Title

Combining On-Demand and Opportunistic Routing for Intermittently-ConnectedNetworks

Author

Garcia-Luna-Aceves, J.J.

Publication Date

2009

Peer reviewed

Combining On-Demand and Opportunistic Routing for Intermittently-Connected Networks

J. Boice, J.J. Garcia-Luna-Aceves and K. Obraczka
Department of Computer Engineering
University of California at Santa Cruz
{boice,jj,katia}@cse.ucsc.edu

***Abstract*— While current on-demand routing protocols are optimized to take into account unique features of mobile ad hoc networks (MANETs) such as frequent topology changes and limited battery life, they often do not consider the possibility of intermittent connectivity that may lead to arbitrarily long-lived partitions. In this work, we introduce the Space-Content-adaptive-Time Routing (SCaTR) framework, which enables data delivery in the face of both temporary and long-lived MANET connectivity disruptions. SCaTR takes advantage of past connectivity information to effectively route traffic towards destinations when no direct route from the source exists. We show through simulations that, when compared to traditional on-demand protocols, as well as opportunistic routing (e.g., epidemic), SCaTR increases delivery ratio with lower signaling overhead in a variety of intermittently connected network scenarios. We also show that SCaTR performs as well as on-demand routing in well-connected networks and in scenarios with no mobility predictability (e.g., random mobility). In the latter case, SCaTR delivers comparable reliability to epidemic routing with considerably lower overhead.**

I. Introduction

The price, performance, and form factors of sensors, processors, storage elements, and radios today are enabling the development of network-supported applications in very disrupted environments, i.e., environments where end-to-end connectivity is not guaranteed at all times because of either the characteristics of the environment or the normal operation of the network nodes. Examples of such applications and environments include monitoring of disrupted phenomena (e.g., wild fires), object tracking, establishment of on-demand network infrastructure for disaster relief or military purposes (in which case, the ad hoc network can be disrupted by terrain, weather, and other natural phenomena, as well as jamming, interference, etc.), peer-to-peer vehicular or interpersonal networks [10] with very sparse connectivity, and mobile ad hoc networks (MANETs) that need not be connected at all times in order to limit interference and contention. In these scenarios, network disconnection is the normal state of operation rather than an exception.

The demand for networking in environments prone to intermittent connectivity poses a challenge because the architects of the IP Internet and MANETs have assumed that physical connectivity exists on an end-to-end basis between sources and destinations for extended periods of time, or at least for the duration of a transaction among communicating parties. This assumption has had profound implications on how communication bandwidth is shared, how routing is accomplished, and how messages are disseminated across computer networks. In particular, routing in packet-switching networks has been based on routing tables that specify the next hop to one or more destinations. Such routing information is derived entirely from topology (or connectivity) information that represents only a snapshot of the state and characteristics of network links at particular instants.

Regardless of the specific mechanisms used in a routing protocol today (e.g., proactive or on-demand routing), computing the routing table entry for a given destination can be viewed as a particular form of searching a database. The routing database can be replicated (as it is done in such topology broadcast approaches as TBRPF [27] and OLSR [9]) or distributed, as in AODV [29] or DSR [20]. Depending on whether the routing database is replicated or distributed, the search algorithm can be centralized (e.g., using Dijkstra's shortest path first algorithm) or distributed (e.g., using a flood search based on route requests and route replies). The routing databases constructed by traditional routing algorithms specify the instantaneous status of a link (up or down), and the value of its parameters such as delay and bandwidth at some specific point in time. The search for routes in such databases produces snapshot paths that have no temporal dimension. Hence, if the network connectivity or link parameters change, multiple paths to

destinations may be affected; the only way most current routing protocol can recover is to search for new paths. This time-independent, reactive approach to changes in network connectivity and link parameters works well as long as the disruptions in network connectivity due to environmental or operational reasons are not so frequent and/or long-lived that they prevent the routing protocol from obtaining time-independent paths to intended destinations.

Starting with the work in the Interplanetary Internet Research Group (IPNRG) [5] of the IRTF (Internet Research Task Force), considerable effort has recently been devoted to the study of networks with intermittent connectivity or very long latencies. Perhaps most prominent in this area is the work by the DTNRG (Delay Tolerant Networking Research Group) [12], which started in 2002 under the IRTF. Section II summarizes prior related work on routing in disrupted environments. From our summary of related work, it becomes apparent that no complete solution exists for on-demand routing that incorporates the network topology's time dependency.

In this paper we describe the SCaTR (Space Content adaptive Time Routing) framework to enable on-demand routing in MANETs with intermittent connectivity [1]. Section III describes SCaTR which we currently implement by extending the Ad-Hoc On-Demand Distance Vector (AODV) routing protocol [29]. Our current instantiation of SCaTR is such that, if the network is connected, it operates exactly as regular on-demand routing, in this case AODV. However, if no direct route is available from source to destination, a node that is deemed closer to the destination than the source will advertise itself as a *proxy*. In this manner, we are assured that the resulting protocol will do no worse than standard AODV in well-connected environments, and better in partitioned networks.

We evaluate SCaTR through extensive simulations comparing its performance against on-demand and epidemic routing under a number of scenarios including random– and predictable node mobility. Section IV presents our experimental methodology while Section V presents our simulation results in detail. They show, for example, that SCaTR's proxies improve delivery reliability in both predictable and random mobility situations, while incurring lower signaling overhead. In predictable mobility scenarios, schedules or trajectories are not assumed to be global knowledge. Instead, the routing algorithm in SCaTR uses mobility histories to improve performance. Further, given enough time, the protocol delivers all possible packets to their intended destinations, achieving perfect reliability. Section VI summarizes our contributions and discusses ideas for future work. We start by reviewing related work in the next section.

[1]This paper builds on our earlier work presented in [1].

II. Related Work

Disruptive networks (also referred to as delay-tolerant, partitioned or disconnected networks) have recently received considerable attention from groups researching topics ranging from interplanetary research [3] to wearable computers [10] and wildlife tracking [21].

Since 2002, the *DTNRG* (Delay Tolerant Networking Research Group) [12] has been active in designing and implementing architectural standards and conducting research in the area of disruptive networks. The group has introduced the concept of *bundle* [14] layer that exists above the transport layer and groups messages into bundles that encompass entire sessions. They have also researched *custody* [15], an approach to reliability in disruptive networks and designed addressing and naming schemes [6] for such networks.

Beginning with Vahdat and Becker's *Epidemic Routing* [34], in which all packets are forwarded to all neighbors, message redundancy has played a large role in delay-tolerant networking. Additional message replicas increase the likelihood of delivery and decrease end-to-end delay by selecting all or many possible paths to their destination. In networks with many nodes and/or data packets, however, the energy and transmission requirements of this scheme can be prohibitively expensive, and recent research [16], [24], [31] has improved on Epidemic Routing by seeking to control message flooding.

Spray and Wait [32] is another, more recent improvement over a pure flooding protocol such as epidemic routing. In this work, only the source can replicate a message, and the amount of replication is proportionate to the number of nodes in the network. It is shown that the method can bound delay proportional to the optimal delay. After 'spraying' several copies of a message, the host 'waits' until one is delivered.

