

A Low-energy, Multi-copy Inter-contact Routing Protocol for Disaster Response Networks

Md Yusuf Sarwar Uddin, Hossein Ahmadi, Tarek Abdelzaher, Robin Kravets

Department of Computer Science

University of Illinois at Urbana-Champaign

{mduddin2, hahmadi2, zaher}@uiuc.edu, rhk@cs.uiuc.edu

Abstract—This paper presents a novel multi-copy routing protocol for disruption-tolerant networks whose objective is to minimize energy expended on communication. The protocol is designed for disaster-response applications, where power and infrastructure resources are disrupted. Unlike other delay-tolerant networks, energy is a vital resource in post-disaster scenarios to ensure availability of (disruption-tolerant) communication until infrastructure is restored. Our approach exploits naturally recurrent mobility and contact patterns in the network, formed by rescue workers, volunteers, survivors, and their (possibly stranded) vehicles to reduce the number of message copies needed to attain an adequate delivery ratio in the face of disconnection and intermittent connectivity. A new notion of *inter-contact routing* is proposed that allows estimating route delays and delivery probabilities, identifying more reliable routes and controlling message replication and forwarding accordingly. We simulate the scheme using a mobility model that reflects recurrence inspired by disaster scenarios, and compare our results to previous DTN routing techniques. The evaluation shows that the new approach reduces the resource overhead per message over previous approaches while maintaining a comparable delivery ratio at the expense of a small (bounded) increase in latency¹.

I. INTRODUCTION

In this paper, we design an energy-efficient multi-copy routing protocol for disaster-response networks that exploits recurrence to achieve improved resource economy. In the aftermath of a natural disaster (such as a hurricane or a strong earthquake), regular communication services are disrupted due to infrastructure damage and power outages. Communication remains essential for conducting a well-managed recovery operation, finding survivors or contacting friends and loved ones. In the absence of power, such communication must rely on battery-operated wireless devices such as cell-phones in an ad hoc communication mode and wireless routers powered by vehicular batteries (e.g., some car manufacturers are already announcing car models that come with routers and wireless network access [1]). Since such devices are not likely to form a single connected network at any given time, disaster response networks must be disruption-tolerant (DTN).

A key performance objective of disaster-response networks is to minimize energy consumption per message in order to prolong the lifetime of battery-operated devices until infrastructure is restored. There are many ways energy-economy

can be achieved. For example, nodes can be duty-cycled, low-power listening modes can be used, and transmission energy can be modulated depending on channel conditions and distance from receiver. In this paper, we focus only on the routing layer. Since the main “knobs” under the control of multi-copy routing are the number of message copies made and the path of each message, we consider the total number of message replica transmissions in the network as a quantification of energy-efficiency of *routing decisions*. In that sense, routing protocols proposed for general DTNs are often not suitable for disaster-response networks, as they concern themselves more with other performance objectives. For example, they often achieve a high delivery ratio at the expense of generous message copying and forwarding which consumes significant amounts of communication energy.

In contrast, our protocol significantly reduces the need for large numbers of copies by discovering recurrent contacts and forming a network view (a routing table) that uses such recurrent contacts to deliver a message (possibly across multiple hops) to the destination. By exploiting recurrence to increase delivery probability, the degree of message replication is reduced and the number of message transmissions drops. Hence, the protocol contributes to saving energy while maintaining a high delivery ratio.

Recurrence exists in disaster-response networks because movement of entities in disaster scenes is not entirely random. Instead, some regularity is observed: medical supplies are delivered to evacuation camps, police vehicles patrol given routes, fire trucks originate at fire stations, and volunteers end up at the same gathering points, such as schools used for relief operation management. Furthermore, a number of static points exist that can be used for message handover. For example, the evacuation camps, schools, fire stations, and vehicles stranded on common routes may all serve as stepping stones to deliver data from one moving entity to the next. Hence, a core of moving entities exists that revisits overlapping sets of static places repeatedly. Our routing protocol takes advantage of this core to build a stable routing “backbone” by observing contact occurrences of nodes over time and identifying recurrent contacts.

The protocol uses the new notion of *inter-contact routing* to uncover patterns that reduce the DTN to a multihop data-muling network. Inter-contact routing allows more accurate estimation of delivery delay and delivery probability, identifies

¹This work was funded in part by NSF grants CNS 06-26825, CNS 06-26342 and CNS 05-54759.

reliable multi-mule paths when they exist, and informs message replication and forwarding decisions. Since each relief operation is unique, the identities of data mules and their sources and destinations are learned rather than hardcoded into the protocol. Evaluation shows that the protocol is effective at reducing the total number of message transmissions in the network while maintaining a high delivery ratio and only a slightly larger delay compared to other DTN protocols in recent literature.

The rest of this paper is organized as follows. In Section II we present some existing literature on DTNs. In Section III we elaborate our proposed routing protocol. Section IV presents a disaster-centric mobility model, followed by Section V that shows evaluation results from simulation. We conclude with some future research directions on routing in disaster response networks in Section VI.

II. RELATED WORK

DTNs have a wide range of applications, which vary from acoustic underwater networks [2] to networks of vehicles [3]. While there has been no explicit DTN model for disaster scenarios, there exist several general architectures and models for DTNs in past literature [4], [5]. Most noticeably, there has been a significant amount of work on routing in disruption tolerant networks [6], [7], [8], [9], [4], [10], [11], [12].

An important category of DTN routing schemes relies on devoting specific nodes (data mules), whose mobility is controlled, to deliver messages among others. Message Ferrying [13], [14] is an example of such protocols, designed for systems with predictable mobility patterns. Other approaches also take advantage of mobile nodes as intermediate ferries, where these nodes can change their trajectories based on requests they receive [11].

A different approach to DTN routing is one where node mobility is not under the control of the routing algorithm. Epidemic routing [6] is the most basic of these methods that replicates messages whenever a node meets another node which has not received the same message so far. Epidemic dissemination gives the best delivery ratio and delay if storage and transfer bandwidth are not limited. SWIM (shared wireless infostation model) [15] introduces info-stations as ‘healing’ centers where a ‘diseased’ node dispatches its messages and attains ‘immunity’ from future ‘infection’. A more recent work [16] presents the resource and performance tradeoffs of deployment of these infostations. Some research uses network coding [17], [18] on top of replication to achieve better delivery in the case of high failure rates. There are a few variants of epidemic algorithms that gather information from the network, with particular emphasis on encounter patterns. PROPHET [19] estimates delivery predictability to destinations using the history of encounters and the transitive property of meetings with nodes. MaxProp [8] computes delivery probabilities from meeting frequencies and sorts messages in the transmission buffer accordingly. Despite the good delivery ratio, epidemic routing and its variants incur

very high communication (and, hence, energy) overhead and are thus inappropriate for disaster-response networks.

