Performance Analysis of Routing Protocols in Delay/Disruption Tolerant Mobile Ad Hoc Networks

Fuad Alnajjar[1] and Tarek Saadawi[2]
The City College and Graduate Center of City University of New York
Electrical Engineering Department,
138th street and Convent Avenue,
New York, NY 10031
USA
[1]fuad@ccny.cuny.edu
[2]saadawi@ccny.cuny.edu
http://www-ee.ccny.cuny.edu

Abstract: - In the last few years research activity in delay/disruption tolerant networks (DTN) is growing and researchers have proposed various types of routing protocols. Those efforts formulate DTN to become the adequate solution for the challenged network environment. DTN architecture provides good performance in the intermittently connected Mobile ad hoc networks (MANET). Routing in DTN architecture is the key challenge because of the nature of MANET environment where the network is an opportunistic connected and topology is changing rapidly.
In this article we analyze the performance of DTN-based routing protocols including our routing approach, History of Encounters Probabilistic Routing Algorithm (HEPRA) in terms of different aspects. We select well-known DTN routing protocols in our evaluation to demonstrate how those protocols act comparing to our approach, HEPRA. We continue developing our algorithm, HEPRA, to provide a detailed analytical as well as simulation-based study. Using simulation we considered in our analysis various factors such as number of nodes, buffer size, speed of nodes, time to live (TTL), movement models and transmission range.

Key-Words: - DTN, Routing protocol, MANET, ONE, Performance Analysis.

1. Introduction

Delay Tolerant Networking (DTN) is an end-to-end network architecture designed to provide communication in and/or through highly stressed networking environments. Stressed networking environments include those with intermittent connectivity, large and/or variable delays, and high bit error rates. Recently, the term disruption-tolerant networking is frequently used instead of Delay-tolerant due to the support from Defense Advanced Research Projects Agency (DARPA). Disruption may occur because of nodes sparsity, radio transmission range, energy resources, attack, and noise. DTN architecture seeks to address the technical issues in the heterogeneous networks that may lack continuous network connectivity.

The DTN Research Group (DTNRG) leads the field in DTN research. Members of the DTNRG created the Bundle Protocol (BP) to implement the DTN architecture. The key capabilities of the bundle protocols include custody-based reliability, ability to cope with intermittent connectivity, ability to take advantage of scheduled and opportunistic connectivity, and late binding of names to addresses.

As an effort to standardize communications for the Interplanetary Internet (IPN), the Delay-Tolerant Networking architecture and protocols were proposed. ('DTN architecture and protocols were proposed as an effort to standardize communications for the IPN'). As work progressed, researchers observed that military networks running tactical protocols, and remote networks where network resources are scarce and data mules might be used to transport data. These networks all had similarities in that they experienced several of these features: asymmetric communication, noisy links, long delays, and intermittent connectivity. As a result, the network community is developing a body of research for which funding has been established by both NASA and DARPA. [1-4].

The remainder of this article is organized as follows: Section 2 presents additional background information and an overall design work of our algorithm. Section 3 reviews routing in DTN and presents common routing protocols in DTN. Section 4, describes the HEPRA routing algorithm design. Section 5 proposes the Time-Slots in HEPRA. Section 6 presents simulation tools. Section 7 presents Behavior of Time-Slot HEPRA (TS-HEPRA). Section 8 summarizes and discusses the results of the evaluation of DTN routing protocols. Finally, Section 9 discusses our conclusion.

2. Background

A Mobile Ad hoc Network (MANET) is a dynamic wireless network with or without fixed infrastructure. Nodes may move freely and arrange themselves randomly. The contacts between nodes in the network do not occur very frequently. As a result, the network graph is rarely, if ever, connected and message delivery must be delay-tolerant.

Traditional MANET routing protocols such as DSR, AODV and OLSR requires that the network graph is fully connected and fail to route messages if there is not a complete route from source to destination at the time of sending. For this reason traditional ad hoc routing protocols cannot be used in environments with intermittent connectivity. [5,6].

To defeat this issue, node mobility is exploited to physically carry messages between disconnected parts of the network. Schemes like these designs are occasionally referred to as Mobility Assisted Routing (MAR) that employs the store, carry and forward model. In store, carry-and- forward networking model messages are forwarded between a set of nodes. When a node receives a message, it determines how to route the message further, and then determines whether or not it has connectivity to the chosen next hop destination(s). If it does, the message is forwarded onward. However if it does not have connectivity, instead of dropping the message, it will store it until the connectivity becomes available, so that when the network becomes available, the forwarding operation is resumed. [7]

Figure 1 shows how the mobility of nodes in such circumstances can be employed to ultimately deliver a message to its destination. In this figure, node A has a message (indicated by the node being sky blue) to be delivered to node F, but a path does not exist between nodes A and F. As shown in figures (a-d), the mobility of the nodes let the message be transferred to node B (fig b), then to node E (fig c), and finally, when node E moves within range of node F to node F which is its final destination.[8],[9].

