

Virtual Repositories in Delay Tolerant Networks

PJ Dillon, Taieb Znati

Department of Computer Science, University of Pittsburgh

Abstract

The lack of contemporaneous end-to-end paths often characterize mobile computing environments, for which Delay Tolerant Networking is designed. In this work, we present a novel routing protocol using Virtual Repositories to assist in location-based forwarding when the location of a destination node is unknown. We compare the protocol to other existing protocols under two mobility models, Random Waypoint and Random Walk, and show its competitiveness while illustrating areas for further work and improvement.

1 Introduction

Delay Tolerant Networks (DTN) are an emerging technology for challenged networks where no end-to-end path may exist or one cannot be maintained. With no other options, the delay tolerant architecture can provide store, carry, and forward service in sparsely populated areas and in the face of frequent network partitions. In light of the new challenges of this paradigm, several areas of research have arisen, such as buffer management and caching, routing, security, and service provisioning.

DTNs facilitate scenarios requiring mobile devices to communicate in a peer-to-peer fashion over opportunistic connections. For instance, DakNet[14] and KioskNet[7, 15] have embedded wireless devices in public transportation that pick up and deliver messages to computer kiosks in rural areas of India and Cambodia. Also, infrastructure cannot easily be deployed in battlefield situations, which, while they would most likely have a time critical nature, may still benefit from DTN communication when nothing else is available. Lastly, a set of DTN capable mobile devices could collaborate to deliver typical text messages over relatively short distances, on the scale of a city, avoiding more costly means that utilize existing infrastructure.

The DTN architecture fits its niche but at some cost. It accounts for network partitions, needs little routing informa-

tion, and can be used when other solutions would usually fail. However, it's often extremely difficult or impossible to locate a given destination, predict when connections will be available, and message delays are typically unbounded and vary greatly, all of which contributes to greater demands on buffer space, necessitating an effort to managing storage and caching, and challenges many of the assumptions of the routing, transport, and application layers.

In this work, we present *Virtual Repositories* (VR) for Delay Tolerant Networking and demonstrate their use in location-based forwarding. A VR is an added geographic message destination towards which messages are routed when the actual location of the destination is unknown. A set of VRs is defined for a given network and distributed through its movement space where each VR in the set is assigned to one node in the network. We seek to determine *how well the protocol performs compared to other DTN routing protocols given a network of GPS-equipped mobile devices randomly moving in a defined, finite space.*

The rest of this paper is laid out as follows: Section 2 describes related work, Section 3 details the design of our Virtual Repositories framework and the routing protocol using it, Section 4 describes our evaluation against other existing protocols, Section 5 discusses areas of improvement and future work, and 6 concludes the paper.

2 Background and Related Work

Delay Tolerant Networking was first introduced by Kevin Fall in [4] for the Inter-Planetary Networking Research Group[2], but is now supported by the DTN research group[1]. In the five years since its publication, researchers applied the architecture to a wider range of networking scenarios than was originally intended, driving an evolution in its design, upon which Fall et al. reflect in [5].

2.1 DTN Routing

Routing in delay tolerant networks was first characterized by Jain et al. in [9]. The authors describe the space of available routing information and define a set *oracles* that provide this information to routing elements of the network, where each oracle provides more accurate information, which is consequently more difficult to obtain in usual circumstances. The simplest oracle provides aggregate information about specific links over time, such as the average time until the link becomes available. The most difficult provides accurate predictions about future buffer sizes and queuing delays. They then present several routing protocols capable of using each oracle to accomplish DTN routing.

Zhensheng Zhang surveyed DTN routing in 2006 [20], categorizing existing routing protocols and enumerating existing research topics. In general, routing solutions fall into two main classes: Deterministic and Stochastic. The former are either theoretical or applicable to situations where all of the nodes follow predetermined, predictable routes. Stochastic protocols make best effort forwarding decisions when little information is known about node movements, which is considered typical of human mobility patterns. In light of so little information, approaches randomly forward messages to other nodes, observe past history to discern more desirable future links from less desirable ones, or introduce entities into the network that have known, predictable mobility characteristics.

2.1.1 Random Techniques

Epidemic Routing[19] implements simple flooding. When two nodes meet, they synchronize message buffers to ensure they contain the same set of messages. This guarantees that the network finds the shortest path but comes at the cost of high demands on buffer space and bandwidth.