Wearable computers [10] have also been studied as network agents for partitioned networks. This work discusses a method of message forwarding based on the learned structure of the network. The work relies on a selected drop policy, as messages can be replicated across many nodes at a time. It was found that the most effective drop policies are to first drop those packets

which are oldest, or those from nodes that are least encountered.

Other research has focused on nodes with predictable or controllable mobility patterns to aid or enable routing in partitioned networks. Shah et al.'s *Data Mules* [30] are mobile devices that provide connectivity to sparse sensor networks with scheduled trips to retrieve messages from data sources and deliver them to their intended destination. More recently, *Message Ferrying* [35], [22], [36] has been investigated for use in highly partitioned networks. The approach utilizes special nodes called *ferries* whose mobility can be controlled to maintain communication between partitions. Much of the work has focused on route scheduling of the ferries, and synchronization between their routes, as a well selected schedule will have a great impact on timely and reliable message delivery. It has also been shown that the ferries can be used as an energy saving device for other nodes in the network; if there are no ferries nearby, nodes can be turned off to conserve energy. This work makes the assumption that controllable nodes exist in the network, however, there are situations in which these nodes are not available. Our work addresses these situations.

Meruga et al. [25] and Jain et al. [19] take advantage of the periodicity inherent to some mobility patterns and explore routing with perfect knowledge of future communication opportunities. They add the *time* dimension to routing tables and select routes based on a combination of the data's destination and the time of message arrival. Among other techniques, these approaches employ a modified Dijkstra's algorithm to determine shortest paths over time in these structures. The resulting routes will be, without regard to conditions such as congestion and storage space, optimal. These approaches take into consideration global knowledge of node mobility schedules to construct these routing tables. In some scenarios, these schedules may not be available, or too expensive to maintain.

Many metrics to approximate the distance in time and/or space to a destination have been proposed for use when node schedules are not available. Some approaches [33], [8] have relied on past mobility and topology knowledge to predict future behavior. The utility functions to determine these links differ, although they all rely on the premise that links that once existed are likely to exist again in the future. One notable protocol, *MaxProp* [2], showed better performance in a deployed network of buses than an oracle with perfect schedule knowledge. The improvement was attributed to the load balancing implicitly caused by MaxProp's imperfect predictions of future routes.

Pocket Switched networking [18], [7] is another phrase for networks in which connected routes are not always guaranteed. The focus of this research has been to model the distribution of contact and inter-contact times in order to better design forwarding strategies. They have found that, in several traces of human mobility data, these distributions have followed an approximate power law. This is further evidence that mobility models commonly used in networking, such as the random waypoint model, are not representative of human-induced mobility. Their observations lead to some proposed mechanisms to extend opportunistic forwarding protocols for better performance in such environments.

Disconnected Transitive Communication [8] is a proposal to add communication across clusters in an ad hoc network. It uses utility values to decide which node in the cluster is best suited to transmit a message to the destination. These utility values are tunable by the application designer, so that various factors can be weighted differently depending on the application. The approach relies on integrating application level knowledge with the routing layer, which may prove difficult if many applications are present in any network.

Work has also been done to predict the future topology of a network based on the current properties of nodes [33]. The main motivation is to determine for *how long* nodes that are connected will remain connected. This allows routing protocols to proactively search for a new route when a link is expected to break. Metrics such as node speed, direction, and radio propagation range are factored into the estimate, and these metrics could also be good indications of future node meetings.

Efficient route discovery [13] has been proposed for use in on-demand routing protocols. The goal is to decrease the amount of overhead incurred by the route discovery process by forwarding route requests *in the direction* of the destination. This is similar to the data propagation approach described in this thesis.

An important consideration in intermittently connected networks is the metric used to determine distance to destinations. The CAR [26] algorithm is one such measurement that uses adaptive weights on several node attributes, as well as Kalman filters. In this work, they maximize the probability of a node's ability to deliver a given message based on a weighing of many factors, including mobility and remaining battery power. The weights for these factors can either be tuned or learned over time.

Location information is a useful tool for routing in

disruptive networks. Both MobySpace [23] and MV routing [4] are methods that use location information to aid routing decisions. They assume that a node who has visited a particular location is likely to revisit it, and therefore is a good candidate to carry messages to that location. MV routing uses location information to facilitate buffer management, while MobySpace is a framework for generating probabilities that a nodes will move to specific locations in the future. Both methods require some sort of localization method such as GPS.

While there has been significant prior work on the topic of routing in partitioned networks, most of the approaches have made one of three assumptions. One assumption is global topology knowledge, which has proved to be useful in determining future connectivity. Another assumption is the existence of controllable nodes, which can be extremely useful in aiding delivery in sparse networks. The third assumption is that data can be duplicated freely among nodes, which can lead to excellent delivery ratios. Our work is an attempt to do away with these assumptions, yielding a more general framework applicable to any network scenario where any such assumption may be unrealistic. Additionally, our framework allows, when available, the use of a-priori knowledge, such as node schedules, node location information, etc. Making use of such information is one direction of future work we plan to investigate.

III. Space-Content-adaptive-Time Routing (SCaTR)

The SCaTR framework is an extension to on-demand routing that takes action only when direct routes cannot be established by the underlying protocol. In the case of a route discovery failure, i.e., the source and destination are in separate partitions, SCaTR tries to route data to the node or nodes in the source's partition that are deemed closest to the destination. These nodes act as *proxies* for the destination and buffer messages until either the destination is discovered, or another node is selected as a better proxy for those messages. Messages are replicated at most one time, resulting in minimal data duplication and duplicate filtering overhead.

Proxies are selected based on past connectivity information which nodes keep in content-adaptive *contact tables*. As it will become clear, contact tables, which are the equivalent of traditional routing tables, use time-dependent and space-dependent routing metrics. These metrics can be different for different types of content or local constraints (e.g., buffer size). For instance, if a proxy is running low in buffer space, it may decide to select as the next proxy for that destination the first node it hears from that has been in contact with the destination; this is done even if the node's contact value is lower that its own; however, if the node has higher buffer availability, it can carry the data for a longer interval. The current version of SCaTR does not explicitly address content adaptively. This is an extension planned as future work.

Because SCaTR takes no action if routes are successfully established, we are guaranteed that it will perform *no worse* than the underlying on-demand routing protocol in any situation. This diversity makes it well suited for adoption by any network scenario.

SCaTR consists of several phases, namely: contact table maintenance, route discovery, route selection, and proxy rediscovery, all of which are described in detail in the remainder of this section. The general operation of the SCaTR framework is shown in Figure 1. Each phase is described in detail below.

A. Contact Table Maintenance

Each node in the network maintains a contact table, containing a measure of time-dependent distances to other nodes in the network. Each entry in the table consists of a destination address and its current contact value. Nodes maintain these tables with information that is piggybacked onto control messages; when a node receives a hello message from a neighbor, it also receives that node's contact table. It uses this table to make changes to its own contact table.

Because it can be very expensive to maintain contact information about all nodes in a network, SCaTR initializes its contact tables on-demand. Each node starts with an empty table, and adds destinations only when it receives a request for that destination, or meets another node who has an entry in its table for that destination. Entries are timed out after a finite period, though this period must be longer than the expected cycle time between node meetings. This method of initialization results in a delay for the first messages introduced into the network, because the contact table request must propagate to the destination and then back to the source before proxies are advertised. In large networks with predefined sinks or a limited number of destinations, however, this method can save significant computational overhead.

