than in the next few minutes. As a result, a Markov model does not quite capture the information we would like to have for use in our system.

Instead, we use results from a large scale study of individual mobility patterns that looked at a trajectory of more than 100,000 anonymized mobile phone users whose location was tracked for a six-month period [11]. The study finds that human trajectories show a high degree of temporal and spatial regularity, each individual being characterized by a time-independent distinctive travel distance and a significant probability of return to a few highly frequented locations. In particular, the cumulative return probability of an individual to a previous location is characterized by several peaks at 24, 48 and 72 hours. In addition, the probability of finding a user a location with rank L, where L represents the L^{th} most visited location for an individual, is well approximated by $P(L) \sim \frac{1}{L}$, independent of the number of locations visited.

In our model, we start out by keeping track of the average interarrival time for the top-K nodes a principal visits, and how long it has been since the last visit. The interarrival time is defined as the time it took for a principal to get back to a node when at least one other node was visited in the interim. This gives us the notion of nodes a principal is likely to visit soon. We combine this with the observed standard deviation of the interarrival time to bound the likelihood of a visit to a node in a given time window as follows: when a principal visits a node, we order the top K other nodes in the principal's history based on the latest estimated time the principal is going to visit the nodes with a threshold probability P. Intuitively, as the principal moves in the system, the nodes it is likely to visit soon start bubbling up to the top of the list. For each node in the principal's top-K list, the model provides:

$$M[n] = \{\mu_t, \sigma_t, T, C\} \quad (1)$$

Where μ_t is the average interarrival time, σ_t is the standard deviation and T is time of last visit. C is the confidence score that describes how similar the recent trend has been to the overall pattern. It is measured by recording how many of the last N observed visits occurred within the estimated bound.

We have adapted the model to deal with phase changes and new patterns. If a principal visits a node before the due time, the average interarrival time and the standard deviation are affected accordingly. On the other hand, if the principal is significantly past-due to a node, it is reflected in the system by extending the average expected time to the node by a multiple of the standard deviation. This extension gets incrementally bigger to reflect more permanent changes in patterns. We use the confidence score to gauge how much the parameters are altered: the lower the confidence score, the more drastically the parameters are changed, allowing for quick learning. While we have considered adding route fingerprints to deal with *multiple, established* sub-patterns of a principal with respect to a single node, our empirical analysis in section 5 suggests that the current model is sufficient for capturing mobility patterns in our system.

3.2 Collective Mobility

Another important piece in our system is the collective transition model that describes the *mobility connectedness* of a node to others in the system. As a node is visited by a number of principals, the transition

model is incrementally updated to reflect the current state of the system.

Consider a principal Z that moves from node A to B with an average trip time $T_{(Z,A\rightarrow B)}$. We define the quantity $\delta_{(A\rightarrow B)}$ to represent the closeness from A to B. $\delta_{(A\rightarrow B)}$ describes the *avergage, collective* trip time in going from A to B due to all principals that visit node A. For our purposes, the average trip time of a principal between two nodes can be sufficiently estimated from its interarrival times at the nodes without having to keep a separate record.

If there was only one principal in the system, we can assign:

$$\delta_{(A\rightarrow B)} = T_{(Z,A\rightarrow B)} \tag{2}$$

Now imagine another principal, Y, that also moves between A and B with an average trip time $T_{(Y,A\rightarrow B)}$. We will need to update the relative closeness factor, $\delta_{(A\rightarrow B)}$ between A and B, to reflect the component contributed by the new principal.

This is equivalent to calculating the average waiting time for a bus between two stations when there are busses running at different intervals. If the first bus leaves from the station every n seconds, and the second bus leaves every m seconds, both with random start times, it can be shown that the average waiting time, W, for any bus leaving the station is:

$$W = \frac{n \times m}{n + m}$$

Using the same approach, the closeness as a result of two principals Z and Y, moving between A and B is given as:

$$\delta_{(A\rightarrow B)} = \frac{T_{(Z,A\rightarrow B)} \times T_{(Y,A\rightarrow B)}}{T_{(Z,A\rightarrow B)} + T_{(Y,A\rightarrow B)}} \tag{3}$$

This can be generalized so that we can incrementally update the relative closeness between any two nodes as we learn more about principal mobility in the system. Given the old closeness between two nodes as $\delta_{(old)}$, if a principal's expected trip time was adjusted from $T_{(old)}$ to $T_{(new)}$, the resulting new collective closeness as a result of this update, $\delta_{(new)}$, can be shown to be:

$$\delta_{(new)} = \frac{1}{\frac{1}{\delta_{(old)}} + \frac{1}{T_{(new)}} - \frac{1}{T_{(old)}}} \tag{4}$$

This gives us a simple way to gauge and update closeness between nodes as principals move in the system, and we study their mobility patterns. Closeness is calculated in a distributed manner where each node is responsible for maintaining part of the transition model that describes how connected the node is to its top-K closest nodes in the system. K is by default 25, but could be configured according to space availability. To keep track of the long tail of not-so-well-connected nodes, we use attenuated Bloom filters, in a manner similar to OceanStore [28]. We have an n-width array of standard Bloom filters, where the nth filter represents an nth level of connectedness – giving us a reference-table like property. Since these filters are used to represent nodes in the tail that are not well connected to the current node, the small false positive rate from the Bloom filter is easily outweighed by the space savings.