Conversely, Grossglauser and Tse showed in [6] that a single copy of a message only needed to be randomly forwarded to one other intermediate node to guarantee delivery to the destination. This assumes a given node has an equal chance of meeting any other node in the network, which over a long enough timeline gives a veritable certainty that the single intermediate node eventually comes in contact with the destination.

The routing algorithm of [6] is a special case of the Spray and Wait Protocol[17] by Spyropoulos et al., which distributes n message copies to n different nodes. These nodes then store the message and wait for direct contact with the destination node. The authors propose two distribution methods: one involving the source distributing all n

copies to n of its own neighbors and the second involving the source distributing half of its copies to its first neighbor and each successive contact receiving half of the remaining copies. The authors show that an ideal number of initial copies can be chosen to ensure certain performance constraints (maximum delay, energy consumption, minimum throughput, etc) given the assumption that every node has a chance of contacting any other node.

2.1.2 Utility-based Forwarding

Spray and Focus [18] is an improvement on Spray and Wait that first distributes (sprays) a fixed number of message copies to its neighbors and uses a utility-based scheme when a node has only a single copy left. A node determines the usefulness of a peer (its utility) by exchanging recorded timestamps of past contacts with its neighbors. Assuming recent contacts will likely meet again in the future, peers having contacted a desired destination recently are assumed to be suitable stewards of a message.

Similarly, PRoPHET[13] defines a probability $P(a, b)$ as the probability that two nodes, a and b , meet. The value is initialized to a configurable parameter when a and b first meet, is renewed with each successive contact, and decays over time. Also, the protocol implements a transitivity property such that $P(a, c)$ is defined when nodes a and c have never met but a has met b and b has met c . To accomplish this, each node disseminates a table of its calculated probabilities to its neighbors, who then use the information to update their own probability table.

2.1.3 Movement-Controlled Techniques

The DateMules Project[16] implemented special mobile entities, called Data Mules, that would move periodically between a sensor network deployment area and conveniently placed central data collection base stations. The mule would collect sensor data as it passed through the area and eventually deliver it to the base station well beyond the transmission range of any sensor. This provided significant power savings at the sensors, which were then absolved of finding routes and rebroadcasting data from other sensors.

In [21], Zhao et al. developed several scheme for collecting data among a set of worker nodes in a deployment area through the use of special nodes, called Message Ferries, that could courier data between nodes in the network. The Node-Initiated MF (NIMF) scheme sets the ferries on a predetermined path and requires any worker node to actively move towards the passing ferry to transfer its data, which

induces a trade-off between the node’s ability to accomplish its task and the delay imposed on the transferred data. The Ferry-Initiated MF (FIMF) scheme requires nodes to use a long range, low bandwidth radio to request data pick up from and ferry, which then actively moves to meet the node. In [22], the authors then consider the use of multiple ferries to improve the system throughput, which is shown to be effective under higher network loads. The authors consider travelling salesman solutions for single ferries, assigning groups of nodes to specific ferries, synchronizing ferry movements, and assigning specific routes to each ferry.

3 Virtual Repositories

Virtual Repository Routing facilitates location-based forwarding in DTN. Where location information is difficult to maintain across the network, Virtual Repositories provide static, easily computed physical locations towards which messages can be routed for a given destination.

A *Virtual Repository* (VR) is a circular area within the overall movement area of the network. Each VR is associated with the unique coordinate at its center, called its *home location* (we also refer to this as a VR location), and with exactly one destination node in the network. The radius of each VR is a system wide parameter. Also, the nodes within each VR are responsible for storing the messages bound for its associated destination.

With each message, in addition to typical source and destination identifiers, we associate a geographic coordinate towards which the message should be routed, called the *focus* of the message. The source of the message initially sets the focus, and any intermediate node along its path may alter it based upon newer location information for the destination of the message. When the location of the destination is unknown, the VR location of the destination is used by default.

The following subsections first describe the individual pieces of our design and then discuss how each is used in our routing algorithm.