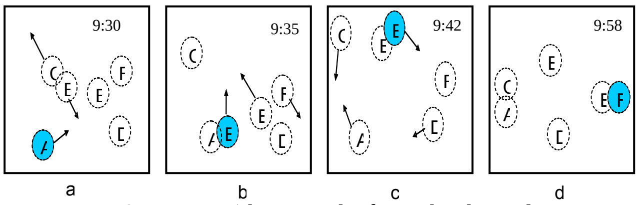


Figure 1. A message (shown in the figure by the node carrying the message being sky blue) is moved from node A to node F via nodes B and E utilizing the mobility of nodes showing the time [8]

Routing in mobile ad hoc networks (MANET) is difficult because the network graph is episodically connected. The topology is changing rapidly because of weather, terrain, highly variable delay links, error rate links, and jamming. A key challenge is to create a technique that can present good delivery performance and low end-to-end delay in an intermittent network graph and opportunistic or scheduled intermittent links where nodes may move freely. DTN is a message-based store, carry-and-forward overlay network architecture. Delay-Tolerant Networking (DTN) architecture is designed to provide communication in intermittently connected networks by moving messages towards destination via store, carry-and-forward' networking model that supports multi-routing algorithms to acquire best path towards destination. [10]

The article presents a detailed analytical as well as simulation-based study of our published DTN-based routing protocol, History of Encounters Probabilistic Routing Algorithm (HEPRA) [11]. We analyze the performance of HEPRA and common routing protocols. Epidemic, PROPHET, Spray and Wait, Maxprop. Using simulation we considered in our performance's analysis various factors such as number of nodes, buffer size, speed of nodes, time to live (TTL), transmission range and movement models.

3. Routing in DTN

In this section we review routing in DTN and presents common DTN routing protocols.

Vahdat and Becker [12] presented a routing protocol called Epidemic. Epidemic routing is flooding-based in nature, since nodes continuously replicate and transmit messages to newly discovered nodes that do not already possess a copy of the message. It utilized the theory of epidemic algorithm to ultimately deliver messages to their destination when nodes encounter each other by doing random pair-wise information of messages between the encountered nodes. If bath to destination is not accessible, the node will buffer the messages in index called summary vector. Each node maintains a buffer consisting of messages that it has originated in addition to messages that it is buffering on behalf of other hosts. Once two nodes meet they exchange the summary vectors. If the node finds any new messages, it requests them from the encountered node. This mechanism of swapping new messages continues as long as buffer space is available, and messages will spread similar to an epidemic of some diseases inside the network whenever infected node meets susceptible node, a copy is forwarded (flooding). In order to avoid duplicate messages during the exchange process each message has a globally unique message ID. Each message contains source and destination addresses. Also, to lower the utilization of nodes resources, each message has a hop counter to determine the maximum number of hops a message can travel to.

In [13] Anders Lindgren and et al presented a Probabilistic routing algorithm called PROPHET. PROPHET stands for Probabilistic ROuting Protocol using History of Encounters and Transitivity. Authors established a probabilistic metric called delivery predictability $P_{(a,b)} \in [0,1]$ at every node a for each known destination b. The procedure of PROPHET is like the Epidemic Routing, in which, two nodes exchange summary vectors when they meet. In addition to that, in PROPHET, it contains the delivery predictability information stored at the nodes. This information is used to update the internal delivery predictability vector and then the information in the summary vector is used to decide which messages to request from the other node. The forwarding strategy depends on the delivery predictability of the encountered nodes. If node a meets node b, a carried message destined for node m will be transferred from a to b only if $P_{(b,m)} > P_{(a,m)}$. PROPHET algorithm relies on calculation of delivery predictability to forward messages to the reliable node. The probability is used to decide if one node is more reliable than the other to forward message to the destination node. It includes three parts about the probability. First is to update the probability metric whenever a node is encountered, the node that is frequently encountered having higher delivery predictability than others. Second, if a pair of nodes do not encounter each others during an interval, they are less likely to be good forwarders of messages to each other, thus the delivery predictability values must be reduced. Third, there is a transitive property in delivery predictability. Based on the observation, if node a frequently encounters node b, and node b frequently encounters node c, then node c probably is a good node to forward messages destined for node a.

Thrasyvoulos Spyropoulos and et al proposed Spray and Wait protocol. Spray and Wait has phases: 1) spray phase and 2) wait phase. When a new message is created in the network, a number M is attached to the message indicating to the maximum allowable copies of the message in the network. In the first phase, spray, the originate node of the message is responsible for spraying, one copy to M intermediate nodes. When the intermediate node receives the copy, it go into the second phase, wait, where the intermediate node buffer that particular message until the destination is encountered directly. [14].

In [15] John Burgess and et al presented a routing protocol uses flooding technique called MaxProp. In MaxProp If a new node discovered, new messages to the node will attempt to be replicated and transferred. MaxProp determines first which messages should be transmitted and or dropped. It maintains an ordered-queue based on the message's destination, and it ordered by the estimated likelihood of the future path to that destination. Path likelihoods estimated by each node in which is maintaining a vector of size $n-1$, where n is the number of nodes in the network, consisting of the likelihood the node has of encountering each of the other nodes in the network. Encountered nodes exchange their estimated node-meeting likelihood vectors when they meet. The vectors are kept updated by every node. Each node can compute a shortest path via a depth-first search where path weights indicate the probability that the link does not occur. Path weights are added to determine the total path cost, and are computed over all possible paths to the desired destinations. The cost for any destination is determined by selecting the path with the least total weight. Then, messages are ordered by destination costs, and transmitted and or dropped in that order.

Authors in [16] proposed RAPID, Resource Allocation Protocol for Intentional DTN. RAPID can optimize a specific routing metric such as worst-case delivery delay is delivered within a deadline. It translates the routing metric into per-packet utilities which determine how packets should be replicated in the system. Each packet in the network will assign a utility function to every packet , which is based on the metric being optimized. RAPID replicates packets first that locally result in the highest increase in utility. Therefore, the protocol replicates the packet that results in the greatest decrease in delay. RAPID, like MaxProp, is flooding-based, and will therefore attempt to replicate all packets if network resources allow.