Since nodes cannot be expected to always be available for updating and querying their slice of the transition model, we use principals as carriers of not only data, but also system transition information. This is possible because a principal visits only a small fraction of all nodes in the system. As a principal visits one of its top-N frequented nodes, it makes a copy of the most recent slice of the transition model kept at the node. If an update to a node is not possible because the node is currently offline, the update is kept on the principal, and merged later when the principal visits the node. On the other hand, when making routing decisions, the latest copy carried by the principal is used. This copy will be slightly out of date, but the freshness of the model is proportional to the frequency at which the principal visits the node in question. As a result, the most recent copies will be from nodes the principal visits often, and will probably visit soon as well, while the most stale copies will be from less frequented nodes—a desirable property.

3.3 Network usage

While bandwidth is scarce in developing regions, not all nodes in the system are poorly connected to the network, and local connectivity usually far exceeds remote connectivity. In addition, most nodes have a low bandwidth, high latency link that can be selectively used when available. Bati can utilize all available connectivity to aid in the delivery of messages in this hybrid network. In particular, when available, low bandwidth links are used for maintaining more up to date routing information at nodes, while the occasional fat pipe is used for routing shortcuts and last-mile delivery.

Bati uses weak links in two ways. First, a node can subscribe for remote network updates about changes in a principal's mobility patterns, subject to network capacity. This subscription is tiered so that the weaker a node's link, the fewer network updates it will receive. This is an optimization, as updates can always be merged when the principal re-visits the node. Second, weak links are used to opportunistically acknowledge envelopes. Since network resources are not always available or reliable, synchronous acknowledgment of envelopes is not practical, and should not be required for correctness. Instead, Bati uses opportunistic, collective, and asynchronous ACKs for space optimization, as an acknowledged envelope no longer needs to be kept in the buffer or be transmitted to future principals. From the node's standpoint, an envelope is kept in the buffer until it is ACKed or eventually kicked out by envelopes that are more likely to be delivered.

The fundamental difficulty in ACKing in networks

like Bati is route multiplicity. Traditional ACKs, where nodes acknowledge envelopes as they receive them, do not work well here because a node cannot truly ACK an envelope without flooding the system, as more principals with extra copies could keep showing up. Instead, Bati uses *inverted ACKing*, a simple and resource customizable strategy targeted at decentralized delivery systems. When envelopes are inserted in the system, they are given a time-till-acknowledgment (TTA), which can be based on average delivery time of envelopes. As long as an envelope is encountered within its TTA, it is simply stored and forwarded. As the TTAs for envelopes at a node starts to expire, they are batched for collective and opportunistic ACK requests from their destinations. An ACK for an envelope consists of its unique ID, 16 bytes in size. If an envelope has not been delivered, or an ACK is not possible due to network failures, the TTA is extended up to a limit. While this approach allows some envelopes to continue propagating for some time even after delivery, it trades network usage for local storage—an advantageous exchange in most developing-world environments.

Some nodes in the hybrid network have better connectivity than others. When good connectivity is available, it is used for establishing route shortcuts through *data staging*, and for last-mile delivery of envelopes to their destination. If an envelope makes it to an intermediate node that is well connected to the final destination, the network, rather than mobility, is used to deliver the envelope.

Data staging positions data envelopes at other, well-connected nodes, in order to gain from expected principal mobility in the system. As a simple example, imagine two nodes A and B that are well connected to each other, and a third node C that is in a challenged network environment but 'close' to B due to principal mobility. Now, imagine there was a data envelope originating from A and addressed to node C. In this case, it would be beneficial to use the well connected network to stage the envelope from A at B, so that it can take advantage of the likely path to C. Bati accomplishes this using *clusters* of nodes that can communicate efficiently though the network and *hot links*—strong mobility connectedness in the transition model.