3.1 Hash Function

A hash function associates each node to the home location of its VR. We assume a defined, rectangular movement area with a point of origin, (x_0, y_0) , a length, l , and a width, w that are globally known to all nodes. The hash function then uniformly maps an m -bit node identifier, id , into the continuous set of points defined by the rectangular area as

shown in Figure 1 where A is a constant, a good value for which is $(\sqrt{5} - 1)/2$ according to Knuth [11].

With a uniform distribution of the load sent to all possible destinations, the uniform distribution of VRs equally distributes the message storage burden across all nodes while significantly diminishing the number of message copies in the network compared to epidemic flooding.

The scheme also allows the hash space to be a subspace of the actual movement area if the nodes congregate in some areas or the typical flow of messages lends itself to a non-uniform VR distribution, for instance, when messages must be directed towards spatial bottlenecks to reach other parts of the network.

Lastly, beyond the coordinates returned from the hash function, the scheme requires no extra setup or storage to “create” and use each VR. Thus, when a node enters the network, its VR is already “created.” Furthermore, its VR persists when a node leaves the network or its user turns off the device.

3.2 Check-In Messages

Each node periodically generates a *check-in* message to inform its associated VR of its actual geographic location. Each check-in message contains this actual location and a creation timestamp. The focus of the message is set to the VR home location of the creating node. Thus, the check-in message carries the location of the node to the nodes in the its VR.

Given that the end-to-end delay from a node to its VR is unknown and varies greatly, check-in messages do not expire to ensure they eventually reach their VR. The creation timestamps order the series of check-in message originating from a node. If another node holding a check-in message receives a newer check-in message for the same VR, it replaces the older one with the newer.

Once a check-in message reaches its VR, a broadcast is initiated within the VR area, meaning any node holding the check-in message will transfer copies to its neighbors inside the VR, which ensures it eventually reaches all VR nodes. Upon reception, any stored messages are then sent towards the location contained within the check-in message.

3.3 Neighborhood Table

Each node maintains a *Neighborhood Table* of last known locations for other nodes in the network. Upon reception

$$H(id) = (x_h, y_h) = \begin{cases} x_h = H_x(id) = \lfloor w * (id * A - \lfloor id * A \rfloor) \rfloor + x_0 \\ y_h = H_y(id, x_h) = \lfloor l * (id * (x_h + 1) * A - \lfloor id * (x_h + 1) * A \rfloor) \rfloor + y_0 \end{cases}$$

Figure 1. Virtual Repository Hash Function

of a check-in message for some other destination node, the receiving node adds or updates the check-in location in its neighborhood table. And, after a given timeout, the entry is removed.

For data messages bound for the destination that subsequently come in contact with it, this node will first check its Neighborhood table for a matching entry for the destination. If found, instead of routing the message towards the VR of the destination, the message is sent towards the last check-in location of the destination. The effect is a short circuit for any message travelling towards the VR of the destination but first comes in contact with a node that has recently seen a check-in message for the same destination.

3.4 Performing a Look

A node most likely continues moving after the creation of a check-in message. Thus, the location contained in each check-in message loses accuracy as time advances. By the time a check-in message reaches its VR, the corresponding destination could be in a significantly different physical location. However, successive check-in messages with newer and newer location information are en route to the VR as well. Data messages travelling in the reverse direction encounter this newer information and are redirected accordingly. However, there is no guarantee of finding the destination, in which case the data messages should return to the VR for the destination.

For this reason, whenever a node learns of the location of a given destination and alters the focus of a corresponding data message, it sets a timeout in the message. If the message does not find the destination before the timeout, the node currently holding the message sets its focus to the home location of the VR. We call this process and time interval where a message is heading for the last known location of its destination a *look*.

The timeout duration is computed from the creation time of the check-in message that provided the location information. The time taken for a check-in message to get from its source to this node is a rough estimate of the time it will take for the data message to travel in the opposite direction. To account for the inaccuracy of this measure, the timeout is set to twice this time.

3.5 Receiving a Message

When a message is received and the receiving node is not the destination, the node processes the message and stores it in its buffer.

If it's a check-in message, the node extracts the sending node, the physical location information, and the creation timestamp from the message and creates or updates an appropriate entry in its Neighborhood Table. Also, if the node is inside the VR for the check-in message, the node finds all the messages in its buffer destined for the source of the check-in message and initiates a *look* for them using the check-in location.