To calculate contact values, time is broken into *hello intervals*. During a hello interval, nodes maintain values for each destination by maximizing their neighbors' advertised contact values. These values are averaged

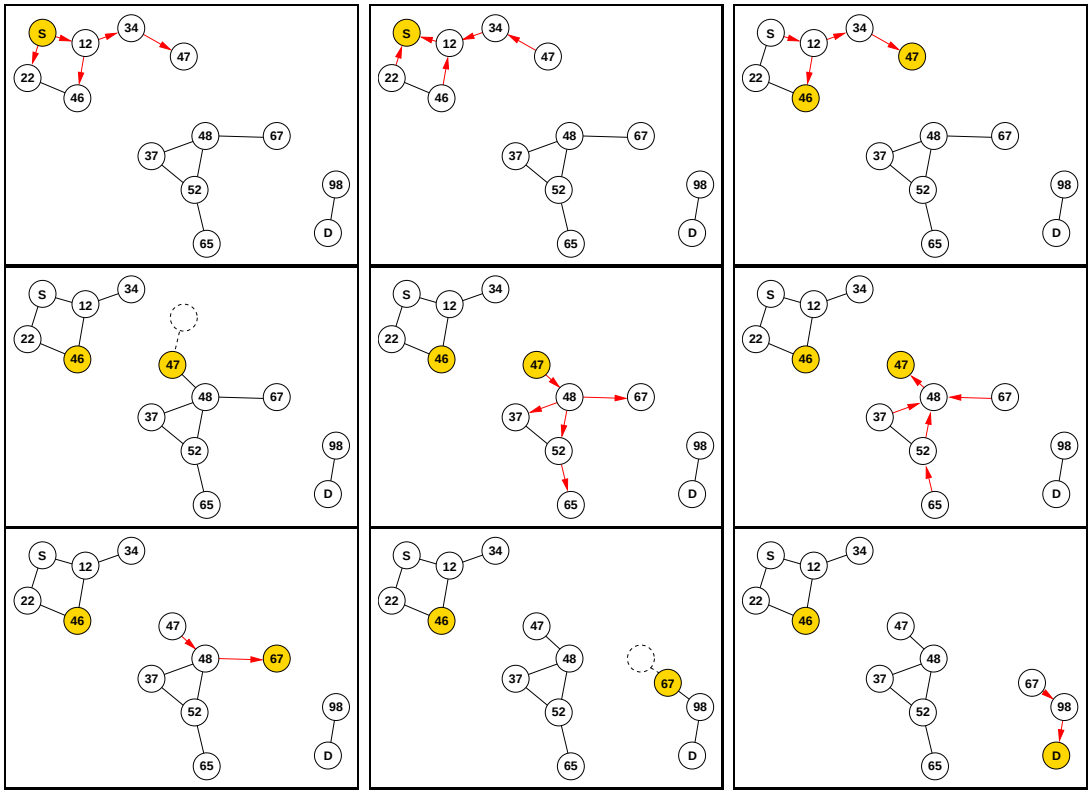

Fig. 1. An overview of the SCaTR framework. The source node (top left) has data (indicated by shading) to send to the destination (bottom right). The source node first initiates proxy discovery in its local partition. After finding the two nodes with the best contact values for the given destination, the source selects them as proxies and sends them data to be buffered. When one of the proxies joins a new partition, it initiates the proxy discovery process, and selects the best proxy in that partition for the destination. Finally, a proxy reaches the partition containing the destination and delivers the data.

over a *hello period*, and the nodes maintain a window of periods, used to generate the next contact value for each destination. Pseudocode, as well as a more detailed description of contact value maintenance, is below.

UPDATECONTACTVALUES

for each $c \in C_n$
  do if $(N_i > MAX_i)$
  then $\{MAX_i \leftarrow N_i$

1) Hello Interval: The length of a hello interval is the amount of time between hello message advertisements. During each hello interval, a node maintains the maximum contact value it receives from its neighbors for each destination. For instance, if a node has four neighbors, and receives contact values of 22, 34, 46 and 47 for a destination during one hello interval, it will use 47 as its value for that interval. Figure 2 illustrates a single hello interval for the node in the center, whose value for the interval is the maximum of its neighboring nodes.

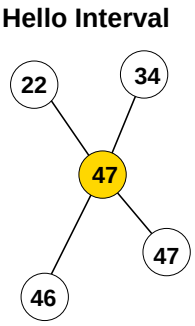

Fig. 2. The value of a hello interval is determined by the maximum value advertised by a nodes' neighbors during the interval.

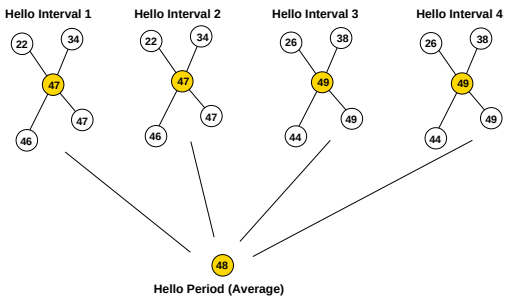

Fig. 3. The value for a hello period is found by averaging τ successive hello intervals.

2) Hello Period: The hello period, given by the parameter τ, is the number of successive Hello Intervals over which a node averages destination-specific contact values. A larger value of τ results in more coarsely grained time periods, while a smaller τ maintains finer time periods. At the end of each hello period, a node averages the values obtained in each hello interval to obtain a single value for the entire period for each destination. Figure 3 illustrates a hello period with $\tau = 4$; the node at the center averages its hello values for each interval to determine its value for the entire period.

B. Period Window

The length of time for which to maintain contact information is specified with the period window κ. A larger value of κ will maintain more mobility history, resulting in routing decisions based on behavior further into the past. A small value of κ will base routes only on recent mobility information. At the end of each hello period, the value obtained during the most recent period is added and the oldest value is removed. Each node's contact values are recalculated at the end of each period by averaging the most recent κ periods. Figure 4 illustrates a node's contact value when calculated at the end of a hello period.

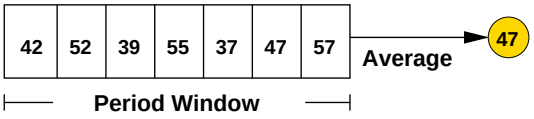

Fig. 4. A node's current contact value is determined by averaging its values for the past κ hello windows.

C. Route Discovery

The main mechanisms for route discovery in the SCaTR framework are the Proxy Request (PREQ) and Proxy Reply (PREP) messages. In the situation that the underlying routing protocol is unable to establish a route to a destination, the source node will issue a PREQ to its partition. This is a request for the node or nodes in the source's partition who are candidates to buffer data and carry it towards its destination.

The PREQ can be a request for multiple destinations, to reduce signaling overhead when a proxy has buffered packets for multiple destinations. Instead of sending out an individual PREQ for each destination, the node will send a single PREQ for all destinations for which it has buffered data. The PREQ also contains the source node's contact value for each destination that is being requested. The contact value is included so that only nodes with a significantly better value reply to the request.