There exist schemes where the number of times a message can be replicated is pre-specified. Example includes Spray and Wait [7] that limits the total number of copies created initially (spray phase). Replicas then wait for a direct delivery opportunity to the destination (wait phase). A variant of this algorithm, called Spray and Focus [20], introduces utility-based forwarding of the last copy instead of just waiting to meet the destination. Although Spray reduces overhead, it does not assume regularity in mobility patterns and contacts. Hence, it achieves generality at the expense of losing opportunities for further optimization.

Yet another approach to DTN routing is to use prior information about network connectivity. This information enables the protocol to achieve very high delivery rates at low overhead. A framework is described in past literature [4] that formulates the DTN routing problem given different amounts of prior information about the network. MEED (minimum estimated expected delay) [9] estimates expected delay for each specific contact during network operation. Using this delay as a link metric, routing is done in a link-state fashion by propagating each individual link delay. In [10], scalable routing is investigated in deterministic DTNs where mobility follows strictly periodic patterns. They propose a DTN Hierarchical Routing (DHR) scheme that yields the optimal time-space algorithm in terms of delay and hop-count. RAPID [12] models DTN routing as a utility-driven resource allocation problem and computes delay-based utility values of messages assuming exponential distribution of inter-node meeting times.

There are forwarding schemes [21], [22], [23] that rely on comparisons between per-node metrics to make forwarding decisions. They intend to reduce communication overhead, while achieving a level of performance close to that of epidemic algorithms. Spyropoulos et al. [24] present an analytical framework for the forwarding-only case (allowing only one message copy) on a grid-based mobility model. FRESH [23] relies on a node’s last meeting time with the destination to make a forwarding decision. Greedy [21] relies on contact rate with the destination and greedy-total uses the total contact rate of a node. Delegation forwarding [25] proposes a metric-based general forwarding scheme; it assumes that each node has an associated ‘quality’ metric and a node forwards a message if it encounters another node whose quality metric is greater than any seen by the message so far. Analysis tried commonly-cited metrics, such as frequency and last-contact time. Our approach incorporates *inter-contact delay* as a new metric in making decisions.

III. LOW-ENERGY INTER-CONTACT ROUTING

We describe *Inter-contact Routing* (IC-Routing), which improves energy efficiency by forwarding fewer messages when paths have a higher probability of delivery. The protocol exploits information about different paths, when available, to estimate message delivery probability. It then forwards messages on the most reliable paths to eliminate unnecessary

message replication. When no information is available it defaults to a prior state of the art protocol [7]. Therefore, the resulting overall number of message transmissions and energy consumption are decreased, when possible, while keeping the delivery ratio high.

There are two key challenges associated with opportunistic exploitation of paths with a higher delivery probability; namely, (i) what information to maintain in routing tables to estimate delivery probability, and (ii) how to make packet replication and forwarding decisions based on that information (as well as what to do when no information is available to tell how to reach a destination).

To address the first challenge (estimation of delivery probability), we observe that since nodes move in a finite space, under broad mobility assumptions, all packets will be delivered given an infinite amount of time. To obtain a more meaningful metric, we define successful delivery as one that occurs within a given latency bound, called the *delivery timeout*. If delay extends beyond that timeout, delivery is said to have failed. Nodes gather statistics of their contacts with other nodes. In particular, *inter-contact delay* statistics are collected. Flooding these data through the network makes it possible to add up inter-contact delays along paths and compute path delay distributions and hence delivery probability within the timeout.

To address the second challenge (making replication decisions), we set a virtual number of copies for each message that bounds the number of actual copies which can be created for it. A similar idea has been used in Spray-and-Wait [7]. Using delivery probabilities contained in our routing information, we partition the virtual copies, putting more copies on routes with a higher confidence. In the absence of information on a destination, we resort to a blind spray of messages.

In the rest of this section, we shall first detail our disruption-tolerant network model (Section III-A), then elaborate on the aforementioned two challenges (Section III-B).

A. Network Model

Each node in our protocol maintains a set of *neighbors*, which are nodes that it encounters recurrently. Encounters with a node are considered recurrent if they occur more often than a given number of times within a specified time window. Intermediate nodes along a message's path hold the message from an encounter with the previous node to an encounter with the next along the path to the destination.

In a classical network, node delay depends only on the conditions of outgoing links. Packets are buffered until such links become available. In particular, the delay does not depend on which input port a packet came from. DTNs are fundamentally different in that the delay a packet experiences at a node depends not only on the next node the packet is forwarded to, but also on the previous node it came from. It is the delay between the above two contacts (i.e., the inter-contact delay) that determines the packet's local residence time. Our network model is inspired by that observation. Hence, rather than thinking in terms of *encounter graphs*, where each vertex corresponds to a node and edges connect

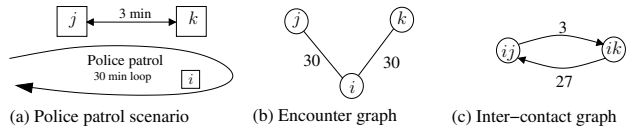


Fig. 1. A scenario showing encounter and inter-contact graph.

nodes that have frequent encounters, we use an *inter-contact graph*, where a vertex represents an encounter between two nodes. An edge between two vertices represents the delay between the (beginnings of) two encounters. Each vertex is labeled with the corresponding node names, such as ij for an encounter between node i and node j . For example, consider a police patrol, i , that makes 30 minute rounds along some city loop. The patrol passes building j then building k at two successive street intersections, approximately 3 minutes apart. Given the above parameters, it will be 27 minutes after seeing k that the patrol sees j again. Hence, a directed edge of delay 3 minutes exists in the inter-contact graph between vertices ij and ik . Similarly, a directed edge of delay 27 minutes exists between vertices ik and ij . This asymmetry explains why edges in the inter-contact graph are directed.

The novelty of inter-contact graphs lies in capturing the dependencies of delay on both the previous and next hops on the path of a message in a DTN. The delay asymmetry mentioned above is not captured in a regular encounter graph that plots the period or frequency of encounters between individual nodes. In such a graph, since node i sees each of j and k every half hour, an edge of delay 30 minutes might exist between i and each of j and k . When i encounters j , it might seem that sending a message from j to k via i would take 60 minutes, whereas in fact it takes only 3 minutes, as captured in our inter-contact graph. Figure 1 illustrates the above mobility scenario together with its representation in an encounter graph and an inter-contact graph.