If the message is a data message and not currently involved in a look, the node checks its Neighborhood Table for an entry corresponding to the destination of the message. If one is found, the focus of the message is set to the found location and a look is initiated.

3.6 Forwarding a Message

We use a greedy location-based forwarding scheme similar to that of LAR[12] for MANETs. A node decides to send a message to the peer physically closest to the destination of the message. If none of the node's neighbors are closer, it holds the message.

Neighbors may become the best forwarding option for a message some time after forming a connection with the node. For instance, a node, *a*, passing by the some other node, *b*, is first geographically (and for the sake of the argument) in front of *b* and then geographically behind *b* after passing, all while maintaining a connection. Thus, each node must have the means to track the physical location of each neighbor. We assume this functionality is implemented at a lower layer but note that routing decisions must be made upon each update to this information.

For a given message, *m*, a node checks its Neighborhood Table for a known location of *m*'s destination. If found, this location is used as the focus of the message instead of the message's actual focus. A look would have been initiated on *m* either when *m* was first received or when an appropriate check-in message was received. However, if another

node initiated a look for m , the current node uses the entry in its own Neighborhood Table to ensure newer routing information is used whenever a look is in progress.

With the given focus, f , a node then computes the distance between its own location and f and the distance between each of its neighbors and f . If its distance is less than the VR radius, meaning the node is inside the VR, a copy of m is sent to each neighbor whose distance to f is also less than the VR radius (broadcast to all inside the VR). If the node is outside VR of the message, the message is forwarded to a neighbor closer to f . Otherwise, the message is held.

3.7 Contact Protocol

Once a forwarding decision has been made, a node and its neighbor engage in a simple exchange protocol, which we assume can be piggy-backed upon lower level transmission packets. The sender informs the receiver of its intent to send a particular message. The receiver determines if the message is already stored in its buffer. It then ensures enough buffer space is available for the message. The transfer is rejected if either check fails. Otherwise, the transfer is accepted.

3.8 Leaving a VR

A special case occurs when a node moves out of the VR for which it is storing messages. It may be the case that the exiting node is the only node carrying these messages, in which case it should continue storing them in hopes of finding a node heading back towards the VR.

If, on the other hand, there are nodes that remain in the VR, the exiting node should be free to remove the messages from its buffer since it's no longer responsible for holding them.

Once outside the VR, the exiting node will enter the normal state of trying to forward these messages towards their VR. If a connection to peer inside the VR is still available, the peer is closer to the message focus and a forward will be attempted. Given that the peer already has the message, it denies the transfer with a special code that directs the exiting node to delete the message.

4 Evaluation

We compare several routing protocols using the Opportunistic Network Environment (ONE)[10]. This simulator currently simplifies the low level physical layer and link layer

Setting	Value
Simulation Time	12 h
Transmission Range	200m
Transmission Speed	2Mbps
Buffer Size	Max INT
Number of Nodes	50
Node Speed Interval	1-4m/s
Message Generation Interval	5-15s/msg

Table 1. Simulation Settings

simulation to improve simulation time while still providing a broadcast medium. The extremely long delays of DTN, on the order of anywhere from minutes to days, eclipse the milliseconds involved in CTS/RTS transmissions and connection establishment.

We demonstrate the performance of the VR protocol against Spray and Focus (SnF)[18], Prophet[13], and the Epidemic protocol with passive cure from [8].

SnF, Prophet, and Epidemic all require nodes to exchange information with their neighbors upon forming a connection (VR does not). SnF and Prophet exchange their respective measurements of delivery probabilities, and Epidemic exchanges lists of stored messages. To decrease simulation time, our implementation of Prophet and Epidemic implement this exchange in an out-of-band manner. This removes two extra forwards for each connection created over the duration of the simulation, consequently causing a reduction in the observed overhead for the two protocols.

Additionally, the cure protocol added to Epidemic, which floods a delivery notification for each delivered message through the network directing the deletion of the delivered message, is also implemented in an out-of-band manner such that it further reduces the observed overhead of Epidemic by one extra forward for every node. Our experiments showed that simulation trials with this version of the Epidemic protocol executed in about twenty minutes where trials with the traditional Epidemic protocol without a cure executed for seven days before running out of memory.

Overhead is defined as the difference between total number of forwards and the total number of delivered messages divided by the total number of delivered messages, which is, therefore, the number of forwards needed to deliver a message.