3.4 Routing

When a node encounters a new envelope, the available network resources are examined to determine if the envelope can be directly delivered or gainfully staged as described above. All remaining envelopes are routed using the mobility of individuals in the system. Given the information from the node's collective mobility model, and the mobility models available from the principal, a node determines an ordered list of envelopes to be carried by the principal as follows:

Step 1: The nodes a principal is expected to visit before coming back to the current node are classified into groups based on how soon the principal is expected to visit the respective node, as given by its mobility model. Group formation ensures that the further out the estimated time of arrival is, a larger set of nodes will belong in the group.

Step 2: A majority of the data capsule's storage space is provisioned equally among the node groups. This equal division ensures that the envelopes that will most benefit from going to the nodes a principal is going to visit sooner will have more real estate on the capsule than envelopes destined for subsequent nodes. A node might be bumped to the next group if the confidence for its estimated time of arrival is below a threshold.

Step 3: For each group, the node determines, of all the remaining envelopes, those that would most benefit from going to any of the nodes in the group. This means, the destination of the envelope needs to be closer at one of the nodes in the group than at the current node. Within this set, envelopes are ordered based on how close they would get to their destination. Any unused capsule space from each group is appended to the next group's slice.

Step 4: The remaining space in the data capsule is filled with a randomized set from the left over envelopes. This is to account for the uncertainty in the mobility of the principal, as well as in the collective mobility models, and benefit from it.

4. IMPLEMENTATION

Bati is implemented as a cross-platform user level library with multiple components. Bati was built with easy deployment in mind, and runs on an unmodified network stack. Each node in Bati starts by running the daemon process `noded`, which in turn coordinates with the rest of system. The major components in our system are the mobility handler, network manager, data manager, principals handler, contact handler, and the system buffer manager.

The **mobility handler** deals with the individual and collective mobility models of the system. By interacting with the principal handler when visitors arrive at a node, it learns the mobility patterns of individuals, and updates the internal models accordingly. Combining patterns from all principals that visit the node, this component helps the data manager in routing decisions.

The **network manager** actively monitors and measures networks for the efficient use of available resources. It has three main components that handle network monitoring, ACKing and data staging. This component also assists the data manager by providing the state of network connections available for use in routing envelopes.

The **principal handler** is the user facing part of our system. It provides the interface for dealing with principals as they visit nodes and serves as a gateway to the data manager and mobility handler for principals.

The **data manager** deals with handling and routing envelopes in the system. It receives envelopes either

from the principal handler or the network manager and makes decisions on how to further process them.

The **contact handler** deals with storing metadata about other nodes in the system, and provides a simple lookup service for other components. By keeping information only about relevant nodes and compiling information using an attenuated Bloom filter as described in section 3, the contact handler scales gracefully with the size of the system.

The **system buffer handler** continuously monitors and maintains storage used at nodes. It uses information from other components to make decisions about buffer usage.

4.1 Life cycle of an envelope

Envelopes in our system can be sent to either a node, or a principal which is associated with a home node. Nodes are named using an opaque ID that is unique across the system. We currently use universally unique IDs as described in RFC 4122 [18], but IDs can also be assigned through a central system where a more hierarchical naming scheme could be employed. On the other hand, principal IDs need only be unique across a node. With this structure, an envelope is uniquely addressed as `<principal ID> @ <node ID>`, much like regular email. When new envelopes are inserted in the system, they are compressed and encrypted with the receiver's public key, leaving a small XML description of the envelope. They are then routed to their destination through Bati's hybrid overlay network. Bati nodes communicate with each other using XML-RPC [33].

4.2 The MobLab emulation platform

Many mobility powered systems are evaluated using network simulation platforms such as NS2, or more specific event simulators [3, 4, 8, 17]. While this is a valid way to test out ideas, and get some numbers, it has the undesired property that the evaluated system has to be written towards a specific simulator, rather than geared towards deployment in the wild. Since Bati was built as a standalone system for deployment, we needed an evaluation platform that can run unmodified production code. To this end, we developed MobLab, a mobility emulation platform built on EmuLab, the network emulation testbed from the University of Utah [1]. The goal for MobLab is to enable developers write readily deployable systems, by providing a simple way to evaluate standalone mobility powered systems.

A MobLab emulation session is configured with a model for principal mobility, the number and connectivity of nodes, as well as the application to be run. The mobility of principals is coordinated through an emulation tracker that is well connected to all nodes in the system. When an emulation session is finished, MobLab collects state information from all nodes in the system, and stores it at the tracker for processing. We evaluated Bati extensively using MobLab running on EmuLab.