In the following discussion, to avoid ambiguity, we shall use the term *node* to refer to a physical device capable of receiving and forwarding messages in the network. We shall use the term *vertex* to refer to a vertex in the inter-contact graph that represents an encounter between two nodes. We use the notation, $c_1 \rightarrow c_2$ to denote a directed edge between two contacts (i.e., vertices) c_1 and c_2 . Each edge, $ij \rightarrow ik$, in the inter-contact graph maintained is annotated by a tuple of two values, $(\delta(ij \rightarrow ik), \sigma^2(ij \rightarrow ik))$, where $\delta(ij \rightarrow ik)$ is the average delay elapsed on node i between meeting node j and node k , and $\sigma^2(ij \rightarrow ik)$ is the corresponding delay variance. We denote a path in the inter-contact graph as *contact* \rightsquigarrow *destination*. For example, $ij \rightsquigarrow w$, is a path from contact ij to (some contact with) node w . Due to the linearity of expectation and under the assumption of edge independence, we can define path delay, denoted $d(ij \rightsquigarrow w)$, and path variance, denoted $\sigma^2(ij \rightsquigarrow w)$, as the sums of the edge delays and variances along the path. If there are multiple paths from an initial contact to the destination, the parameters of the 'best' path will be stored for routing purposes. Since, in general, not all nodes have the same view of the network, it is useful to define path delay and variance from the perspective of a given node.

Hence, we define $d_i(ij \rightsquigarrow w)$ and $\sigma_i^2(ij \rightsquigarrow w)$ to denote path delay and variance computed by node i .

B. The Routing Algorithm

As observed above, DTNs are different from regular networks in that message delay at a node depends not only on the node downstream but also on the one upstream. A routing table in a typical (e.g., wired) network has one entry per destination (or destination mask). In our algorithm, the entry for destination w at vertex ij in the inter-contact graph contains the corresponding path delay, $d_i(ij \rightsquigarrow w)$, and variance, $\sigma_i^2(ij \rightsquigarrow w)$. We store both values as a parametric summary of the delay distribution along the optimal route, as seen by node i . The delay distribution determines the message delivery probability (by a given timeout).

A question arises as to why not store the path delivery probability directly in the routing table. The problem lies in that it is message-dependent. For example, if the network timeout is 6 hours, and a message arriving at a node has already been en route for two hours, successful delivery refers to delivering this message in the next four hours. Estimating the probability of successful delivery in four hours requires knowledge of the delay distribution, which we estimate from a mean and variance. Hence, by storing the mean and variance of the delay distribution of a path, we can individually compute the different delivery probabilities for different messages.

A more subtle question lies in the notion of path optimality. How to choose the path whose delay and variance should be stored in the routing table entry for a given destination? The typical answer is, we should store the parameters of the optimal path. However, as we just mentioned, the delivery probability depends on the message, as it depends on how long the message has already spent en route. Hence, an optimal message-parameter-independent path cannot be established. Instead, to determine the path whose parameters to store in the routing table, we come up with an alternative *message-independent* path cost defined as the 95th-percentile of path delay, or $cost = d_i(ij \rightsquigarrow w) + 1.65 \times \sqrt{\sigma_i^2(ij \rightsquigarrow w)}$. This metric favors paths with an appropriately low delay and delay variance, hopefully leading to a higher delivery probability (within the timeout).

Hence, our routing protocol stores the delay and variance of the minimum cost path to each known destination as defined by the above message-independent cost metric. A separate routing table is constructed at each node i for each vertex ij (i.e., for each vertex in the inter-contact graph that refers to an encounter of node i with one of its neighbors, j). The table for vertex ij at node i has an entry per destination w (known to i), storing $d_i(ij \rightsquigarrow w)$ and $\sigma_i^2(ij \rightsquigarrow w)$ of the optimal path cost to that destination, should the message be forwarded via the encountered node, j . Observe that, a vertex ij in the inter-contact graph is therefore associated with two routing tables; one maintained at node i and one at j .

1) *The Inter-contact Delay Table:* In a typical connected network, all outgoing links are available simultaneously. The situation is different in DTNs in that we must account for

inter-contact delays. We use a second table, called the *inter-contact delay table*, to store the average inter-contact delay and delay variance between contacts of the node in question with every two of its neighbors. Hence, this table contains one entry per neighbor pair. For simplicity, we ignore the time it actually takes to forward a message to another node once the contact is made. Typically, such delay is orders of magnitude lower than the inter-contact delay, which may be of the order to tens of minutes or hours. For the same reason, we consider contacts to be points in time (rather than time intervals). If a contact lasts for a long duration, the relevant point in time is the beginning of such interval, as that is when buffered messages are exchanged.

When node i meets a neighbor k , it computes $x(ij \rightarrow ik)$, the delay between contacts ij and ik , for each neighbor j seen more recently than the previous encounter with k . Let the current time be T_{now} , the last encounter time with j be T_i^j and the last encounter time with k be T_i^k . Node i computes the individual sample $x(ij \rightarrow ik)$, as well as the mean and variance as follows:

$$\begin{aligned} x(ij \rightarrow ik) &= T_{now} - T_i^j, \text{ if } T_i^k < T_i^j \\ \delta(ij \rightarrow ik) &= \bar{x}(ij \rightarrow ik) \\ \sigma^2(ij \rightarrow ik) &= S_n^2 = \frac{\sum (x - \bar{x})^2}{n-1} \end{aligned}$$

where n is the number of samples in the summation and S_n is the standard deviation of these samples.

2) *Routing Table Updates:* Routing table updates occur in a distance-vector manner. When two nodes i and j meet, they update each other's routing tables for vertex ij . Consider node j that meets node i . Node j recomputes its optimal paths to each of the destinations known to j and shares the result with i to store in i 's routing table for vertex ij (which describes the path costs via j). The path delay and variance from vertex ij to each destination w is computed by considering the set of j 's neighbors, N_j . For every neighbor $l \in N_j$, the mean delay, variance and corresponding cost are first computed. The optimal cost is then found:

$$\begin{aligned} delay_l &= \delta(ji \rightarrow jl) + d_j(jl \rightsquigarrow w) \\ var_l &= \sigma^2(ji \rightarrow jl) + \sigma_j^2(jl \rightsquigarrow w) \\ cost_l &= delay_l + 1.65 \times \sqrt{var_l} \\ l^* &= \arg \min_{l \in N_j, l \neq i} cost_l \end{aligned}$$

Node j then sends $delay_{l^*}$ and var_{l^*} to i as the mean delay and variance of the optimal path to destination w via j (for each destination w). The routing table for vertex ij at node i is updated. Similarly, node i updates the routing table for node j . In principle, the above computation can be done in advance of the actual encounter. We use a "lazy" approach, since we might not encounter a node for a long time (and need not precompute its table entries), and because the inter-contact delays (used in the above computation) are updated upon each encounter, making previous results stale anyway.