Each protocol was configured in a fashion consistent with its original paper. SnF was set to spray four copies of each message into the network, and the transitivity of its contact timers were adjusted for the mobility model (see [18]). Prophet was configured with an initial probability of 0.75, a transitivity value (β) of 0.25, an aging constant (γ) of 0.98,

and an aging interval of 10 seconds. VR was configured with a VR radius of 100 meters and a check-in interval of 5 minutes.

We performed two experiments described below with the configuration in Table 1. We, first, compare the protocols using a Random Waypoint mobility model and, then, a Random Walk model to govern node movements.

4.1 Comparison under Random Waypoint

The 50 nodes were placed in a square area, the dimensions for which were varied from $500m \times 500m$ to $4km \times 4km$ in increments of $500m \times 500m$. A Random Waypoint model governed node movement selecting speeds from the interval in Table 1 and wait times from the interval $[0, 120]s$. As the space grew, the nodes spread and the network becomes increasingly sparse. This results in fewer contact opportunities and an overall smaller potential throughput across the network. Thus, the limited communication becomes an increasingly more valuable resource to manage well.

4.1.1 Results

We had expected the accepted wisdom that Epidemic would demonstrate the best delivery ratio and have the smallest latency. At $500m \times 500m$, though, with a fully connected network¹, the broadcast medium seems to serialize the message flood such that it slows message propagation to each destination, allowing the single message copy in VR to reach the destinations with the smaller average latency shown in Figure 2(b) and despite the larger average hop count shown in Figure 2(d). The extra overhead of VR, shown in Figure 2(c) (the first two points not shown are 1050 and 255), corresponds to the check-in messages, which were not counted in the average hop count measurement and many of which may have been redundant in such a small space. Note, however, that our implementation of the other three protocols removes a large amount of overhead.

As the space grows and the network becomes more sparse, the latency generally increases for each protocol until its delivery ratio, shown in Figure 2(a), begins to decrease since the latency is only computed for delivered messages. At $1500m \times 1500m$, the extra air space benefits the latency of Epidemic, but the message load combined with its large amount of replication seem to impose significant queuing delays on the messages such that it still exceeds VR despite the growing hop count of VR. The overhead for the larger movement areas suggests, though, that VR trades a

larger number of data message forwards for the informational routing messages exchanged by each of the other protocols (most of which is not implemented and shown in Figure 2(c)).

The major disadvantage of VR is the steep drop in delivery ratio as the movement area increases beyond $2500m \times 2500m$ shown in Figure 2(a). While Epidemic and Spray and Focus remain statistically competitive, VR drops to the performance of Prophet. This suggests that the single copy, greedy location-based forwarding in use is ill-suited for extremely sparse networks where the series of forwards needed to transfer a message from source to destination never coincide with geographic straight line from source to destination.

4.2 Comparison under Random Walk

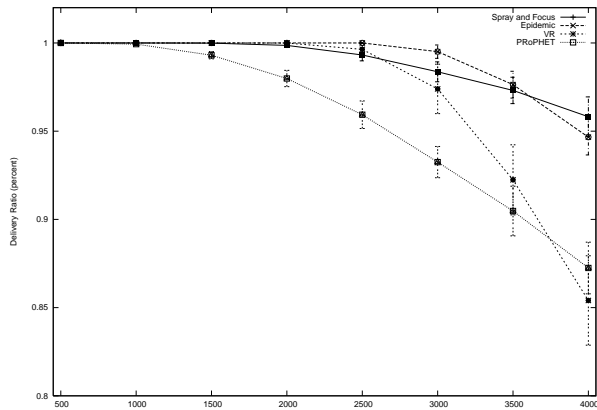
DTN routing relies heavily upon mobile nodes to physically carry messages to their destinations, which implies the results obtained above in Figure 2 are specific to the Random Waypoint mobility model. As such, we repeat the above experiment with the same configuration but using a Random Walk mobility model selecting node speeds from the same interval and wait times from the interval $[0, 30]s$. Where the nodes under Random Waypoint control move across the movement area fairly quickly and consequently display a high degree of intermingling, nodes under Random Walk move across the network slowly and have a smaller degree of intermingling[18]. Thus, the benefits of storing a message on a particular node to exploit its mobility dramatically decrease under Random Walk.