Michael Demmer and Kevin Fall presented DTLSR, Delay Tolerant Link State Routing, in which is modeled on classic link state algorithms. DTLSR works similarly as OSPF. When the network state changes, link state announcements are flooded in the network. Nodes maintain a graph representing their current view of the state of the network, and use a shortest path computation to find routes for messages. Each node in the system is assigned to an administrative area, and a link state protocol operates only within a single area. Nodes that have neighbors in other areas learn the set of endpoint identifiers reachable via the other area and announce themselves as a gateway to those endpoint identifiers. [17].

Paolo Costa and et al in [18] SocialCast, an interest-based routing protocol to support delay tolerant communication in human networks. Authors assumed that socially bound hosts are likely to be co-located regularly: The collocation patterns are used to efficiently route the messages from publishers to interested subscribers. The social ties selection is made by taking into account predictions about contextual parameters such as mobility patterns based on previous observations.

4. HEPRA: History of Encounters Probabilistic Routing Algorithm

Delay tolerant networks have been proposed to address data intermittent communication challenges in networks where an instantaneous end-to-end path between a source and destination may not exist, and the links between nodes may be opportunistic, predictably connectable, or periodically-(dis)connected [19].

In this article, we continue our development in our DTN-based protocol, History of Encounters Probabilistic Routing Algorithm (HEPRA), and present more details on the operation of it. HEPRA designed to maximize message delivery rate, minimize the total resources consumed in message delivery, minimize the number of hops used in routing and minimize message latency. We focus on the Delay-Tolerant Mobile Ad Hoc Network to design a probabilistic routing protocol applicable to work in this intermittently connected environment to improve the end-to-end message delivery ratio in a multihop scenario where link availability can be low. The operation of HEPRA relies on the knowledge of the mobility of nodes to forward messages based on encountered nodes in the past. HEPRA utilize history on encountered nodes for forwarding strategy. Messages will be transferred towards destination via 'store, carry and forward' technique that is used in DTN based routing protocols.

HEPRA uses the history of encountered nodes to predict its future suitability to deliver messages to next node toward destination. An index of encountered nodes called a summary vector is kept by each node. Each Node maintains the summary vector that lists all encountered nodes during its mobility. The buffer size of each node controls the size of the summary vector. The information in the summary vector is used to decide which messages to be requested from the other node based on the History of encounters factor used in the forwarding strategy. Our forwarding strategy depends on the History of encounters of nodes in the network. We create a metric called History of encounters at every node. This indicates how highly-encountered the node is, which the number of nodes encountered till that moment is. The calculation of messages delivery depends on the History of encounters metric. Figure 2 defines the necessary variables while Figure 3 contains the pseudo-code for HEPRA routing protocol.

When two nodes meet, the first thing to do is to update the metric (increase the metric by one), then they swap the number of encountered nodes till moment of meeting so that nodes that are often encountered more nodes have a high delivery Probability. Encountered nodes exchange only the number of earlier contacts without any details of those nodes. If they met the same number of nodes in the past they exchange new messages and if one of them encountered more nodes than the other in the past, only the node with low number of earlier contacts will deliver the new messages to the node with high earlier contacts. When a message arrives at a node, there might not be a path to the destination available so the node has to buffer the message. Upon each encounter with another node, a decision must be made on whether or not to transfer that particular message.

Variables:

1. *Seconds in time unit:* The unit in seconds for the hourly representation of the *sim clock*
2. *Duration*: The unit in seconds for the latest sociality information for the latest simulation time
3. *KnownHostsMap*: The Map having the current *sim clock* as key and the *other host* as the value.
4. *noofHosts:* The counter that counts the number of hosts.

Figure 2. Pseudo-code definitions of HEPRA

```
1 .Getknownhosts: The method that returns the knownHostsmap for a particular host.
2. ChangedConnection:
{       The connection given as input
        if (connection is up for a node)
        then
        otherHost = the host of the other node
        in knownHostsMap put (SimClock, otherHost)
}
3. update:
{       checks for the transferring condition
        if noof messages ==0 or connections size ==0
        then comes out of the loop
        if (there are deliverable messages?)
        [get messages for connected ()]
        Try messages for the connected {start transfer}
        If there is connection, return the connection
        Call tryothermessages()
}

4. Knowledge calculator:
{     Integer S = knownhostsmap key() i.e simclock
      Initialize noofHosts = 0;
      Iterator i
      while(i.hasNext())
       if (i.next() >= (SimClock()- duration))
      then
      increment noofhosts
      return noofHosts;
   }
5. Tryothermessages
      {
      Otherhost =  get the other host
      Otherrouter =  get the router from other host
      If this router noofhosts < other router noofhosts
      Then continue
      For all the messages
      Check for unseen messages
      Add.messages
      }
```

Figure 3. Pseudo-code of HEPRA routing protocol

As example, When node i meets node j they update the summary vector and. then they exchange the summary vector. Each node will check the History of Encounters metric of each other. If the history of encounters metric of node i is less than node j, node i will transfer any unseen messages to j but not vice versa. Node i will deliver messages to destinations if path to destination available, otherwise, it will store the messages in the buffer and continue mobility till encountering new node. Employing the concept of s history of encounters factor increases the probability of delivering messages to intermediate nodes and destinations since the probability of delivering messages by highly encountered -connected nodes is higher than lower encountered connected nodes. HEPRA utilizes information about the earlier contacts to predict how good nominee a node is to deliver the message to the recipient. In HEPRA, messages carried by the node with a higher probability, based on the history of encounters condition, only are transferred.