The above parameters are then used at node i to compute the *message-dependent* delivery probabilities that decide if any of its messages are to be forwarded to j , and how many copies should be forwarded, as described below.

3) *Forwarding and Message Replication*: Every message in our network starts with a certain number of *initial message copies* from its source and an original *timeout* value, which defines a time-frame for successful delivery. A remaining time-to-live (TTL) field is initialized to the original timeout. When a node i that has copies of a message meets the destination, the message is delivered. When the node meets one other than the destination, it may forward some copies of the message to the encountered node. Obviously, the same message is not sent repeatedly, but rather sent once. The intended number of copies it represents is carried in its header. Suppose, node i has a message with L copies to deliver to w , and it meets its neighbor j . The node updates the TTL field of the message (by subtracting the time since the last update). It then computes the delivery probability p_k via each one of its neighbors, k (including $k = j$), as follows:

$$p_k = P\{0 < \text{delay} \leq \text{TTL} | \text{delay} > 0\} \quad (1)$$

$$= \frac{\Phi\left(\frac{\text{TTL} - \text{delay}_k}{\sqrt{\text{var}_k}}\right) - \Phi\left(\frac{-\text{delay}_k}{\sqrt{\text{var}_k}}\right)}{1 - \Phi\left(\frac{-\text{delay}_k}{\sqrt{\text{var}_k}}\right)} \quad (2)$$

where $\Phi(\cdot)$ is the CDF of normal distribution, and:

$$\begin{aligned} \text{delay}_k &= \delta(ij \rightarrow ik) + d_i(ik \rightsquigarrow w) \\ \text{var}_k &= \sigma^2(ij \rightarrow ik) + \sigma_i^2(ik \rightsquigarrow w) \end{aligned}$$

Neighbors are then logically ordered by p_k in descending order. Each neighbor k , in that order, is allotted $p_k L$ copies and these allotments are subtracted from L . The process continues until L runs out or until all neighbors have been considered. If all neighbors have been considered yet some copies remain, it must be that the probabilities p_k were low. The remaining copies are sprayed, which means half of them are given to the encountered node, j . Hence, there are three possibilities with regard to node j :

- L runs out before we disburse j 's allotment. This means there are sufficiently better nodes k ahead of j in the ordered list by p_k . No copies are forwarded to j .
- L runs out when or after we disburse j 's allotment. In this case, j gets $p_j L$ copies (or the remainder of L if less than $p_j L$)
- L does not run out even after all neighbors are considered. In this case, j gets $p_j L$ copies plus half the remainder of L .

When the message is forwarded, its TTL is current. The receiver notes the time it arrived and uses that timestamp to update the TTL of the message later by subtracting local residence time. The above forwarding algorithm has some interesting properties:

- If $p_j = 1$: The encountered node j takes all copies of the message. The case $p_j = 1$ can happen only if the variance along the path is extremely low indicating a very stable path. In that case, this path alone carries all copies of the message and no other paths are tried. No physical message replication occurs.
- All $p_k = 0$: This is the case by default in the beginning when no information is known about any destination, or in the case when no recurrence has been observed.

In this case, our protocols defaults to the Spray-and-Wait protocol [7], allowing node i to forward half of the message copies to each encountered node j .

- $0 < p_k < 1$: In this case, the protocol opportunistically exploits paths to the degree to which they are recurrent, hence reducing the number of copies needed compare to Spray-and-Wait, and achieving better resource economy commensurately with the degree of recurrence in the network.

4) *Pruning Low-confidence Links and Bad Neighbors*: It remains to show how a node decides on which neighbors and links to consider in routing decisions. When only a very small number of encounters were observed with others, the confidence in the computed mean delay and delay variance could be very low. Such low-confidence links should not be part of the inter-contact graph. They are identified and eliminated. Specifically, let the real mean of the inter-contact delay of a link be denoted by μ , where $\mu = E[\delta(ij \rightarrow ik)]$. Under fairly broad assumptions, we can determine how close the computed $\delta(ij \rightarrow ik)$ is to the real mean μ by applying the central limit theorem:

$$P\left\{\mu \in \left[\bar{x} \pm Z_\alpha \frac{\sigma}{\sqrt{n}}\right]\right\} \approx 1 - 2\alpha$$

where Z_α is the critical value such that $\Phi(Z_\alpha) = 1 - \alpha$, $\Phi(Z)$ is the CDF of standard normal distribution, defined as $\Phi(Z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^Z \exp(-\frac{t^2}{2}) dt$. We can now compute a confidence value $\gamma(ij \rightarrow ik)$, representing the probability that the estimated average delay is within some bound τ from the true mean by setting $\tau = Z_\alpha \frac{\sigma}{\sqrt{n}}$ above. Hence, $Z_\alpha = \tau \frac{\sqrt{n}}{\sigma}$. Therefore:

$$\begin{aligned} \gamma(ij \rightarrow ik) &= P\{\mu \in [\bar{x} \pm \tau]\} \\ &= 1 - 2\alpha \\ &= 2\Phi\left(\tau \frac{\sqrt{n}}{\sigma}\right) - 1 \end{aligned}$$

Hence, given the interval τ , we can use the above equation to find the probability that the measured sample mean is within that interval from the true process mean. Since ultimately, we are not interested in accurate delay measurements, but rather in rough assurances of bounded delivery, we allow for a considerably large interval (nearly the same width as the sample mean). We use $\tau = 30$ minutes and the constraint $\gamma > 0.75$. Otherwise the link is dropped as untrusted.

Observe that since link delay and variance are updated only on upon encounters with a node, a neighbor that disappears leaves state that may become stale. Hence, if the last encounter with a neighbor becomes "too old", confidence in inter-contact delays involving that neighbors is exponentially decreased. The corresponding links are eventually dropped per rules above. A node with no links is dropped from the tables.