The PREP is a message that advertises the responding node as a possible proxy for one or more of the destinations in the PREQ. The PREP contains several fields so that the source can decide which proxy or proxies to use for each destination. The PREP contains the proxy's contact value for the destination, as well as its remaining buffer space, and the number of messages it has already buffered for the source-destination pair. It is important to distinguish that the PREP specifies the destination, not the proxy itself. The source need not know the address of the proxy; it must only know the next hop towards the destination. The source-destination pair obtained from the PREQ packet is entered into a buffering table at the proxy, and any packets that are received by the proxy with this source-destination pair are buffered.

ROUTEDISCOVERY

for each $M_d \in M_D$
do { $SendRREQ(M_d)$
if $(NoRREP(M_d))$
then { $Buffer(M_d)$
$SendPREQ(M_d)$
$SetBufferTimer(M_d)$

Each node receiving a PREQ takes action based on its contact values for the requested destinations and the source's contact values that are included in the request. A single PREP message is formed for the request based on all requested destinations. For each destination included in the PREQ, there are several possible actions that can be taken by a receiving node. These are outlined below:

PREQ Case 1: The receiver has no entry for the destination in its contact table. If a receiving node has no contact information for a destination, the destination is added to the receiver's contact table. The receiver will now begin to maintain contact information about that destination. The PREQ is rebroadcast as received.

PREQ Case 2: The receiver's contact value for the destination is not significantly better than the source's. In this situation, the receiver sends no reply to the source and rebroadcasts the PREQ as received. It is not a valid proxy for the destination.

PREQ Case 3: The receiver has an improved contact value for the destination. In this case, the receiver responds to the source as a possible proxy for that destination. It adds the destination to its buffer table,

and any subsequent data packets that are received are buffered. It is possible that there are other proxies for this destination in the partition, so the PREQ must be rebroadcast. It is rebroadcast with the *receiver's* contact value for the destination to reduce the number of nodes that reply to the request.

PREQ Case 4: The receiver has an active route to the destination. In the case that the receiver has an existing route to the destination, it advertises the route to the source. This indicates that the destination is in the same partition as the source, and SCaTR functionality is not needed. The source will add this destination to its routing table as an ordinary route. Because there is no need to find proxies for the destination, the PREQ is not rebroadcast.

PREQ Case 5: The receiver has already processed a PREQ for the given destinations in the recent past. In this case, the receiver takes no action, and the PREQ is not rebroadcast.

D. Route Selection

As the source collects PREPs from the nodes in its partition, it compares contact values for each of the destinations. If a new PREP has a higher contact value than the one currently in its routing table, it replaces the entry. Routing tables can contain two types of routes. One is an active, connected route, while the other is a route to a destination hat was advertised by a proxy. These are distinguished only because a reply that advertises a direct route will always take precedence over a proxy route.

Because one of the primary goals of SCaTR is to reduce overhead caused by duplicate messages, and based on prior work [11], in our current implementation, a source can select at most two proxies for each message.

E. Proxy Rediscovery

After a proxy buffers packets, it must ensure that those packets reach their destination. Proxies have several methods of initiating the route discovery process for buffered messages. One method is 'listening' as updates to its contact table are received. If it receives an improved contact value for a destination for whom it has buffered packets, the node assumes that a better route to the destination is available in its partition, or the destination itself is nearby. This information initiates route discovery behavior in the proxy. The proxy sends out a single PREQ for all nodes for which it has buffered packets.

In the same manner as the route discovery process described above, nodes reply to this request, and the proxy selects the best next destination for the data. The proxy with data also listens to any RREPs or PREPs that it relays, and if route information pertaining to destinations in its buffer is relayed, a route is established along the advertised path. In the case that a proxy does not make any updates to the contact value for a buffered destination, or relay a reply for that destination, it sends a PREQ periodically. Pseudocode for proxy behavior is shown below.

PROXYBEHAVIOR

```
while (not Empty(Buffer))
do { SendPREQ(Buffer_D)
     for each (PREP)
     do { if (PREP_Contact > CurrentRoute_Contact
              and
              PREP_Contact > MyContact_Dest + thresh)
          then { ReplaceRoute(PREP)
                 SendMessages(PREP_Dest) }
```

F. Buffer Considerations

SCaTR employs separate buffers for a node's own messages and those sourced at other nodes in the network. The proxy buffer, or the buffer holding messages for other nodes, deletes messages based on their age. Oldest messages are dropped first, regardless of their destination or other attributes. The buffer for a node's own messages is assumed to be large enough to hold all messages that are produced.

G. Message Replication

There are many approaches to message replication, ranging from the minimal approach of a single copy of each message, to the unlimited duplication (up to the number of nodes in the network) of Epidemic routing. More message redundancy requires increased overhead in the form of replicated data, as well as the computational and memory requirements required to maintain buffers and detect duplicate messages.

One approach to the problem is to allow each proxy to replicate a message a certain number of times. However, as the network scales, it is clear that this number will grow exponentially. In addition, a proxy holding a message has no indication of whether another proxy has successfully delivered the message already. Thus, any replicas made by this proxy are in waste. This clearly is not a feasible solution. It is also possible to control replication in a tree fashion with the message source as the root of the tree. There can be a defined

branching factor or replication degree that each level of the tree is permitted to use. Using the depth of a message in the tree and the branching factor, one can specify an exact number of duplications for each message. This reduces message duplication, but has the same problem of communicating message deliveries as mentioned above.

In SCaTR, we have decided to minimize the amount of message replication. Prior work by one of the authors [11] has indicated analytically that under constrained conditions, the optimal number of message duplications is one. The overhead resulting in any further duplication is not justified by the increase in delivery ratio. For these reasons, the source can select at most two initial proxies for message transmission, and proxies may not duplicate messages.

IV. Performance Evaluation Methodology

We evaluated SCaTR's performance through extensive simulations using the GloMoSim [28] network simulator. We employed a variety of scenarios including different network topologies and mobility patterns. We compared SCaTR against on-demand and epidemic routing. The latter is used as baseline as it achieves the best possible delivery rate. Because the networks we simulated are sparse, it is often not possible to deliver all messages.

Our results show that the addition of SCaTR to an on-demand routing protocol causes significantly improved performance in terms of delivery ratio and signaling overhead in all mobility scenarios that were tested. In our experiments, signaling overhead includes control messages such as route requests, as well as message replicas.

A. Network Connectivity Metric

Network connectivity has been defined as a metric for evaluating how well connected simulation scenarios are. It is defined as the ratio between existing routes and all possible routes in the network, averaged over the lifetime of the simulation.

Given n, the number of nodes in the network, we can calculate M_t, the maximum possible number of routes at time t as

$$M_t = \frac{n^2 + n}{2}$$

. With knowledge of P_t, the set of partitions at time t, and $|P_t|$, the *number* of partitions at time t, we can calculate R_t, the actual number of routes at time t as

$$R_t = \sum_{\{P_t\}} \frac{|P_t|^2 + |P_t|}{2}$$

.

Therefore, network connectivity at time t is

$$C_t = \frac{R_t}{M_t}$$

and the percent of connectivity for the duration of the simulation is

$$PercentConnectivity = \frac{1}{t} * \sum_{t=1}^{MaxT} C_t$$

.