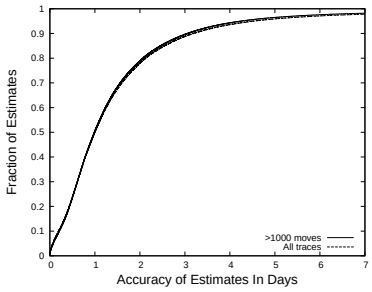


Figure 1: Accuracy of Mobility Models

5. EVALUATION

5.1 Mobility Models

An important component of Bati in making routing decisions is understanding and modeling the mobility patterns of individuals in the system. Bati's mobility model is based on abstractions from a study that tracked more than 100,000 users for six months using information provided by a European cell phone carrier. To evaluate how well these abstractions apply to a different, blind dataset, we use the CROWDAD mobility data from Dartmouth College [14]. This dataset has been extensively used to gauge a number of mobility models [24, 31]. The data was collected nearly continuously for two years, and has a record of more than 6000 traced users as they move about more than 500 nodes. The entire dataset contains close to 9 million 'visits' recorded as `(timestamp, node)` for each user.

Our experiment was set up as follows. Every time an individual visits a node, the prediction made by Bati's mobility model as to when the principal is expected to visit the node is compared against the ground truth. In each comparison, we record how much the estimated arrival time deviates from the actual arrival time. Figure 1 summarizes the results for traces with more than 1000 moves (the only subset used in [24, 31]) as well as the entire dataset. On the X axis, we give the accuracy of the estimates for the arrival time, measured in terms of the deviation from the actual result, and on the Y axis is the fraction of estimates. In either case, the model was more than 50% accurate within a day, more than 80% accurate within three days and more than 97% accurate within a week – enabling Bati to make informed routing decisions for delay tolerant data.

5.2 Envelope Delivery

In evaluating Bati for envelope delivery, we needed to pick a number of parameters for the system. We have based our assumptions on pessimistic expectations of the environment for Bati's deployment:

Principal and node buffer: Storage is increasingly getting cheaper, with less than 10 cents per gigabyte [23]. In stark contrast to that, we limit the amount of principal buffer in our system to 100MB for the base run, and experiment with values up to 400MB. Node

storage is limited to 3GB. Anecdotally, the average free disk space at a local partner's internet cafe in Addis Ababa was around 22GB.

Envelope size: Network is much more constrained than storage in developing counties. As a result, the bigger the message size, the better Bati fares compared to the current state of affairs due to its ability to leverage storage as well as network connections. We have thus limited our envelopes to a modest 1-5MB range. This could represent an eBook, an MP3 file etc.

Network conditions at nodes: Based on experiences using internet kiosks in Developing countries [25, 26], we set poorly connected nodes in our system to have a 15Kbps bandwidth and 300ms latency. We use Akamai's State of the Internet report [27] on average international bandwidth to set well connected nodes with 1.5Mbps bandwidth and 60ms latency. We experiment with different penetration rates of well connected nodes in our system at 0%, 10% and 20%.

ACK intervals: As discussed in section 3, Bati uses an asynchronous, opportunistic and collective ACKing strategy. The ACK interval at a node is basically a tradeoff between bandwidth and storage. One extreme is to make it infinite (mimicking the behavior of non-ACKing ad-hoc systems) and kick out envelopes when the buffer is full regardless of delivery. The other extreme is TCP style synchronous ACKing where envelopes are almost immediately ACKed. Bati employs a strategy more suited to challenged environments where envelopes are ACKed collectively and opportunistically at higher time intervals. We experiment with a number of these intervals in our runs and report on them.

5.3 Experimental setup and analysis

We evaluate Bati under two scenarios. The first scenarios uses a mobility model abstracted from the mobility study [11]. We use this setup to evaluate a number of different characteristics of the system. We then run our system using mobility traces from the CROWDAD dataset. Both scenarios were constructed using MobLab running on the EmuLab platform.

5.3.1 Using an individual mobility model

The first scenario is based on the mobility model abstracted from the large scale individual mobility study. Mobility of individuals is based on the observation that the probability of finding a user at a location with a given rank L, where L is based on the frequency of visits to the node, is well approximated by $P(L) \sim \frac{1}{L}$, and that people spend a significant majority of their time in 5 or fewer places. For a given number of nodes, this comes out to a multiple of a harmonic series which can be solved for individual probabilities of principals visiting the different nodes in the ranking. This distribution is supplied to MobLab and guides how principals move in the system.

In addition, based on observation of what real life nodes could look like, we identify two kinds of nodes: private nodes and public nodes. Compared to public nodes (such as internet kiosks, university computer labs), private nodes (such as homes, offices) are not visited by as many people. However, private nodes are the top ranked destinations for the principals who visit them, while public nodes mostly aren't. In our experiments, each private node has two principals that have it as their top ranked destination, and these principals visit four other nodes from the public nodes, picked and ranked randomly. Principals move among the nodes using a distribution determined from the above harmonic series calculation. There are twice as many public nodes as private nodes in the experiment, and a modest total of 21 nodes. With EmuLab's traffic shaping nodes, this translates to a little under 50 PCs, which was on the higher end of what we could consistently reserve in the few months we tested the system. A challenge with building real systems is the need for physical resources during evaluation.