5. Time-Slots in HEPRA

We develop HEPRA by employing a metric called SimClock to determine the time and number of encountered nodes at anytime in the simulation. In [11] HEPRA counts encountered nodes during the mobility of the nodes throughout the simulation time. We used in our simulation the Opportunistic Network Environment simulator (ONE-V1.3). [20]. The time simulation set to 12 hours so HEPRA count number of encountered nodes in this time of simulation. In this work, we evaluate the performance of HEPRA by dividing the simulation time to 12 time-slots. Each time-slot equals to 1 hour. We monitor performance of HEPRA in terms of delivery rate, overhead, latency, buffer size, number of hops. Figure 4 defines the parameters use in the evaluation and analysis. The performance of HEPRA was consistent with increasing the time-slot. The results were expected since the additional nodes encountered will increase the delivery probability, minimize latency and increase overhead.

Created messages: number of messages created during simulation
Started messages: number of messages whose transmission was initiated between network nodes
Relayed messages: number of messages successfully transmitted between network nodes
Dropped: number of messages dropped from nodes buffers because of full buffer
Removed: number of messages removed from nodes buffers because it was delivered to final destination.

Delivered messages: number of messages successfully delivered during simulation

$$\text{Delivered messages} = \frac{\text{Delivered messages}}{\text{Started messages}} =$$

overall message delivery %

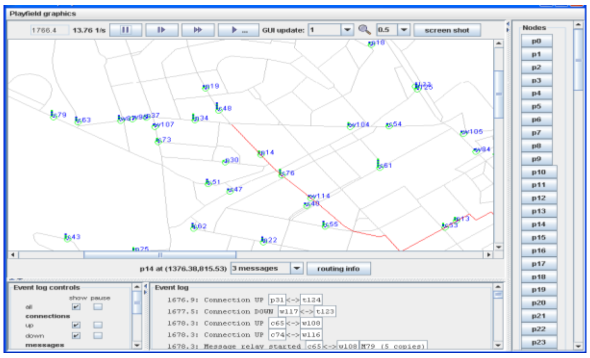
$$\text{Overhead} = \frac{\text{Relayed messages - Delivered m}}{\text{Delivered messages}}$$

Latency: overall message average delay (average time between messages creation and delivery)
hopcount: average number of hop counts between the source node and the destination node
buffertime: how long messages stay in the message buffer from receiving/creating them until they're dropped or removed

Figure 4. Parameters used in evaluation in the simulation. [20].

6. Simulation Tools

The Opportunistic Network Environment simulator (ONE-V1.3) is created by A. Keranen and J. Ott [20] to address the routing in DTN environment. It provides a powerful tool for generating mobility traces, running DTN messaging simulations with different routing protocols, and visualizing simulations interactively in real-time and results after their completion. ONE-V1.3 was used in our simulation since it includes different routing protocols such as Epidemic, Spray and Wait, PRoPHET and MaxProp. Figure 5 shows a screenshot of ONE simulator.



```
1676.9: Connection UP p31<->t124
1677.5: Connection DOWN w117<->t123
1678.3: Connection UP c65<->w108
1678.3: Connection UP c74<->w116
1678.3: Message relay started c65<->w108 M79 (5 copies)
```

Figure 5. Simulator ONE Screenshot

7. Behavior of Time-Slot HEPRA (TS-HEPRA)

Table 1 shows the simulation setup used in our model. To capture this type of behavior, the model is based on dividing the simulation time into time-slots. We evaluate the performance of HEPRA by changing the mechanism of the routing and use Time-Slot technique instead of the whole time of simulation. Now, when two nodes meet they update the summary vector and then they exchange the summary vector. Each node will check the History of Encounters metric of each other. We monitor the history of encounters metric of each node in the last time-slot. The 12 hours simulation time was divided into 12 time-slots.

Table 1. Simulation parameters for TS-HEPRA

ENVIRONMENT PARAMETERS	VALUE
Simulation Area (W x H) meter	4500 x 3400
Simulation duration (hr)	12
Number of nodes	50
Movement Model	Shortest Path Map Based Movement
Message TTL (minutes)	60
Host speed (m/s)	0.5 -1.5
Buffer size (Mbyte)	10

Results in figures 6.1- 6.5 illustrate how the TS feature works in HEPRA. We evaluate the performance in term of delivery rate, overhead, latency, hopcount, buffertime. Figure 6.1 illustrate that TS-HEPRA deliver more messages when TS increases since number of encountered nodes will be increases by time. Figure 6.2 shows that overhead (defined in figure 4) increases since number of relayed and delivered increase. Latency in figure 6.3 decreases when TS increases of number of encountered nodes increase. Figure 6.4 illustrate that hopcount reduces when TS increase since the amount of encountered nodes increase so nodes exchange more messages. In figure 6.5 buffertime increases because nodes buffer more messages.