B. Simulation Setup

The SCaTR framework was implemented as an extension to the AODV routing protocol. Epidemic routing, which passes messages to all neighbors, was used as a basis for comparison. Because each message takes all possible routes, the protocol provides an upper bound on delivery rates, however, it incurs large amounts of signaling overhead due to data replication.

A "controlled" version of epidemic routing was also implemented. In this variant, packets are only sent to nodes that are deemed closer to the destination based on the same heuristic used by SCaTR. This limits the scope of the flood, although messages may still be replicated many times.

Additionally, we also compare SCaTR against AODV using both the standard version ([29] and an AODV variant in which data sources, when failing to establish a route to a destination, retry their RREQ periodically. Standard AODV limits the number of RREQ retries to 2 in order to reduce routing overhead. Employing this AODV variant in our performance study was our attempt to compensate for the fact that AODV was not designed for partitioned networks.

In all experiments, twenty CBR data flows generated messages at one second intervals for the first 400 seconds of the simulation. The experiments run for 2000 seconds to give messages time to propagate to their destination. The simulations contained 64 nodes, and used an 802.11 MAC layer. All experiments were run with 20 different seed values and the data points presented in the graphs are the average over all the runs.

Two mobility scenarios, in addition to the random waypoint model, show the effectiveness of the protocol over different topologies and connectivity models. We describe the mobility scenarios we used below.

C. Mobility Models

1) Gridded Random Waypoint: Gridded random waypoint mobility was implemented to illustrate a network in which nodes move randomly but in predefined areas. The resulting scenario exhibits limited connectivity, where nodes are selected as proxies based on their distance to the destination.

More specifically, in this scenario, nodes are arranged in a square field with a 377m radio propagation range. Within each square, a fixed number of nodes move according to the random waypoint mobility model; random locations are selected within the square, and a node moves there at a rate of between 5m/s and 10m/s. After reaching its destination, a node pauses for 100s. 20 constant bit rate flows exist between nodes on opposite ends of the grid to provide maximum route lengths. Throughout the experiments using this mobility pattern, the number of nodes and flows are fixed, while the dimensions of the scenario are varied to provide more or less connectivity in the network. A sample topology is shown in Figure 5.

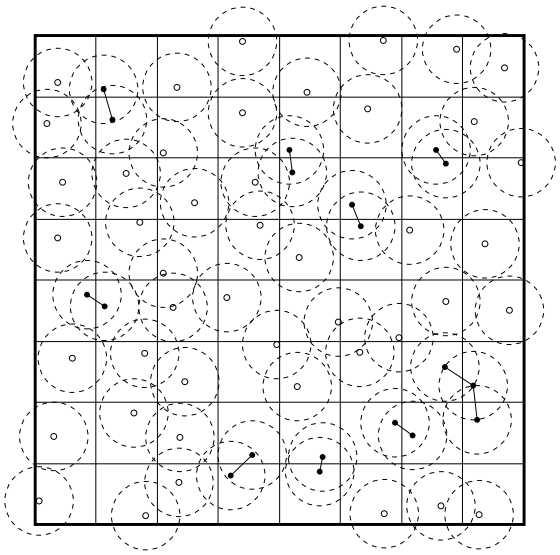

Fig. 5. Gridded random waypoint mobility: each node moves randomly within its square in a grid. The dotted circles indicate the radio propagation range. Source and destination nodes are all at the edges.

2) Scheduled Routes: The scheduled routes scenario was generated to illustrate predictable motion in a network. Because each node follows a predefined path, it provides a situation where past topologies are a good indication of future connectivity. In this case, the contact tables of SCaTR can more accurately represent distance to destinations.

For these experiments, 64 nodes are arranged in a square field with 20 constant bit rate flows, and a radio propagation range of 377m. Ten nodes are positioned around the perimeter of the network and act as sources and destinations. All other nodes are assigned a randomly sized and positioned rectangle over which to travel at a random speed of between 5m/s and 20m/s. Using these parameters, links will be somewhat periodic and thus predictable, although they will not always occur at the exact same time. Throughout the experiments, the size of the scenario is increased to provide less connectivity in the network. A sample topology is shown in Figure 6.

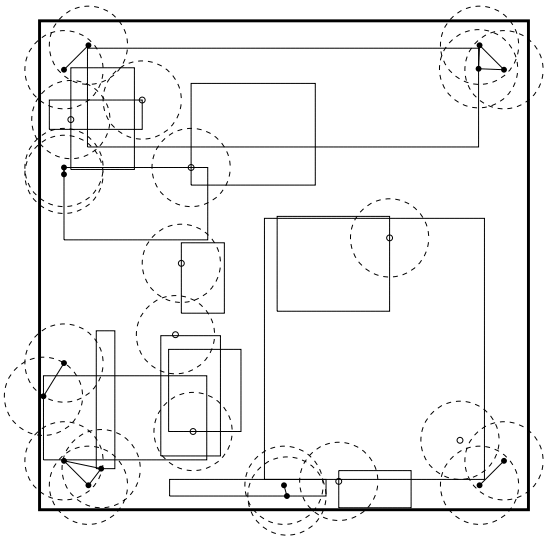

Fig. 6. Scheduled routes mobility: rectangles indicate scheduled node trajectories while dotted circles indicate their radio propagation range. Source and destination nodes are placed around the perimeter. This particular scenario is quite sparse for legibility.

3) Random Waypoint Setup: We also run experiments with the random waypoint mobility model to illustrate the performance of SCaTR in a purely random network; in this case, past mobility information gives no indication of future topologies. This scenario is especially interesting as it is not favorable to SCaTR and consequently provides a lower bound on the performance of the algorithm.

The scenario has 64 nodes move according to the random waypoint mobility model within a square area 4000m x 4000m. Due to the sparseness of the network, the connectivity is poor. The experiments vary the speed of each node to change the duration of connected paths in the network. Pause time remains a constant 30s throughout the simulations.

V. RESULTS

A. Gridded Random Waypoint

We first evaluate SCaTR in the gridded random waypoint mobility scenario. Connectivity is varied by increasing the size of the grid; a larger grid results in lower connectivity. At small scenario sizes with good network connectivity, the delivery rate for all protocols is shown in the left-hand side of the graph in Figure 7. Both controlled- and regular epidemic routing deliver 100% of messages, while SCaTR delivers 94%. AODV delivers 72% of messages whether or not it buffers packets at the source. As the scenario size increases (towards the right size of the graph) and network connectivity diminishes, the performance of AODV quickly decreases, as there is rarely an entire connected path from source to destination. Once the scenario size reaches 3000m x 3000m, AODV is unable to deliver any messages. Epidemic routing maintains 100% delivery rate until the scenario size reaches 3200m x 3200m, and SCaTR sustains a delivery rate near to that of controlled epidemic routing throughout the experiment. At 4000m x 4000m, connectivity is very poor, and none of the protocols deliver more than 10% of the messages. It is also worth noting that using one or two proxies does not impact performance considerably. This is true for most scenarios we studied.