An individual experiment proceeds as follows. The system is started and allowed to run for 500 seconds. After that, for the next 1000 seconds, each principal sends two messages to randomly selected nodes in the system every 10 seconds from wherever it is currently located, as directed by the mobility model. The system is then allowed to run for another 1000 seconds before it is terminated and information is collected. Each set is run four times, and the collective results are reported. The data is analyzed in a number of different ways, including delivery rates, buffer space used at nodes and the number of network and mobility hops.

5.3.2 Base run

For comparison against different configurations, we establish a base run with parameters selected from section 5.2. For this run, we have 1MB messages sent, with the principal buffer left at 100MB. We have a network penetration of 20%, and a collective ACKing interval of 400 seconds. Figure 2 exhibits the analysis for the base run. The CDF shows more than 50% the enevelopes were delivered within 500 seconds, and more than 90% with in 1000 seconds. Figure 2(b) shows the average delivery time for envelopes sent earlier in the run is shorter than those later, and plateaus to around 500 seconds for later envelopes. This is because principal buffer space is more plentiful earlier in the run. Figure 2(c) shows the average buffer used at nodes, taken every 100 seconds at all nodes. Buffer used at the nodes also plateaus after about mid way through the run, and declines afterwards. As envelopes get delivered in the system and are ACKed, they are kicked out of non-destination nodes, which reduces the buffer footprint. The number of network hops made by envelopes is proportional to the availability of good connectivity while mobility hops are determined by the individual movement patterns and available principal buffer space.

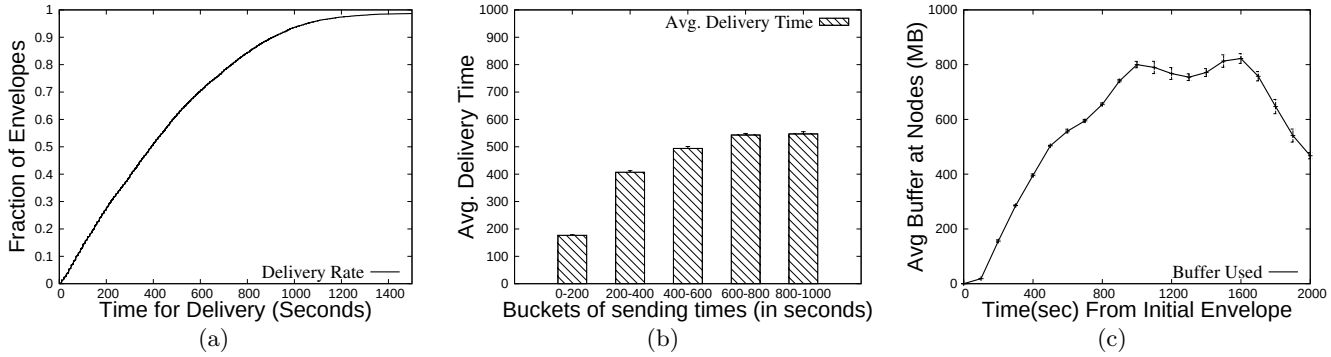


Figure 2: Base run

5.4 System comparisons

We begin our evaluation by comparing Bati's performance to current alternatives for moving bulk data in challenged networks. These include mobility based ad-hoc networks and delay tolerant use of all available network links.

To draw some comparison with mobility-powered ad-hoc networks, we implemented the highly popular Probabilistic Routing Protocol using the draft RFC with the IETF's DTN Research Group [20]. The implementation was done in the context of a standalone application that was evaluated using the MobLab framework under similar conditions as the base run. To establish a baseline, we also implemented and evaluated an epidemic routing ad-hoc network. Bati improves the delivery rate of envelopes by more than 40% for probabilistic routing, and even higher for an epidemic routing network.

We then compare Bati to using the available network links efficiently, with in-network storage for delay tolerance. Considering the amount of bytes transferred from delivering the envelopes in 2000 seconds of the base run with a 1MB envelope size, it takes the average node upwards of 15 hours to accomplish the same using the network. A weakly connected node takes slightly over 20 hours to deliver envelopes sent from it during the run. The savings with using Bati are even higher (nearly two orders of magnitude) for a 5MB envelope run, where Bati's ability to leverage unstructured mobility serves it well. Figure 3 summarizes the results.