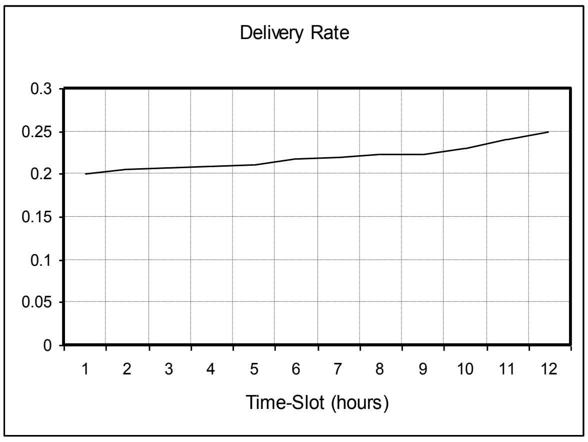


Figure 6.1 Delivery Rate increases when Time-Slot (TS) increases

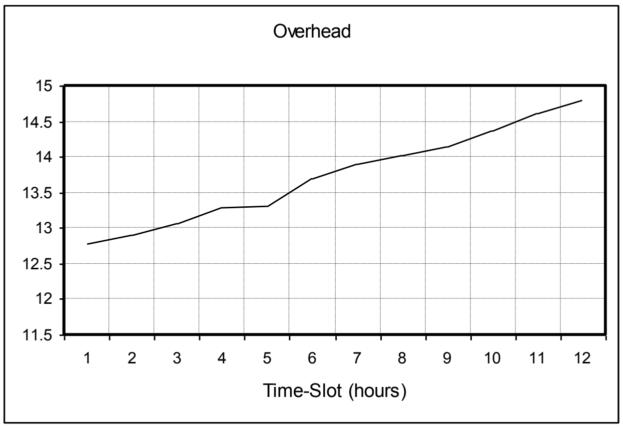


Figure 6.2 Overhead increase when TS increases

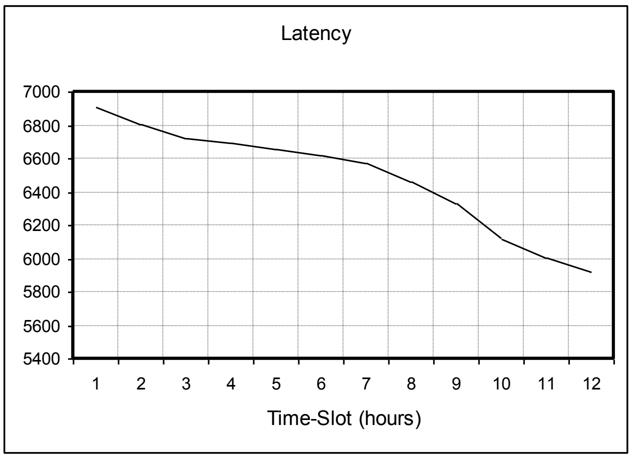


Figure 6.3 Latency decreases when TS increases

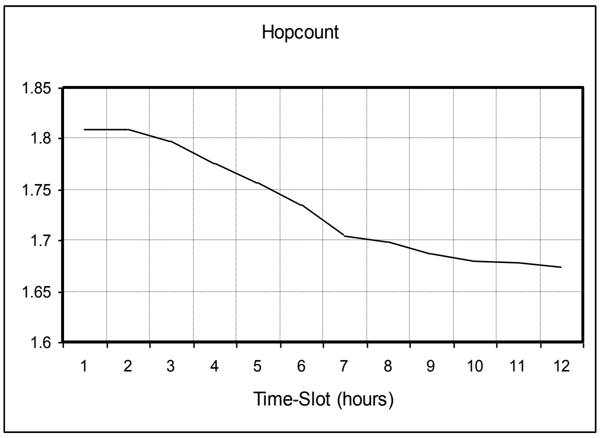


Figure 6.4 Hopcount decreases when TS increases

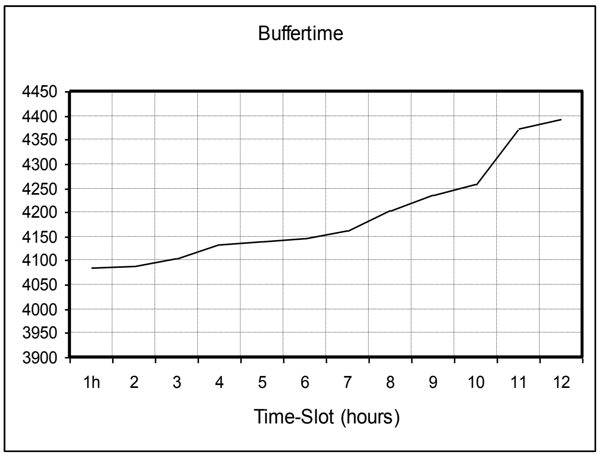


Figure 6.5 Buffertime increases when TS increases

8. Evaluation of DTN routing protocols

In this section we describe the simulation setup used to evaluate the performance of DTN Routing Protocols. We select Epidemic, PROPHET, Spray and Wait (SnW), MaxProp and HEPRA since the first four protocols are included in the ONE simulator. Simulation settings is demonstrated table 2.

Table 2. Simulation setting used in the evaluation

PARAMETERS	VALUE
Simulation Area (W x H) meter	4500 x 3400
Simulation duration (hr)	12
Number of nodes	50
Movement Model	Shortest Path Map Based Movement Random Waypoint Map Based Movement
Message TTL (minutes)	60-600
Host speed (m/s)	0.5 -1.5
Buffer size (Mbyte)	5-500
Transmission Range (meter)	1-50

Simulation results are organized in table 3. The table illustrates the good and weak performances in terms of different behaviors such as buffer size, speed of nodes, number of nodes, transmission range, Movement models, and packet time to live. In general, MaxProp and Spray and wait came in first place in terms of the pervious behavior, HEPRA in the middle and Epidemic in the last.