4.2.1 Results

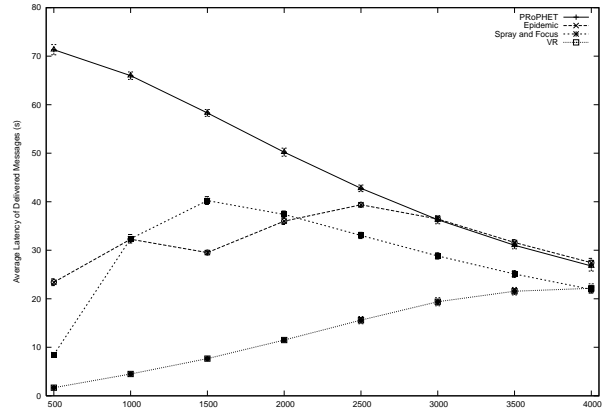
The results in Figure 3 show more of our original expectation. Epidemic seems to display the best delivery ratio in Figure 3(a) but not with much statistical significance. It, along with VR, also exhibits the smallest latency for denser networks and a statistically comparable latency for sparser networks.

The overhead of VR in Figure 3(c) is surprisingly small when compared with its counterpart in Figure 2(c) for dense networks. With nodes travelling shorter distances, they're more likely to remain inside a given VR and store its messages for a longer period of time. The overhead of the check-in messages depends upon the number of nodes generating them and the frequency with which they do so. As the number of delivered messages decreases with the increase in movement space, this fixed cost for our scenario is

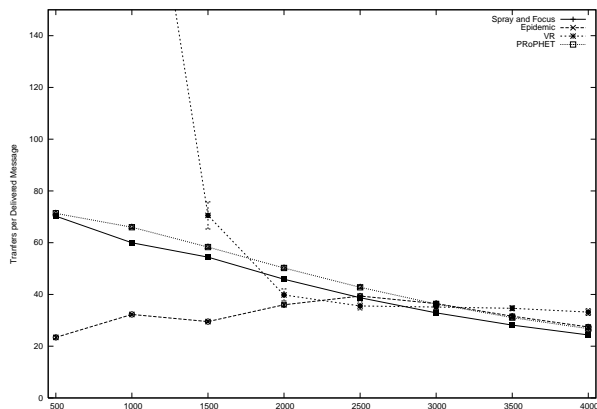
¹The simulation is likely to produce slightly unrealistic data in this case



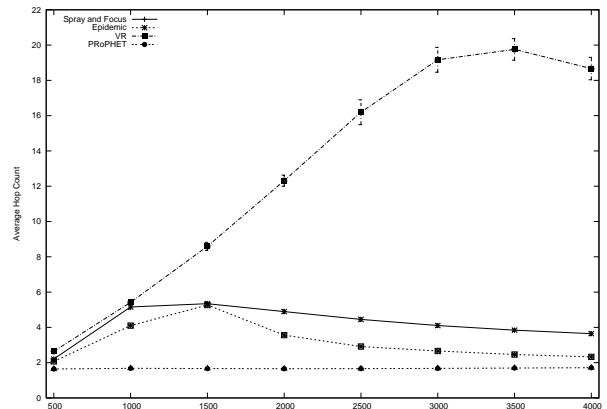
(a) Completion Ratio



(b) Latency of Delivered Messages



(c) Routing Overhead



(d) Average Hop Count of Delivered Messages

Figure 2. Increasing Movement Area with Random Waypoint Mobility

amortized over fewer and fewer delivered messages, exhibited by the upward trend for VR.

5 Future Work

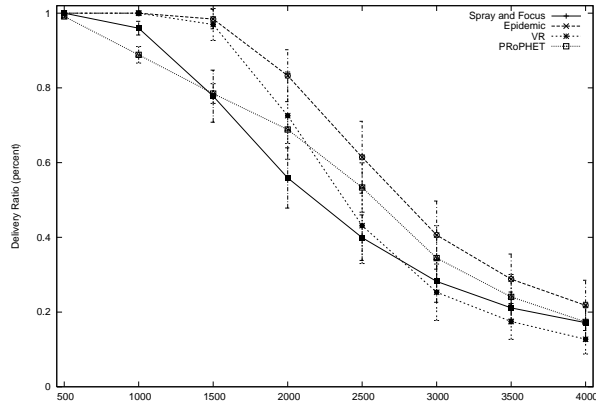
Our experimentation shows that the greedy forwarding protocol used with VR does not perform well when the network becomes extremely sparse. In addition, a case arises when two nodes moving in opposite directions come in contact. Initially, for the sake of argument, they are each in front of the other and would trade any message also heading in front of them. If other connections exist behind each node, forwarding may be the best option. If no connections exist behind them, though, each node will receive back the same messages once the nodes have passed one another, resulting in a number of superfluous forwards.