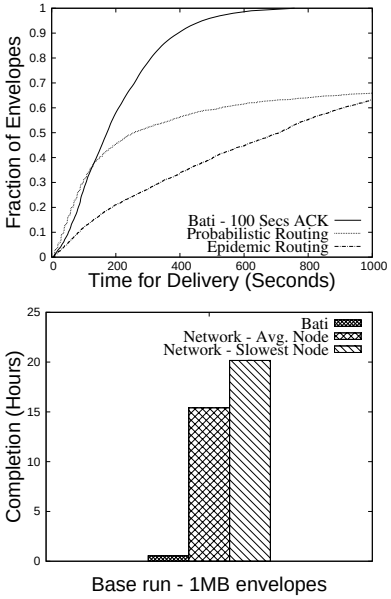


Figure 3: System comparisons

5.5 System characterizations

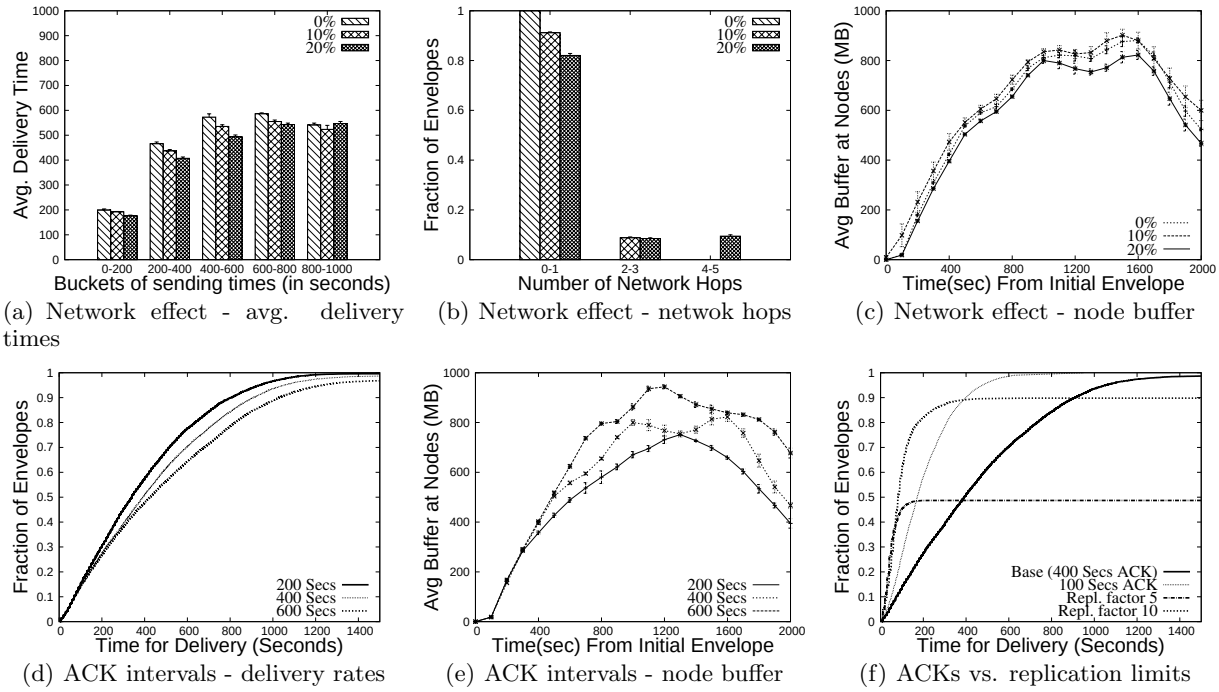
In this section, we will look at different factors that affect Bati's performance. In the interest of space, we have left out our experiments on the effect of principal buffer space, and will highlight only two to three results from each run. The full analysis will be available as a tech report.

5.5.1 The effect of network penetration

In order to gauge the effect of good network penetration in the system, we vary the percentage of nodes that are well connected in our system from 0% to 20%, while keeping everything else similar. The number of mobility hops decreases and the number of network hops increases as the penetration rate increases, shown in figure 4(b). This is because more envelopes are able to use the network rather than mobility for delivery. Buffer space used, shown in 4(c) also slightly reduces as connectivity increases because more envelopes are delivered using the network, which takes a shorter time and allows nodes to kick out envelopes sooner. This also reduces the average delivery times of envelopes as shown in figure 4(a).