Table 3. Simulation setting used in the evaluation

Behavior	***Good performance***	***Weak performance***
<u>Buffer size</u> ***Delivery Rate*** ***Overhead*** ***Latency***	SnW & MaxProp SnW SnW	HEPRA Epidemic PROPHET
<u>Speed</u> ***Delivery Rate*** ***Overhead*** ***Latency***	MaxProp HEPRA SnW	PROPHET & Epidemic Epidemic & MaxProp MaxProp
<u>Number of nodes</u> ***Delivery Rate*** ***Overhead*** ***Latency***	MaxProp SnW HEPRA	Epidemic Epidemic MaxProp & PROPHET
<u>Transmission range</u> ***Delivery Rate*** ***Overhead*** ***Latency***	MaxProp & SnW HEPRA SnW	Epidemic Epidemic & MaxProp MaxProp
<u>Shortest Path Movement</u> ***Delivery Rate*** ***Overhead*** ***Latency***	MaxProp HEPRA SnW	Epidemic Epidemic MaxProp
<u>Random Way Movement</u> ***Delivery Rate*** ***Overhead*** ***Latency***	SnW HEPRA HEPRA & Epidemic	Epidemic Epidemic MaxProp
<u>Map Based Movement</u> ***Delivery Rate*** ***Overhead*** ***Latency***	PROPHET HEPRA SnW	Epidemic MaxProp & Epidemic MaxProp
<u>Time To Live (TTL)</u> ***Delivery Rate*** ***Overhead*** ***Latency***	MaxProp SnW SnW	Epidemic Epidemic MaxProp