We intend to explore an epidemic distribution of node location information, similar to Spray and Focus and Proph

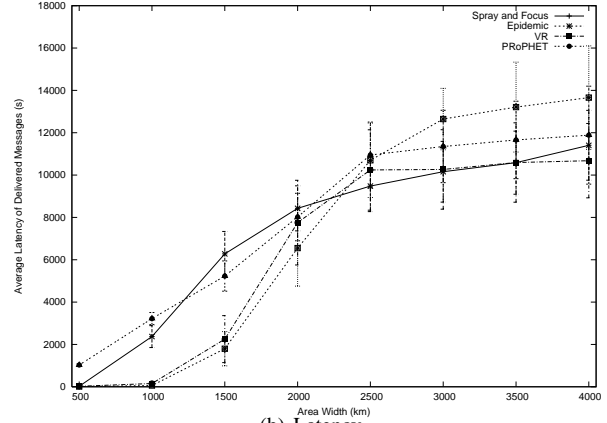
et, as a possible replacement for check-in messages. This work did not provide an adequate comparison of overhead between VR and the other protocols due to simplifications in their implementations. By creating a similar out-of-band simplification, the comparison would be more applicable and may potentially achieve better results.

Additionally, a simple improvement may be a multi-copy location-based forwarding scheme. For instance, a node could spray a fixed sized set of copies to the first n nodes encountered that are closer to the destination coordinate, each of which would then engage in single copy greedy location-based forwarding. The added redundancy improves reliability in the face of node failures and would force the network to find a disjoint set of paths for each message, which may vary greatly as time advances.

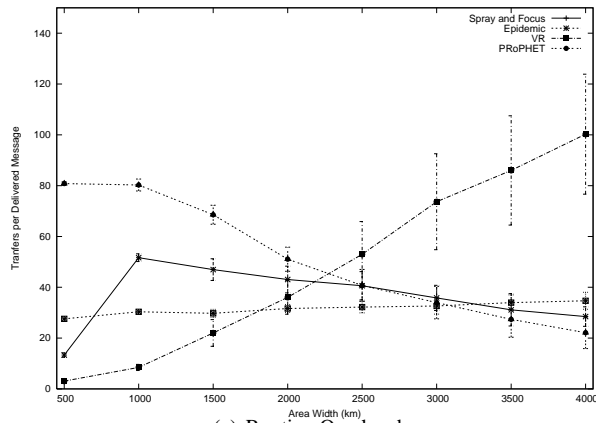
We also intend to explore more complex mobility models, such as community-based, cluster, and map-based models, such as that in [3]. With the dependence upon node mobil-



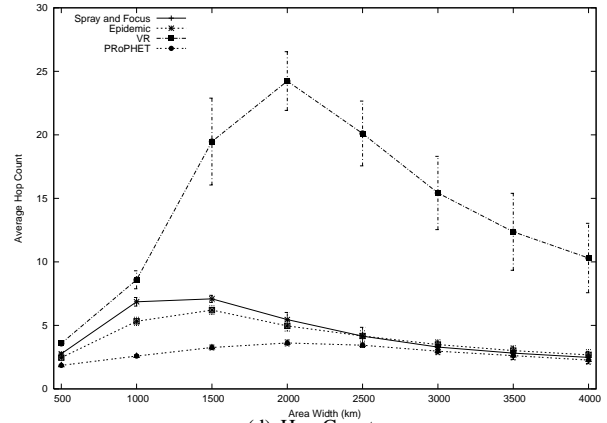
(a) Completion Ratio



(b) Latency



(c) Routing Overhead



(d) Hop Count

Figure 3. Increasing Movement Area with Random Walk Mobility

ity, any results obtained are only as realistic as the mobility model.