Signaling overhead, seen in Figure 8, is similar for all protocols at high network connectivity. AODV's route discovery process reaches most nodes, causing large amounts of overhead. Epidemic routing, although it does not incur any route discovery overhead, transmits all messages to all nodes, creating many duplicates of each message. As connectivity decreases, AODV's signaling overhead rises sharply due to its many route discovery failures. The overhead for both epidemic routing and SCaTR remains fairly constant at 63 and 25 messages per delivery, respectively. Epidemic routing remains constant at approximately 63 messages per delivered packet because each message that reaches the destination is likely to reach all 64 nodes in the network, since source and destination are at opposite ends of the grid.

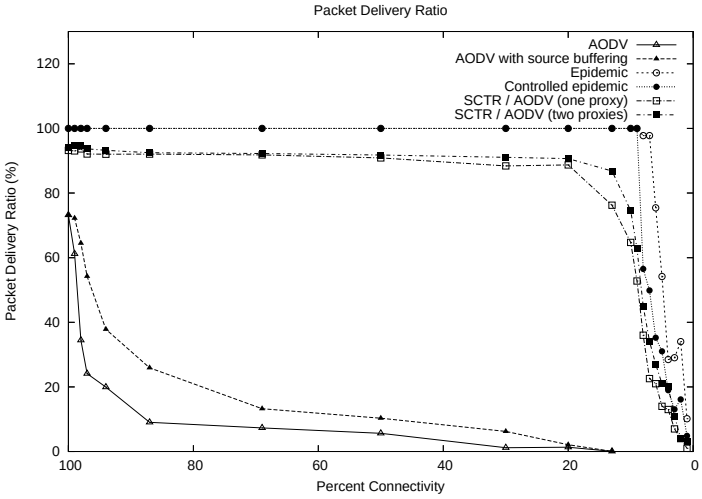

Fig. 7. Delivery ratio for gridded random waypoint mobility model with varied network connectivity.

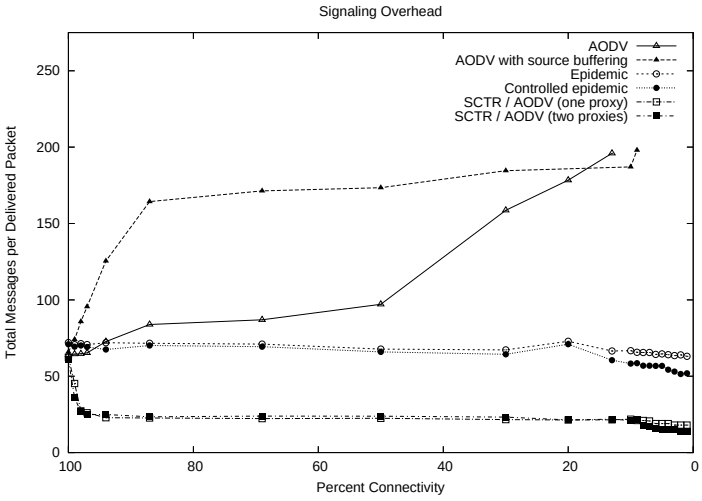

Fig. 8. Signaling overhead for gridded random waypoint mobility model with varied network connectivity.

B. Scheduled Routes

Experiments with scheduled routes show that SCaTR takes advantage of predictability to provide high delivery rates with low signaling overhead. Figure 9 illustrates delivery rates for the various protocols. The behavior is similar to gridded random waypoint mobility; AODV's performance decreases sharply as connectivity decreases, while AODV extended with SCaTR maintains delivery rates comparable to epidemic routing. Figure 10 shows signaling overhead for the scheduled routes model. The results are similar to those for the gridded scenario and show that the addition of SCaTR enables AODV to maintain fairly constant overhead.

C. Buffer Limitations

When buffer limitations are introduced, SCaTR's performance significantly improves as compared to epidemic routing. As shown in Figure 11, excessive message replication in epidemic routing has a detrimental effect on performance for smaller buffer sizes. In these

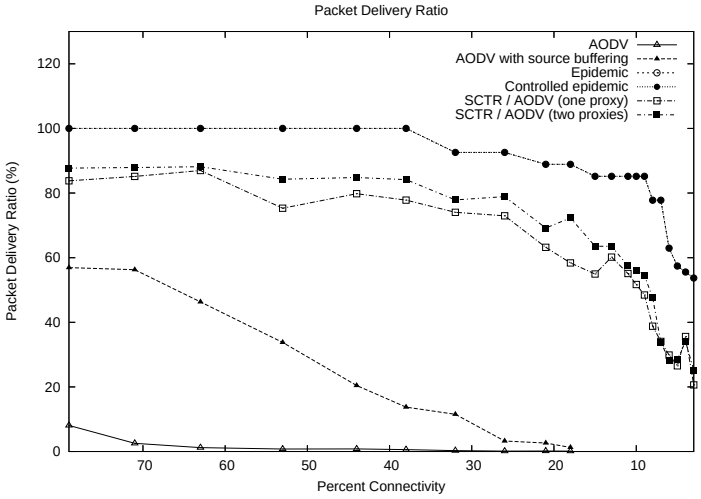

Fig. 9. Delivery ratio for scheduled routes mobility model with varied network connectivity.

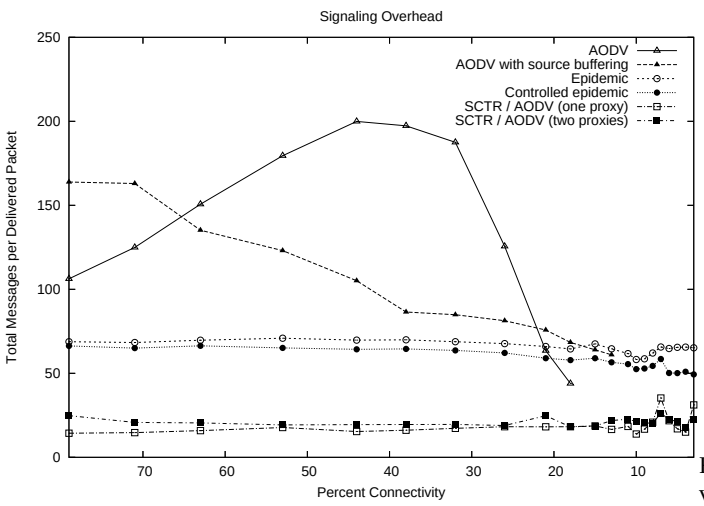

Fig. 10. Signaling overhead for scheduled routes mobility model with varied network connectivity.

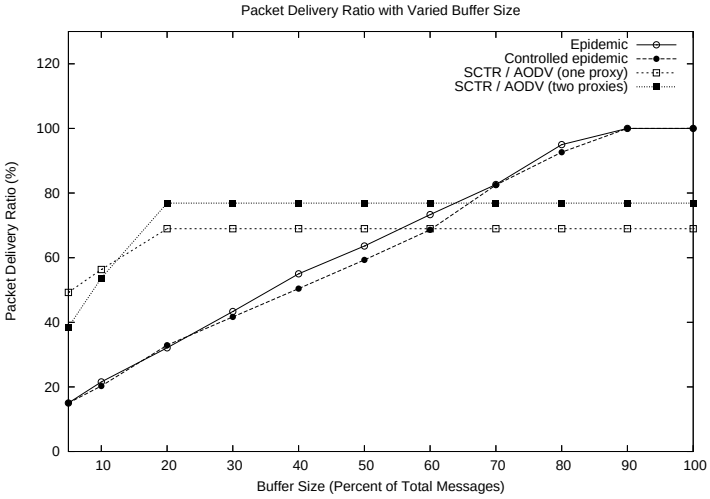

Fig. 11. Delivery ratio for scheduled routes mobility model with limited buffers.