5) *A Numeric Routing Example*: Let us consider an example to illustrate the routing protocol. Figure 2 shows the inter-contact table of node i and the routing entry for destination w via its three neighbors j , k and m (entry contains (μ, σ^2) of the associated delay in minutes). Let node i have

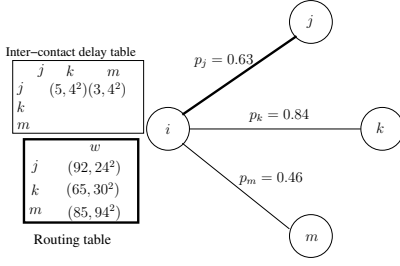


Fig. 2. Node i is deciding message copies for j .

a message with 20 copies to deliver to w within the delivery timeout of 100 minutes. Node i meets j . We are to compute delivery probabilities p_j , p_k and p_m . Path delay and variance via j are given in the routing table, $(d_j, \sigma_j^2) = (92, 24^2)$. Similarly, $(d_k, \sigma_k^2) = (5 + 65, 4^2 + 30^2) = (70, 30.26^2)$ and $(d_m, \sigma_m^2) = (3 + 85, 4^2 + 94^2) = (88, 94.08^2)$ are obtained by adding the inter-contact delay with the routing table entry. By using Equation 2, $p_j = \frac{\Phi(\frac{100-92}{24}) - \Phi(\frac{-92}{24})}{1 - \Phi(\frac{-92}{24})} = 0.63$, similarly, $p_k = 0.84$ and $p_m = 0.46$. Node k gets $p_k L = 0.84 \times 20 = 17$ copies first, since it offers the highest delivery probability. Node j receives $0.63 \times 20 = 12$ or $20 - 17 = 3$, whichever is smaller. Hence, 3 message copies are forwarded to j .

IV. A POST-DISASTER MOBILITY MODEL

In this section, we briefly highlight the main aspects of the mobility model that we use to evaluate our protocol in a disaster-response setup. Abstract mobility models, such as the classical Random Waypoint Model (RWP) and the Map-Based Random Walk Model (MBM) (a variant of Random Walk on a map), do not capture well the recurrence inherent in mobility patterns of people, objects and activities in post-disaster scenarios. In contrast, the mobility model we use in this paper, named the *post-disaster mobility model* (PDM), attempts to mimic the aftermath of disasters such as hurricanes or large earthquakes.

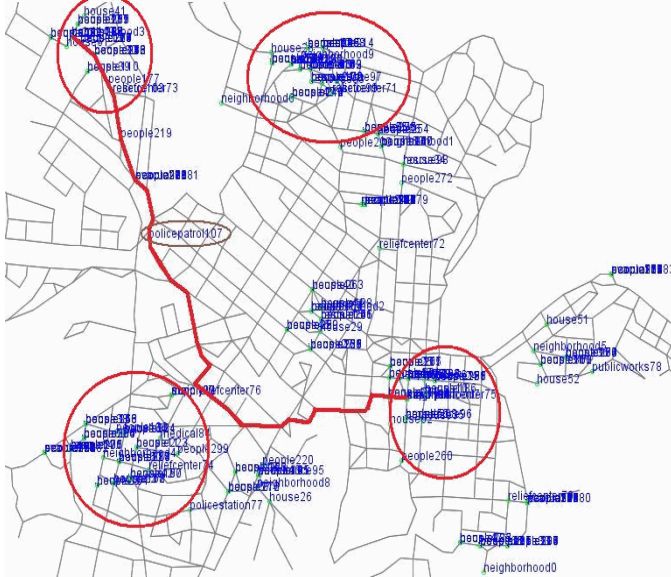


Fig. 3. The disaster area. Circles are neighborhoods. Highlighted path is the part of police patrol.

In the spirit of abstraction and simplification, which has

been the virtue of several widely-used mobility models in current literature, we do not attempt to model the disaster to a high degree of realism, but rather create a recognizable abstraction of the disaster, focusing on key aspects whose effects and ramifications we want to explore. In this case, the key aspect is the existence of some degree of recurrence in modeled activities. For example, consider a Hurricane site in the aftermath of the disaster. Coordination centers and relief camps are established nearby. Vehicles may carry supplies between the coordination centers and evacuation camps. A good number of rescue workers and volunteers are engaged to locate survivors and offer help. A few emergency vehicles may run between relief centers and distressed neighborhoods. Police officers may patrol neighborhoods to prevent unwanted activities.

We have implemented PDM as an extension to the ONE simulator on top of ONE's Map-Based Mobility Model (MBM). MBM uses map data of roads and streets instead of a flat Euclidean 2-D space. We added four movement classes; the InterCenterClass (recurrent motion of supplies between centers), the RescueWorkerClass (localized random motion starting and ending at a given center), the PolicePatrolClass (a recurrently patrolled path through multiple neighborhoods) and EmergencyClass (dispatch of vehicles from a center to a random destination and back). The final class was the class of fixed nodes including centers, households, and stranded vehicles. We used a small city map (that comes with ONE) to describe an urban disaster area with centers, neighborhoods, houses, people, vehicles, rescue workers, and police patrols. Figure 3 depicts a post-disaster scene. We assume that the nodes above (vehicles, rescuers, centers) are equipped with wireless networking devices that can act as 'routers'. We treat all people inside evacuation camps and other gathering areas as attached to the access points in those areas.

V. EVALUATION AND SIMULATION RESULTS

We simulate IC-Routing in ONE (Opportunistic Network Environment) [26] simulator and compare its performance to three other DTN routing protocols, Spray-and-Wait [24], Spray-and-Focus [20] and MaxProp [8]. We are interested in three performance metrics: i) message overhead, ii) delivery ratio and iii) average message delivery delay. Message delay measures the amount of time a message takes to be delivered to the respective destination, whereas delivery ratio gives the fraction of total messages that has been able to reach to the destination. The overhead is computed as the total number of one-hop message transmissions divided by the total number of messages created. We use messages created instead of messages delivered to tease apart the overhead from the delivery ratio. From the routing protocol perspective, message overhead is directly related to the energy consumption. The more message transmissions there are, the more energy is consumed. Hence, energy saving is achieved by reducing overhead. We show that IC-Routing offers lower overhead compared to other protocols while giving a competitive delay and delivery ratio.