6 Conclusion

The recent rise in mobile computing introduces a set of new challenges in mobile network design particularly where a contemporaneous end-to-end path may never exist. Delay Tolerant Networking provides a store, carry, and forward paradigm for these scenarios, but losing the end-to-end path assumption requires researchers to revisit and redesign networking semantics.

In this work, we introduced a novel routing protocol employing geographic data to facilitate location-based forwarding in DTN. We assume difficulty in disseminating location information and create the concept of a Virtual Repository to add globally known message destinations to

the network when the location of the actual destination is unknown.

We show that, combined with a greedy location-based forwarding scheme, the protocol provides the smallest latency and extremely low overhead but demonstrates a greater sensitivity to node sparsity. This study illuminated several areas of possible improvement. And, given the comparable performance to existing protocols, it demonstrates a promising direction for further work.

References

- [1] Delay tolerant networking research group. <http://www.dtnrg.org>, 2007.
- [2] Inter-planetary networking research group. <http://www.ipnsig.org>, 2007.

- [3] F. Ekman, A. Keränen, J. Karvo, and J. Ott. Working day movement model. In *MobilityModels '08: Proceeding of the 1st ACM SIGMOBILE workshop on Mobility models*, pages 33–40, New York, NY, USA, 2008. ACM.
- [4] K. Fall. A delay-tolerant network architecture for challenged internets. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 27–34, New York, NY, USA, 2003. ACM Press.
- [5] K. Fall and S. Farrell. Dtn: An architectural retrospective. *IEEE Journal on Selected Areas in Communications*, 26(5):828–836, June 2008.
- [6] M. Grossglauser and D. N. C. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Transactions on Networking*, 10(4):477–486, 2002.
- [7] S. Guo, M. Falaki, E. Oliver, S. Rahman, A. Seth, M. Zaharia, U. Ismail, and S. Keshav. Design and Implementation of the KioskNet System. In *Proceedings of the International Conference on Information and Communication Technologies and Development (ICTD 2007)*, pages 300–309, 2007.
- [8] K. A. Harras, K. C. Almeroth, and E. M. Belding-Royer. Delay tolerant mobile networks (dtmns): Controlled flooding in sparse mobile networks. *IFIP Networking*, May 2005.
- [9] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. *SIGCOMM Comput. Commun. Rev.*, 34(4):145–158, 2004.
- [10] A. Keränen, J. Ott, and T. Kärkkäinen. The one simulator for dtn protocol evaluation. In *SIMUTools '09: Proceeding of the 2nd International Conference on Simulation Tools and Techniques*, New York, NY, USA, 2009. ACM.
- [11] D. E. Knuth. *The Art of Computer Programming*, volume 3. Addison-Wesley Professional, 1998.
- [12] Y.-B. Ko and N. H. Vaidya. Location-aided routing (lar) in mobile ad hoc networks. In *MobiCom '98: Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*, pages 66–75, New York, NY, USA, 1998. ACM Press.
- [13] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):19–20, 2003.
- [14] A. S. Pentland, R. Fletcher, and A. Hasson. Daknet: Rethinking connectivity in developing nations. *Computer*, 37(1):78–83, 2004.
- [15] A. Seth, D. Kroeker, M. Zaharia, S. Guo, and S. Keshav. Low-cost communication for rural internet kiosks using mechanical backhaul. In *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*, pages 334–345, New York, NY, USA, 2006. ACM.
- [16] R. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: modeling a three-tier architecture for sparse sensor networks. In *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications*, 2003.
- [17] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259, New York, NY, USA, 2005. ACM Press.
- [18] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility. *IEEE International Conference on Pervasive Computing and Communications Workshops*, 0:79–85, 2007.
- [19] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke University, April 2000.
- [20] Z. Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. *Communications Surveys and Tutorials, IEEE*, 8(1):24–37, Quarter 2006.
- [21] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MobiHoc '04: Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 187–198, New York, NY, USA, 2004. ACM Press.
- [22] W. Zhao, M. Ammar, and E. Zegura. Controlling the mobility of multiple data transport ferries in a delay-tolerant network. In *INFOCOM 2005. Proceedings of 24th Annual Joint Conference of the IEE Computer and Communications Societies*, 2005.