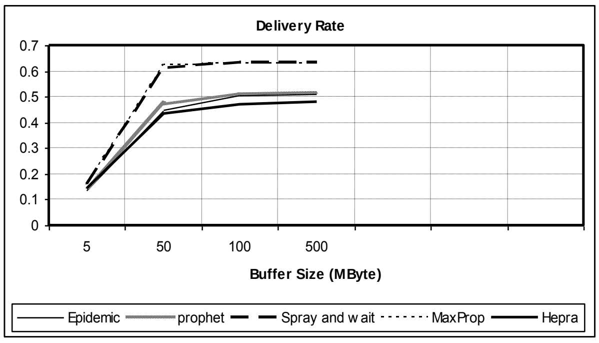


Figure 7.1 SnW and MaxProp delivers more messages when buffersize increases

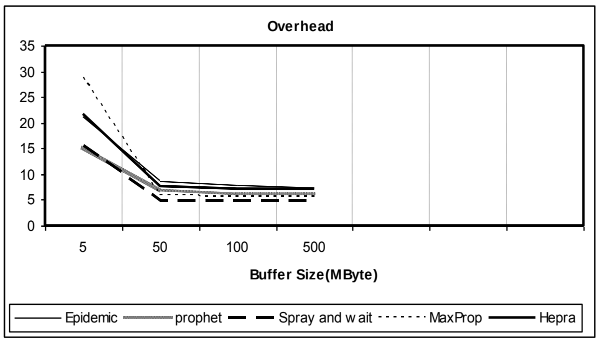


Figure 7.2 Overhead of Epidemic is higher than others

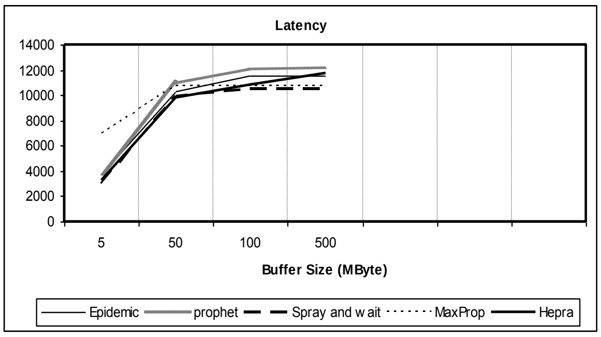


Figure 7.3. Latency of PROPHET has more delay

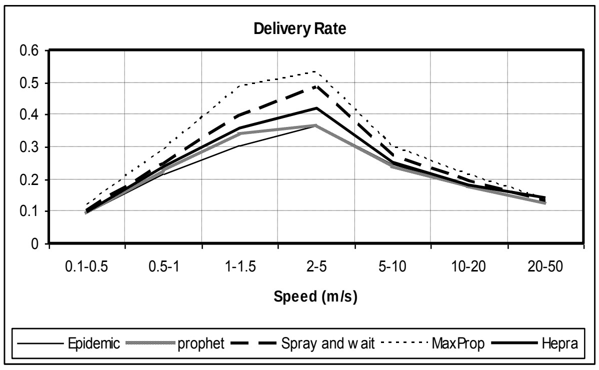


Figure 8.1. MaxProp deliver messages

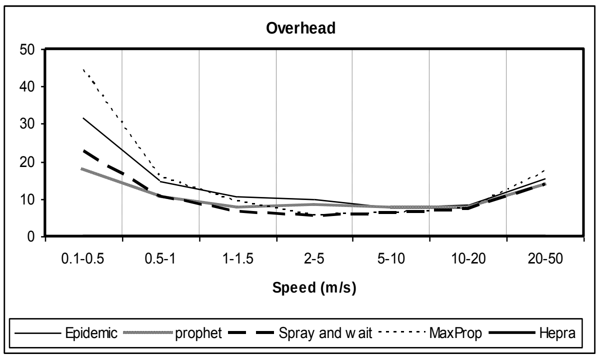


Figure 8.2. Overhead of Epidemic and MaxProp is higher than others

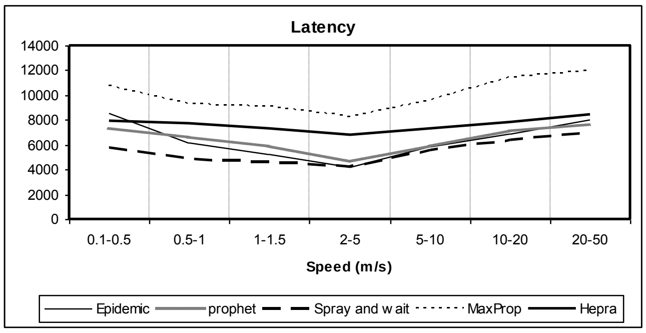


Figure 8.3. MaxProp has the highest average Latency when speed increases

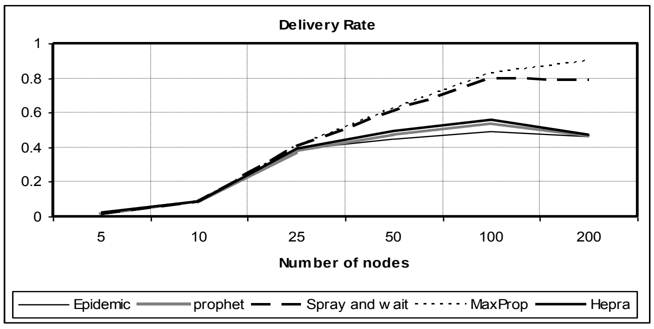


Figure 9.1. MaxProp delivers more messages when nodes' number increases

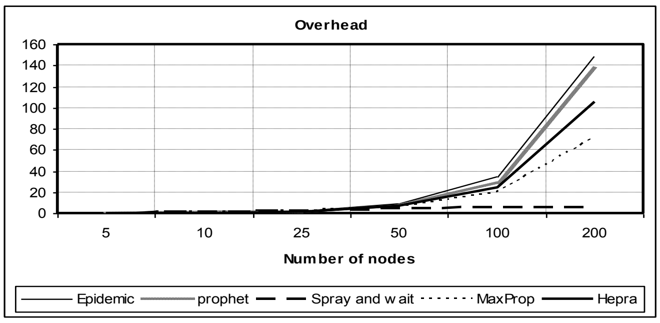


Figure 9.2. SnW has a low overhead

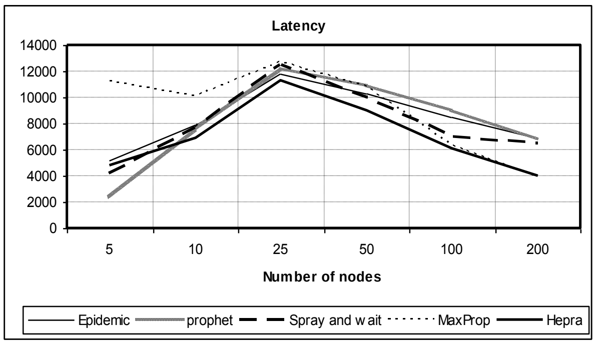


Figure 9.3. HEPRA outperform other in terms of latency number of nodes increases

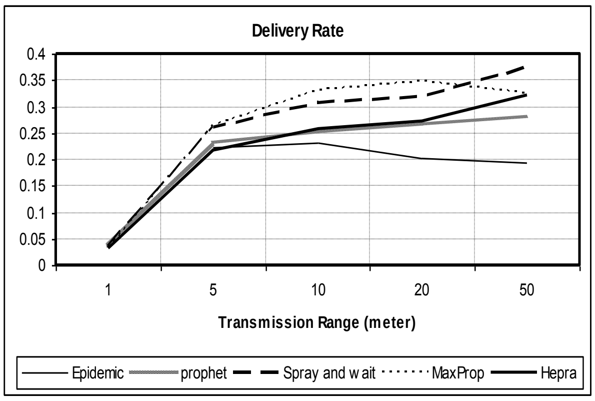


Figure 10.1 SnW and MaxProp delivers more messages when range increases

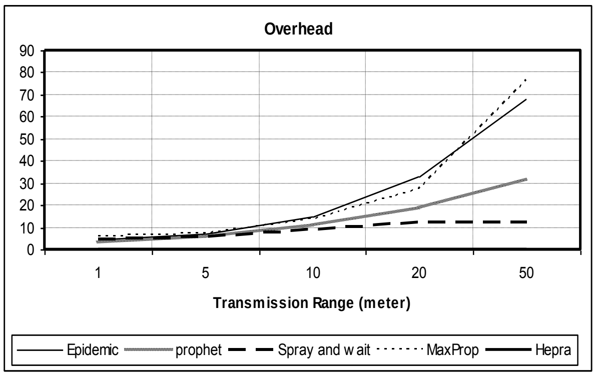


Figure 10.2 HEPRA' overhead is perfect when range varies.

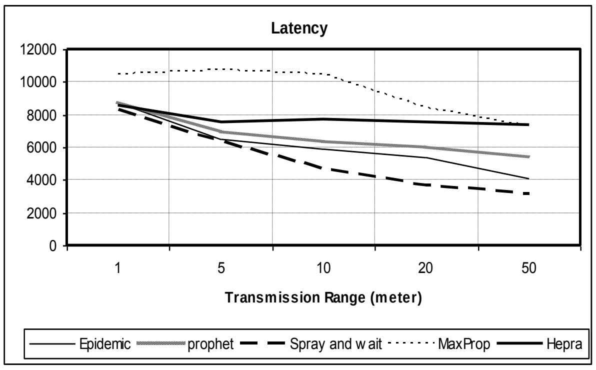


Figure 10.3 SnW has low delay and MaxProp has high delay.