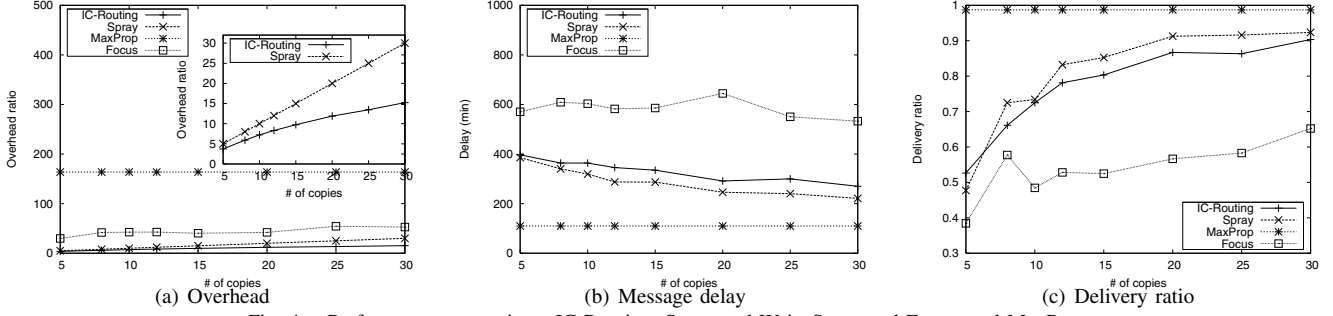


Fig. 4. Performance comparison, IC-Routing, Spray-and-Wait, Spray-and-Focus, and MaxProp.

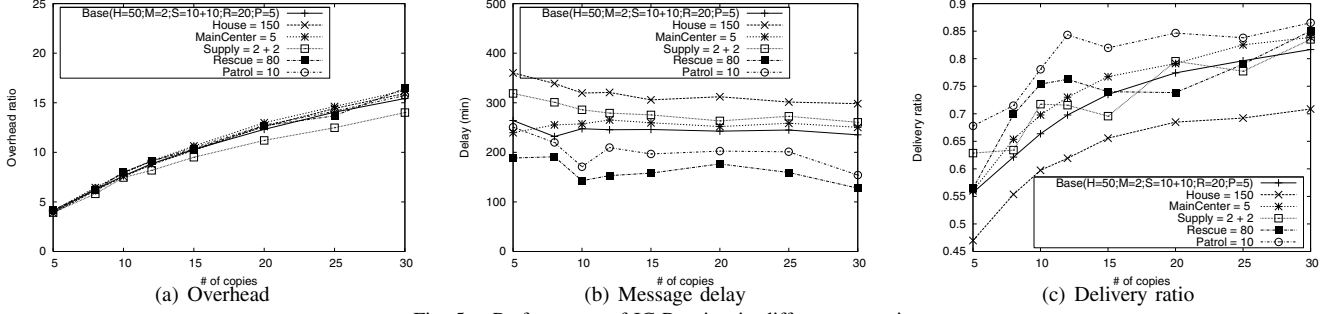


Fig. 5. Performance of IC-Routing in different scenarios.

All experiments, unless otherwise stated, are done for 5 neighborhoods, 2 main centers, 10 relief and evacuation camps, 20 supply vehicles, 20 rescue workers, 5 police patrols, 5 emergency vehicles, and 50 distressed households (although the total number of houses can be much larger). Data traffic is generated as a Poisson process at a rate of 1 message per 5 minutes. For IC-Routing settings, we use *confidence threshold* = 0.75 and *message delivery timeout* = 6 hours. Router's buffer size is 10MB. The simulation duration is 96 hours. For each experiment, the average is taken over 5 runs.

In Figure 4, we compare the performance of IC-Routing to others with the varying number of initial message copies. As we see, IC-Routing gives the smallest overhead among all protocols, whereas MaxProp, an epidemic scheme, has the highest overhead. IC-Routing reduces the number of message transmissions to as low as half of the Spray, without deteriorating the delay and delivery ratio much. As the number of initial copies increases, more messages are delivered to destinations (Figure 4(c)). Usually the nodes that are far away from the source cannot be reached with few copies, because all copies are exhausted before they reach the destination. However, those messages can be delivered if the number of message copies is increased. Hence, message delivery ratio increases with the rise of message copies for both IC-Routing and Spray. MaxProp gives the lowest message delay (150 min) and the best delivery ratio (≈ 1.0). Spray-and-Focus, an improvised version of Spray-and-Wait with the last-copy forwarding, does not perform well in this situation; both the overhead and delivery ratio are very poor. Since we are concerned more about overhead than delay and MaxProp and Focus offer very high overhead, we limit all subsequent comparisons to Spray-and-Wait in the rest of our experiments. We choose Spray because it has an explicit controllable bound on overhead.

IC-Routing is a low-energy multi-copy forwarding scheme.

So, it should keep its overhead very low. As we see from Figure 4(a), IC-Routing has substantially lower overhead than Spray – for the number of copies > 10 , IC-Routing has nearly 50% less overhead than Spray, while maintaining a comparable delay and delivery. Moreover, we see that the overhead of IC-Routing becomes stable and remains constant at higher values of initial message copies whereas for Spray it just goes up, because it spreads messages blindly as long as message copies exist. Hence, Spray must choose its initial message copies carefully. A higher number of copies incurs higher overhead (hence, higher energy consumption) and fewer message copies yields a lower delivery. In contrast, IC-Routing gathers information from the network and regulates message replication/forwarding adaptively and efficiently depending on the degree of information available on recurrent routes. Hence, it can run with a high value of initial message copies without increasing the overhead much, while raising the delivery ratio. IC-Routing ensures the best utilization of message copies by avoiding (unnecessary) duplication of messages.

Figure 5 shows the performance of IC-Routing in different scenarios by changing the number of participants compared to the base setup. We consider houses, main centers, supply vehicles, rescue workers, and police patrols. We observe that when we change the number of nodes keeping their movement pattern the same, the overhead doesn't change much; all variations are reflected in delays and delivery ratios. The number of hops that the first copy of any message takes to reach to the destination remains unchanged when the number of participants changes (usually its 3–4, not shown in figure). Hence, overhead remains unaffected as the number changes, but delays and delivery are affected. When the distressed household count increases to 150, delivery drops and delay becomes high. When rescue workers are increased to 80, both metrics improve, particularly the message delay. When

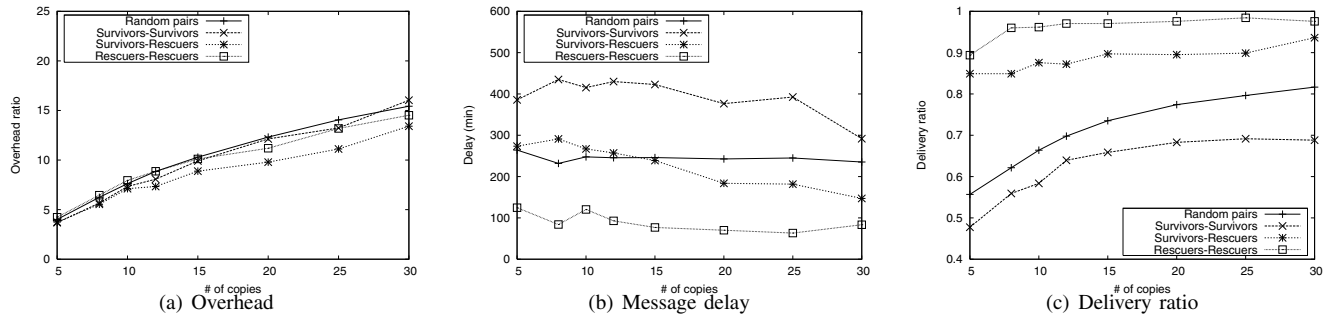


Fig. 6. Performance in various traffic patterns.