5.5.2 The effect of ACKing intervals

Another factor we consider in our evaluation is different ACKing intervals. Leaving everything else similar in our base run, we experiment with 200 second, 400 second and 600 second ACKing intervals. As shown in figure 4(e), the most significant tradeoff between these runs is buffer space at nodes. The more frequent ACKs are, the less space that is used at nodes. In a similar fashion, less and less delivered envelopes are propagated



(a) Network effect - avg. delivery times (b) Network effect - netwok hops (c) Network effect - node buffer

(d) ACK intervals - delivery rates (e) ACK intervals - node buffer (f) ACKs vs. replication limits

Figure 4: Effects of Network penetration and ACK intervals

through Bati with the increase in frequency of ACKs. This is reflected in the slightly higher delivery rate for the 200 second run over the others as seen in figure 4(d). The number of mobility hops also slightly increases with increased ACKing intervals. The number of network hops, however, is not affected very much because network delivered envelopes do not need a separate ACK anyway. The small increase in network hop count with higher ACKing intervals is due to envelopes that are delivered through mobility propagation to nodes with good network connections, which end up staging or delivering the envelopes through the network.

As described earlier, one of the design principles in Bati is to use collective ACKs and likelihood of delivery for buffer space management rather than a strict limit on hop-counts or replicas. Although this approach results in envelopes propagating for some time even after delivery, thereby using some buffer space, it delivers more envelopes in the long run. To illustrate this point, we ran two simple variation of Bati where the number of replicates propagated by a node is initially set to 5 and 10. These approaches in fact have a quicker start in delivering messages, but also introduce an asymptote in the delivery rate. This asymptote depends on the size of the system, and how far envelopes have to travel to their destination. As a result, we opt for the more graceful scheme in Bati. On the other hand, with a 100 second ACKing interval, Bati achieves the quick start property, while still maintaining a high delivery rate. On average, the size of a collective ACK was 9.7KB for the 400 second interval and 3.7KB for the 100 second interval—representing 1.3% and 1.8% of a weakly connected node's available bandwidth respectively. Figure 4(f) compares the results.

5.6 Using recorded traces

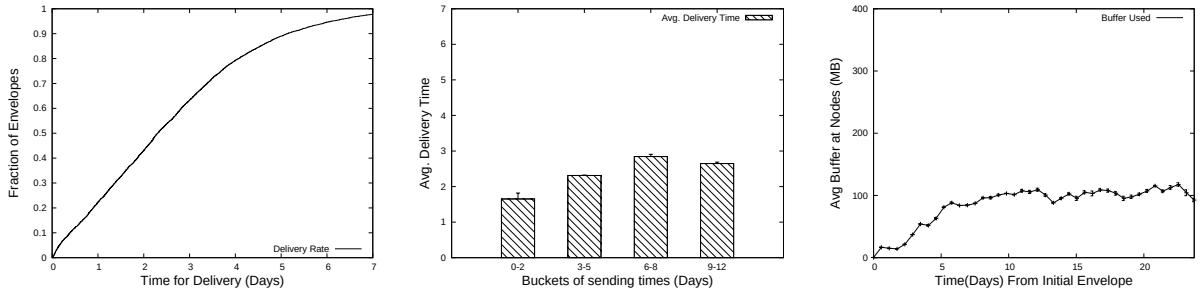
After evaluating various interesting characteristics of our system using Scenario 1, we were curious as how our system would operate on real world mobility traces. Ideally, the two should be equivalent as the mobility model we used is an abstraction of real world mobility.

To test the hypothesis, we again use the mobility data from Dartmouth College. We consider the data collected from February of 2003, the last full month available in the dataset, because it potentially has the highest coverage of nodes and principals. There are more than 1700 active principals and more than 500 nodes visited during the month.

Our experiment was setup as follows. For each run, we pick 21 nodes randomly and consider the principals that visit these nodes. In order to reduce the resource consumptions of our emulation, we make two simplifying reductions. We emulate only:

1. The top 30 mobile principals visiting the nodes
2. The first 500 steps of these principals

Both reductions work against our system because the more principals we emulate, and the more they move around, the more chances there are for data transfer. The long tail of unemulated principals could have served as additional carriers in the network. We divide the warm up, message sending and running portions of this experiment in a similar fashion as the first scenario. As such, the system is initially run for a week, then for the



(a) (b) (c)

Figure 5: Trace Run

next 12 days, every time a principal visits a new node, it sends two messages to randomly selected nodes. After that, the system is allowed to run for the rest of the trace. We run the experiment four times.

The results we obtain are fairly equivalent to those in scenario 1, as shown in figure 5. More than 50% of the envelopes were delivered within 3 days, and more than 90% within a week. The average delivery time was between 2 and 3 days for the various clusters of sending times. Buffer and hop counts also have a predictable behavior as shown in figure 5(c).

6. CONCLUSION

This paper described phoretic networking, a hybrid overlay network for bulk data in the challenged environments endemic to developing regions. It uses estimates of natural mobility along with available network resources to transfer data. Our Implementation, Bati, is easy to deploy as an additional, standalone service on an unmodified network stack, and was evaluated using a mobility emulation platform, MobLab. Our results with conservative estimates of available resources show substantial savings in using Bati for delivering bulk data compared to the network, while improving delivery rates compared to solutions that use only ad hoc routing between mobile peers.