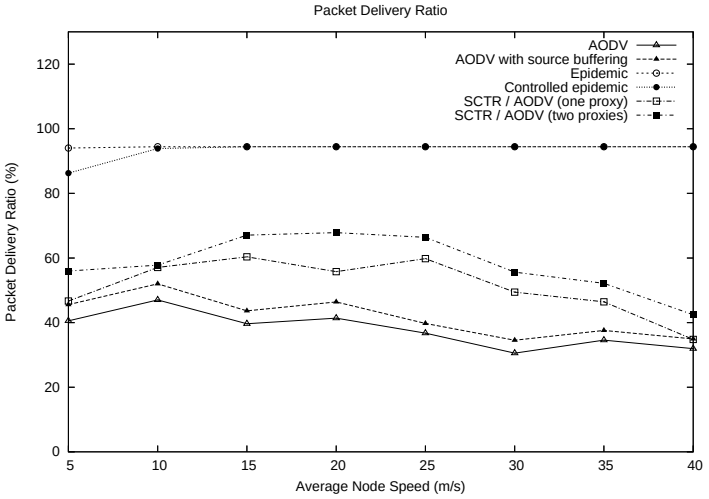

Fig. 12. Delivery ratio for random waypoint mobility model with varied node speed.

scenarios, only when the size of the buffer reaches approximately 60% of the number of messages originated in the network does epidemic routing's delivery rate surpasses that of SCaTR. This improvement, however, is still offset by the high overhead of epidemic routing. It is interesting to note that with very constrained buffer sizes (in these scenarios around 5% to 10% of the total number of messages), it is beneficial to select only one proxy for each message.

D. Random Mobility

Figures 12 and 13 illustrate performance of the protocols with random waypoint mobility in a 64-node, 4000m x 4000m scenario. Node speeds were varied to provide more or less connectivity in the network. SCaTR performs better than standard AODV, though not significantly better, because it is unable to take advantage of past contact information to predict future topologies.

E. Performance Over Time

Figures 14 and 15 show that SCaTR's performance improves as the duration of simulation experiments in-

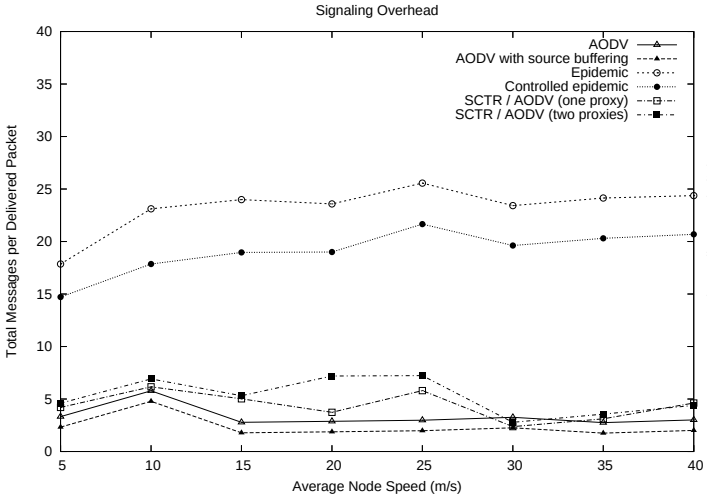

Fig. 13. Signaling overhead for random waypoint mobility model with varied node speed.

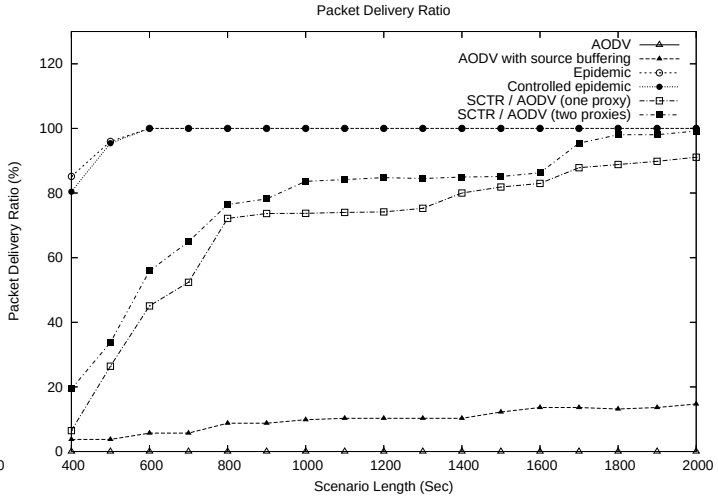

Fig. 14. Delivery ratio for scheduled routes mobility with varied scenario length.

creases. When simulation time increases, delivery rates increase and signaling overhead remains constant. This is because messages are sent at a constant rate for the first 400 seconds, after which, no new messages are introduced into the network, and existing messages are given time to propagate to their destination. Despite additional time, AODV does not improve its delivery rate significantly, because there is never a fully connected route to the destination. Epidemic routing quickly achieves a 100% delivery rate, while SCaTR is able to achieve better than 95% after approximately 2000 seconds.

Despite the fact that SCaTR takes longer than epidemic routing to deliver messages, it does so with far less signaling. After an initial spike in signaling overhead due to low delivery ratio (most packets are en route to their destination at this point), SCaTR settles at approximately 20 messages per delivered packet. Epidemic routing settles at approximately 63 messages per delivered packet, since there are 64 nodes in the network.

Despite the fact that SCaTR takes longer than epidemic routing to deliver messages, it does so with far less signaling. After an initial spike in signaling overhead due to low delivery ratio (most packets are en route to their destination at this point), SCaTR settles at approximately 20 messages per delivered packet. Epidemic routing settles at approximately 63 messages per delivered packet since there are 64 nodes in the network.

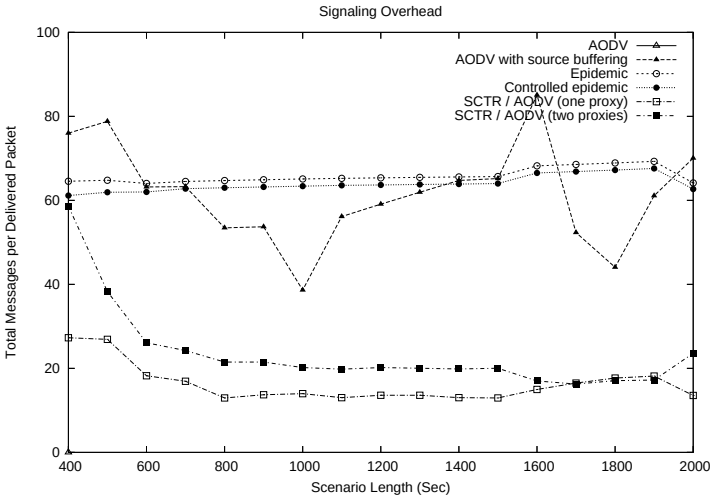

Fig. 15. Signaling overhead for scheduled routes mobility with varied scenario length.