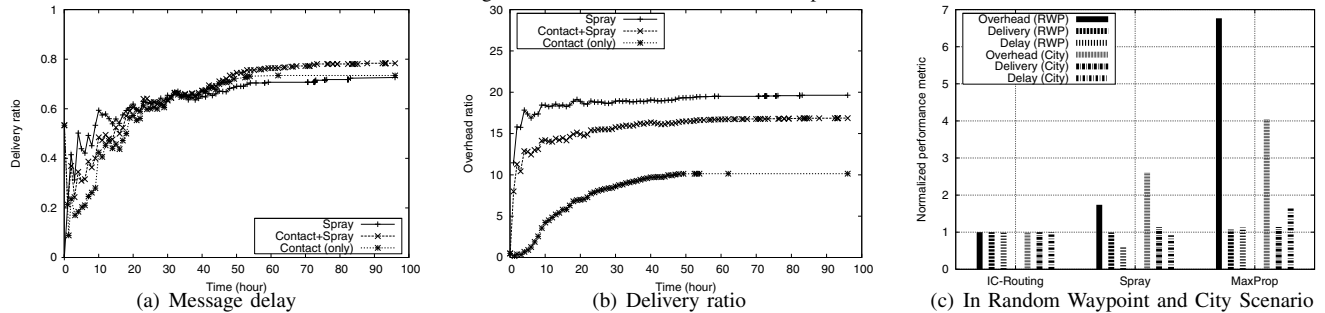


Fig. 7. Performance over time (20 initial message copies) and in other scenarios.

5 main centers are opened instead of 2, delivery ratio increases sharply, but delay increases as well, because meetings with supply vehicles decrease as they report to more main centers. Police patrols have large effects on both metrics. Since patrols connect neighborhoods, increasing the number of patrols provides better connectivity among neighborhoods. Both delay and delivery improve. Table I shows the effect on performance (delay and delivery), as the number of various agents is increased.

TABLE I
EFFECT OF VARIOUS AGENTS, WHEN THEIR NUMBER IS INCREASED.

Agents	Effect	Reason
Centers and camps	improves	exploit recurrence
Back-&-forth vehicles	improves	shorter inter-contact delay
Target dest. locations	drops	sparse, less meeting
Regular patrols	improves	more coverage and meeting
Mobility within cluster	improves	frequent meeting

Figure 6 shows results for different traffic patterns. Earlier results are shown for random source-destination pairs. However, we show that performance varies based on intended traffic. We use three cases: survivors-to-survivors (people at houses or camps send messages to other survivors), survivors-to-rescuers (people asking for help to rescue workers, emergencies, polices), and rescuers-to-rescuers (rescue workers, supplies, main centers, polices, emergency responders, etc pass messages among themselves for relief coordination). For each case, a node in source group randomly chooses a node from destination group and generates message at rate 1 per 5 min. We see that in terms of performance $R-R > S-R > S-S$. Rescue workers are mostly connected by the underlying movement of vehicles and centers, hence they achieve the highest delivery and lowest delay. On the other hand, survivors are the marginal participants in the scene and exploit the movements of rescue operation to carry out their messages.

They encounter the least delivery and highest delay. In all cases, the overhead remains pretty same, but S-R has the lower value, because survivors and rescuers (who forward messages to others) have a bit of repeated meeting.

Figure 7 shows how metric stabilizes over time. Recall that IC-Routing constructs routing table; yet uses *spray* when paths are not constructed or path confidences are too low. Messages are sprayed at the beginning when the routing table is yet to be populated. In Figure 7, ‘Contact (Only)’ uses only routing entries to forward messages, whereas ‘Contact+Spray’ uses both routing entries and spray. We see that ‘Contact’ has smaller delivery ratio at the first few hours, but copes up after a while (after 1 day). However, ‘Contact’ makes far smaller message transmissions (because it does not spray) than the ‘Contact+Spray’ and the Spray-and-Wait. While in case of ‘Spray’ the overhead jumps up at the very first hour of operation, ‘Contact’ takes time to rise and stabilize. This is the time during which IC-Routing learns the network and routes.

In Figure 8, we show the effect of *timeout* on the delivery rate. IC-Routing uses the timeout to compute the message delivery probability to certain destination. At the smaller timeout value, many potential paths appear as ‘bad’ because of their considerably long path-delays. This is why the delivery ratio is low at the smaller timeouts (< 5 hours). When the timeout is set to higher values, more paths become feasible and messages are propagated along them. However, there is a threshold (here it’s around 7 hours) beyond which the delivery ratio does not improve much. This is because at that timeout value, all possible paths are discovered with their mean delays and setting to higher values does not include newer paths any more. Figure 8 also shows the fraction of messages delivered within the timeout against the total delivery. It shows that at the smaller timeout value, there is a gap in timely delivery; but as timeout value rises, the gap vanishes.

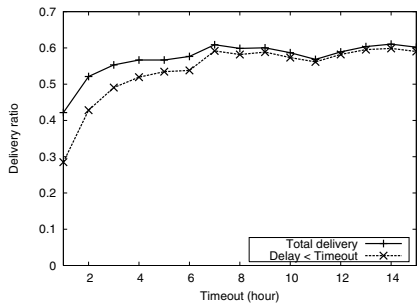


Fig. 8. Effect of delivery timeout on message delivery.

In Figure 7(c), we show the results of our protocol in scenarios other than disaster. We used MapBased Random Waypoint (RWP) mobility and a city mobility (pertains to the ONE simulator) with cars, buses, and pedestrians. We normalized all performance metrics (overhead, delivery, delay) to the IC-Routing’s performance value (IC-Routing’s performance becomes unity). Despite that IC-Routing is tailored for recurrent scenario, it performs well in these scenes also, particularly in terms of overhead. IC-Routing has the smallest overhead, whereas Spray and MaxProp produce overhead (1.7 \times , 2.5 \times) and (6.8 \times , 4.1 \times) of IC-Routing for (RWP, City) respectively. Moreover, IC-Routing has a very competitive delivery and delay compared to others.