We intend to extend this work in a few ways. First, we would like to address incentives, since the value of networked systems increases as a function of participation. An economic model that uses micropayment systems is one such alternative. In addition, we would like to facilitate fair use of resources by all participants. We plan to address this using built in mechanisms that regulate the value derived by principals from the system to be proportional to their contribution.

7. REFERENCES

[1] Emulab network emulation testbed. http://emulab.net.

[2] N. Azzouna and F. Guillemin. Analysis of ADSL traffic on an IP backbone link. In *Global Telecommunications Conference,IEEE*, pages 3742–3746 vol.7, Dec. 2003.

[3] A. Balasubramanian, B. Levine, and A. Venkataramani. DTN routing as a resource allocation problem. *SIGCOMM Comput. Commun. Rev.*, 37(4):373–384, 2007.

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

[5] X. Chen and A. Murphy. Enabling disconnected transitive communication in mobile ad hoc networks. In *POMC 2001*, pages 21–27, Newport, RI, USA.

[6] C. Cheng, R. Jain, and E. van den Berg. Location prediction algorithms for mobile wireless systems. 2003.

[7] M. Demmer and K. Fall. DTLSR: delay tolerant routing for developing regions. In *NSDR '07*, New York, NY. ACM.

[8] K. Fall. A delay-tolerant network architecture for challenged internets. In *SIGCOMM '03*. ACM.

[9] J. Gantz. The diverse and exploding digital universe. Technical Report White paper, IDC, 2008.

[10] N. Glance, D. Snowdon, and J.-L. Meunier. Pollen: using people as a communication medium. *Comput. Netw.*, 35(4):429–442, 2001.

[11] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, June 2008.

[12] S. Guo, M. H. Falaki, E. A. Oliver, S. U. Rahman, A. Seth, M. A. Zaharia, U. Ismail, and S. Keshav. Design and implementation of the kiosknet system. In *ICTD '07*.

[13] Haggle Project. `www.haggleproject.org`.

[14] T. Henderson, D. Kotz, I. Abyzov, and J. Yeo. CRAWDAD trace set movement. http://crawdad.cs.dartmouth.edu.

[15] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. In *SIGCOMM '04*, Portland, Oregon. ACM.

[16] M. Jensen. Open Access: Lowering the costs of international bandwidth in Africa. *APC Issue Papers*, 2006.

[17] E. P. C. Jones, L. Li, and J. K. Schmidtke. Practical routing in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 6(8):943–959, 2007.

[18] P. Leach, M. Mealling, and R. Salz. Rfc 4122: A universally unique identifier (uuid) urn namespace, 2005.

[19] J. Leguay, T. Friedman, and V. Conan. Evaluating mobility pattern space routing for DTNs. In *INFOCOM*, 2006.

[20] A. Lindgren and A. Doria. Probabilistic routing protocol for intermittently connected networks. http://tools.ietf.org/html/draft-irtf-dtnrg-prophet-00, 2008.

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

[22] K. W. Matthee, G. Mweemba, A. V. Pais, G. van Stam, and M. Rijken. Bringing Internet connectivity to rural Zambia using a collaborative approach. In *ICTD '07*.

[23] Newegg, 2009. `www.newegg.com`.

[24] A. J. Nicholson and B. D. Noble. Breadcrumbs: forecasting mobile connectivity. In *MobiCom*, 2008.

[25] B. Petrazzini and M. Kibati. The Internet in developing countries. *Commun. ACM*, 42(6):31–36, 1999.

[26] A. Reda, B. Noble, and Y. Haile. Distributing private data in challenged network environments. In *WWW 2010*.

[27] A. Reports. The State of the Internet, 2nd quarter, 2009. `www.akamai.com/stateoftheinternet/`.

[28] S. Rhea and J. Kubiatowicz. Probabilistic Location and Routing. In *INFOCOM'02*, volume 3, pages 1248–57, 2002.

[29] A. Seth, D. Kroeker, M. Zaharia, S. Guo, and S. Keshav.

Low-cost communication for rural internet kiosks using mechanical backhaul. In *MobiCom 2006*.
[30] C. Song, Z. Qu, N. Blumm, and A.-L. Barabsi. Limits of predictability in human mobility. *Science*, 327(5968), 2010.
[31] L. Song, D. Kotz, R. Jain, and X. He. Evaluating location predictors with extensive wi-fi mobility data. In *INFOCOM'04*.
[32] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks, 2000.
[33] D. Winer. XML-RPC Specification. `www.xmlrpc.com/spec`.