VI. Conclusion and Future Work

We have introduced the Space-Content-adaptive-Time Routing (SCaTR) framework to efficiently and effectively route data in networked environments where connectivity is intermittent. SCaTR extends on-demand routing to operate in environments where, often times, there may not be a direct route between source and destination. Thus, if the network is connected, SCaTR operates exactly as regular on-demand routing. However, if source and destination do not have a connected route, SCaTR chooses a node that is deemed closer

to the destination than the source as a *proxy* for that destination. The proxy will either deliver the data to the destination directly, or choose another proxy closer to the destination than itself. In summary, the resulting protocol will do no worse than standard on-demand routing in well-connected environments, and far better in partitioned networks. Through extensive simulations in environments with varying connectivity and mobility patterns, we showed that SCaTR can yield considerably higher delivery ratio with lower signaling overhead than traditional on-demand routing.

SCaTR makes no assumptions regarding nodes that can broadcast schedules; our routing decisions are based solely on information gleaned from past topologies. It is conceivable, however, that a heterogeneous network may be able to provide different routing information depending on the type of device. It would be useful to combine a-priori mobility knowledge with our predictive model to improve routing decisions; when no definite schedule information is available, a node will fall back to forwarding data based on predicted mobility. In addition, nodes could broadcast trajectory or location information if available. We have explored routing without these assumptions, with the possibility of adding them later to improve performance.

In addition, we are working with more realistic network scenarios taken from actual network traces, such as one available from Dartmouth College [17].

One discussion this work stimulates relates to the problem of implementing end-to-end reliable transport in partitioned networks. Since average latency can be very high, lost packets are not noticed until they propagate to their destination, and a response message reaches the source. The amount of state needed to guarantee end-to-end reliability (a la TCP) would be enormous. This calls for lower level reliability so that packet losses are noticed immediately and can be repaired. We plan to explore the impact of low-level reliability on routing in partitioned networks.

In our approach to on-demand routing in disrupted networks, we first attempt to establish an active route with the underlying protocol, and, as an alternative, forward messages towards the destination. Depending on the scenario, it may be more efficient to first try forwarding data, and if a connected route is found to exist between the source and destination, establish the route. In this manner, we would effectively be using the data as route request messages.

These ideas we leave as future work, along with the refinement of the contact heuristic and further scenario experimentation.

REFERENCES

[1] J. Boice, J.J. Garcia-Luna Aceves, and K. Obraczka. On-demand routing in disruptive environments.

[2] J. Burgess, B. Gallagher, D. Jensen, and B.N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networking. In *Proceedings of IEEE Infocom*, 2006.

[3] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, R. Durst, K. Scott, and H. Weiss. Delay-tolerant networking: An approach to interplanetary research.

[4] B. Burns, O. Brock, and B. N. Levine. Mv routing and capacity building in disruption tolerant networks. In *Proceedings of IEEE INFOCOM*, 2005.

[5] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss. Delay-tolerant network architecture, 2004.

[6] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss. Delay-tolerant network architecture, 2005.

[7] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on the design of opportunistic forwarding algorithms. In *Proceedings of IEEE Infocom*, 2006.

[8] X. Chen and A.L. Murphy. Enabling disconnected transitive communication in mobile ad hoc networks. In *Workshop on Principles of Mobile Computing, colocated with PODC'01*, pages 21–27, 2001.

[9] T. Clausen and P. Jacquet. Rfc 3626: Optimized link state routing protocol (olsr), 2003.

[10] J.A. Davis, A.H. Fagg, and B.N. Levine. Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks. In *Proceedings of the International Symposium on Wearable Computing, Zurich, Switzerland, October 2001*, pages 141–148, 2001.

[11] R.M. de Moraes, H. Sadjadpour, and J.J. Garcia-Luna-Aceves. Throughput-delay analysis of mobile ad-hoc networks with a multi-copy relaying strategy. In *Proceedings of IEEE SECON*, 2004.

[12] DTNRG. Delay tolerant networking research group.

[13] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli. Age matters: Efficient route discovery in mobile ad hoc networks using encounter ages. In *Proceedings of Mobihoc*, 2003.

[14] K. Fall. A delay tolerant networking architecture for challenged internets. In *Proceedings of SIGCOMM*, 2003.

[15] K. Fall, W. Hong, and S. Madden. Custody transfer for reliable delivery in delay tolerant networks.

[16] K. Harras, K. Almeroth, and E. Belding-Royer. Delay tolerant mobile networks (dtmns): Controlled flooding schemes in sparse mobile networks. In *IFIP Networking*, 2005.

[17] T. Henderson, D. Kotz, and I. Abyzov. The changing usage of a mature campus-wide wireless network. In *Proceedings of the Tenth Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2004.

[18] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and the consequences of human mobility in conference environments. In *Proceedings of ACM Sigcomm (DTN Workshop)*, 2005.

[19] S. Jain, K. Fall, and S. Patra. Routing in a delay tolerant network. In *Proceedings of ACM SIGCOMM*, 2004.

[20] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[21] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proceedings of ASPLOS*, 2002.

[22] H. Jun, W. Zhao, M. Ammar, C. Lee, and E. Zegura. Trading latency for energy in wireless ad hoc networks using message ferrying. In *Third IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'05)*, 2005.

[23] J. Leguay, T. Friedman, and V. Conan. Dtn routing in a mobility pattern space. In *Proceedings of SIGCOMM 2005*, 2005.

[24] A. Lindgren, A. Doria, and O. Schelen. Probabilistic routing in intermittently connected networks. In *The First International Workshop on Service Assurance with Partial and Intermittent Resources (SAPIR 2004)*, 2004.

[25] S. Merugu, M. Ammar, and E. Zegura. Routing in space and time in networks with predictable mobility.

[26] M. Musolesi, S. Hailes, and C. Mascolo. Adaptive routing for intermittently connected mobile ad hoc networks. In *Proceedings of SIGCOMM 2005*, 2005.

[27] R. Ogier, F. Templin, and M. Lewis. Rfc 3684: Topology dissemination based on reverse-path forwarding (tbrpf), 2004.

[28] UCLA PCL. Global mobile information systems simulation library.

[29] C. Perkins and E. Belding-Royer. Rfc 3561: Ad hoc on-demand distance vector (aodv) routing, 2003.

[30] R. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: Modeling a three-tier architecture for sparse sensor networks.

[31] T. Small and Z. J. Haas. Resource and performance tradeoffs in delay-tolerant wireless networks. In *Proceedings of ACM SIGCOMM*, 2005.

[32] T. Spyropoulos, K. Psounis, and C. Raghavendra. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *Proceedings of SIGCOMM 2005*, 2005.

[33] W. Su, S. Lee, and M. Gerla. Mobility prediction and routing in ad hoc wireless networks, 2000.

[34] A. Vahdat and D. Becker. Epidemic routing for partially connected ad-hoc networks, 2000.

[35] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proceedings of ACM Mobihoc*, 2004.

[36] W. Zhao, M. Ammar, and E. Zegura. Controlling the mobility of multiple data transport ferries in a delay tolerant network. In *Proceedings of IEEE Infocom*, 2005.