IC-Routing introduces additional communication due to exchange of routing table entries. When node i meets node j , each will update the other’s table for vertex ij . Assume the routing table has entries for 100 known destinations. The table size is 800 bytes (delay and variance, 4 + 4 bytes per destination). If the average neighborhood size is 10 nodes and 2 encounters occur with each neighbor per hour, the total bidirectional hourly transfer rate is $10 \times 2 \times 2 \times 800 = 32\text{KB}/\text{hour}$, which about 20 Ethernet packets. Given sufficiently high traffic, such hourly overhead is acceptable.

VI. CONCLUSION AND FUTURE WORKS

This paper presents a novel approach to DTN routing tailored for disaster response networks. In contrast to flooding-based DTN routing techniques, we propose a routing table based approach that computes paths and costs to destinations using a novel routing metric called ‘inter-contact delay’. To construct these paths and costs, we exploit the underlying recurrent movement pattern that prevails in a rescue and relief operation after a disaster. The proposed protocol, IC-Routing, has very low message overhead resulting in big energy saving, one of the most fundamental requirements at disaster time.

REFERENCES

- [1] LA Times, “Chrysler will offer wireless internet access in 2009 models,” June 25, 2008.
- [2] I. Akyildiz, D. Pompili, and T. Melodia, “Underwater acoustic sensor networks: Research challenges,” *Ad Hoc Networks*, vol. 3, no. 3, pp. 257–279, March 2005.
- [3] J. Zhao and G. Cao, “Vadd: Vehicle-assisted data delivery in vehicular ad hoc networks,” in *Proc. of 25th IEEE INFOCOM ’06*, April 2006, pp. 1–12.
- [4] S. Jain, K. Fall, and R. Patra, “Routing in a delay tolerant network,” in *Proc. of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM ’04)*, Portland, Oregon, 2004, pp. 145–158.

- [5] K. Fall, “A delay-tolerant network architecture for challenged internets,” in *Proc. of SIGCOMM ’03*, Karlsruhe, Germany, 2003, pp. 27–34.
- [6] A. Vahdat and D. Becker, “Epidemic routing for partially connected ad hoc networks,” Department of Computer Science, Duke University, Tech. Rep. CS-2000-06, April 2000.
- [7] T. Spyropoulos, K. Psounis, and C. Raghavendra, “Spray and wait: an efficient routing scheme for intermittently connected mobile networks,” in *Proc. of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking (WDTN ’05)*, Philadelphia, PA, 2005, pp. 252–259.
- [8] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, “Maxprop: Routing for vehicle-based disruption-tolerant networks,” in *Proc. of 25th IEEE International Conference on Computer Communications (INFOCOM ’06)*, April 2006, pp. 1–11.
- [9] E. Jones, L. Li, and P. Ward, “Practical routing in delay-tolerant networks,” in *Proc. of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking (WDTN ’05)*, Philadelphia, Pennsylvania, 2005, pp. 237–243.
- [10] C. Liu and J. Wu, “Scalable routing in delay tolerant networks,” in *Proc. of the 8th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc ’07)*, Montreal, Quebec, Canada, 2007, pp. 51–60.
- [11] Q. Li and D. Rus, “Sending messages to mobile users in disconnected ad-hoc wireless networks,” in *Proc. of the 6th annual international conference on Mobile computing and networking (MobiCom ’00)*, Boston, MA, 2000, pp. 44–55.
- [12] A. Balasubramanian, B. Levine, and A. Venkataramani, “DTN routing as a resource allocation problem,” in *Proc. of ACM SIGCOMM ’07*, Kyoto, Japan, August 2007, pp. 373–384.
- [13] W. Zhao, M. Ammar, and E. Zegura, “A message ferrying approach for data delivery in sparse mobile ad hoc networks,” in *Proc. of MobiHoc ’04*, Tokyo, Japan, 2004, pp. 187–198.
- [14] M. A. Wenrui Zhao and E. Zegura, “Controlling the mobility of multiple data transport ferries in a delay-tolerant network,” in *Proc. of IEEE INFOCOM ’05*, vol. 2, Miami, FL, March 2005, pp. 1407–1418.
- [15] T. Small and Z. Haas, “The shared wireless infostation model: A new ad hoc networking paradigm (or where there is a whale, there is a way),” in *Proc. of MobiHoc ’03*. New York, NY, USA: ACM Press, 2003, pp. 233–244.
- [16] T. Small and Z. J. Haas, “Resource and performance tradeoffs in delay-tolerant wireless networks,” in *Proc. of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking (WDTN ’05)*, Philadelphia, PA, 2005, pp. 260–267.
- [17] J. Widmer and J.-Y. L. Boudec, “Network coding for efficient communication in extreme networks,” *Proc. of ACM SIGCOMM 2005 workshops*, 2005.
- [18] S. Jain, M. Demmer, R. Patra, and K. Fall, “Using redundancy to cope with failures in a delay tolerant network,” *Proc. of ACM SIGCOMM 2005*, pp. 109–120, 2005.
- [19] O. S. Anders Lindgren, Avri Doria, “Probabilistic routing in intermittently connected networks,” 2003.
- [20] T. Spyropoulos, K. Psounis, and C. Raghavendra, “Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility,” in *PerCom Workshops 2007*, 2007.
- [21] V. Erramilli, A. Chaintreau, M. Crovella, and C. Diot, “Diversity of forwarding paths in pocket switched networks,” in *Proc. of the 7th ACM SIGCOMM conference on Internet measurement*, 2007, pp. 161–174.
- [22] E. M. Daly and M. Haahr, “Social network analysis for routing in disconnected delay-tolerant manets,” in *Proc. of MobiHoc ’07*, 2007, pp. 32–40.
- [23] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, “Age matters: efficient route discovery in mobile ad hoc networks using encounter ages,” in *Proc. of MobiHoc 03*, 2003, pp. 257–266.
- [24] T. Spyropoulos, K. Psounis, and C. Raghavendra, “Single-copy routing in intermittently connected mobile networks,” in *Proc. of 1st IEEE conference on Sensor and Ad Hoc Communications and Networks (SECON ’04)*, Santa Clara, CA, October 2004, pp. 235–244.
- [25] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot, “Delegation forwarding,” in *Proc. of 9th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc 07)*, 2007, pp. 251–260.
- [26] Opportunistic Network Environment (ONE) simulator, “http://www.netlab.tkk.fi/jo/dtn/index.html.”