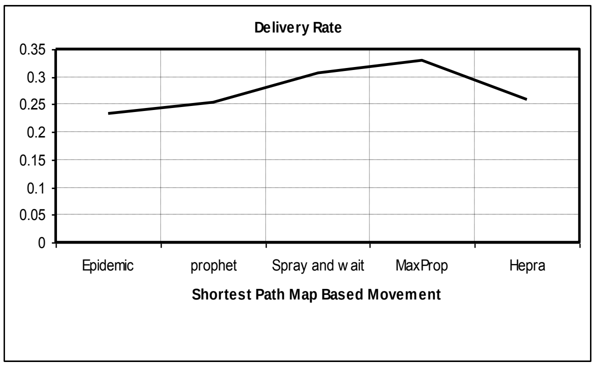


Figure 11.1 SnW has low delay.

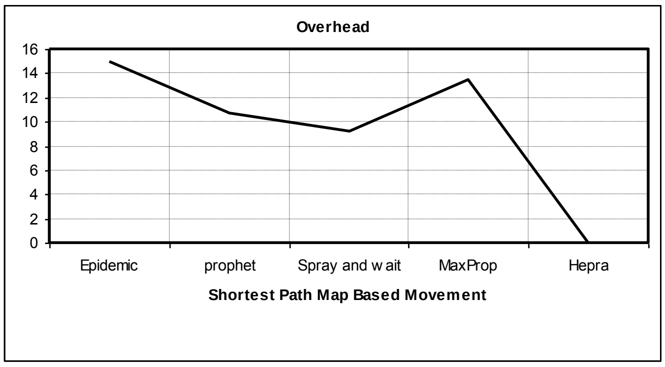


Figure 11.2 HEPRA has the lowest overhead in the shortest path movement.

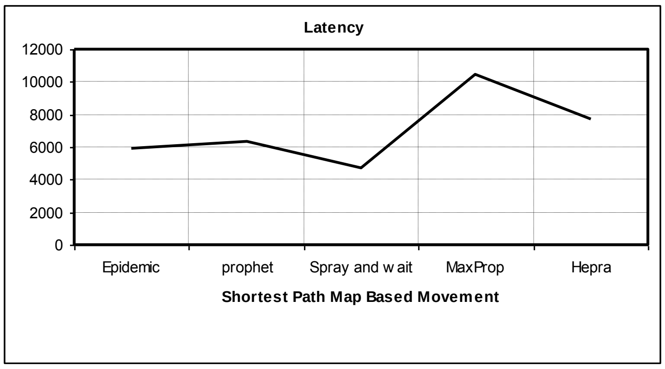


Figure 11.3 SnW's delay is the lowest in the shortest path movement.

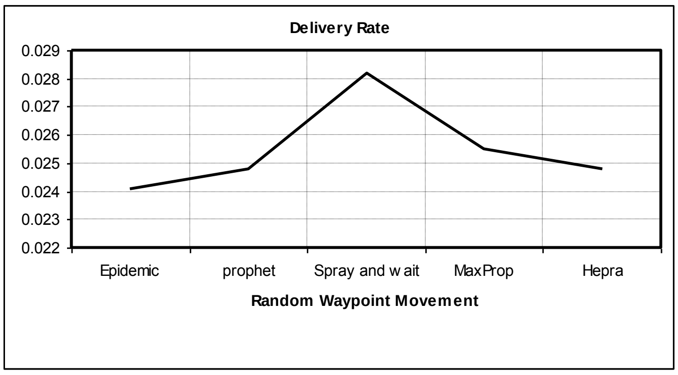


Figure 11.4 SnW delivers more messages than others in the random waypoint movement

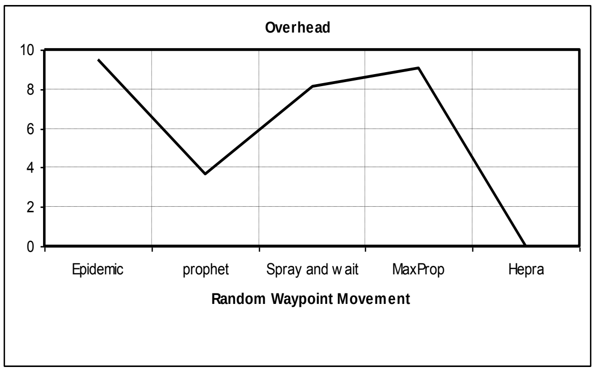


Figure 11.5 HEPRA has the lowest overhead in the random waypoint movement

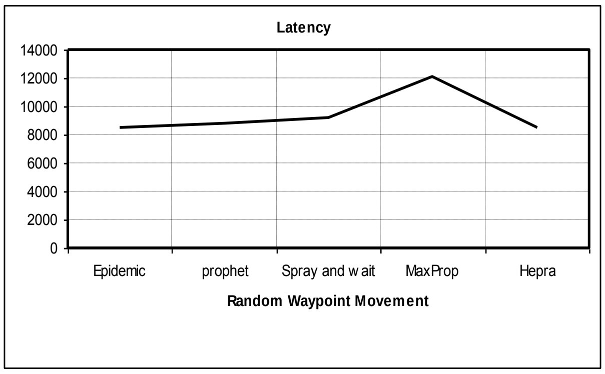


Figure 11.6 Delay of HEPRA and Epidemic is low in the random waypoint movement

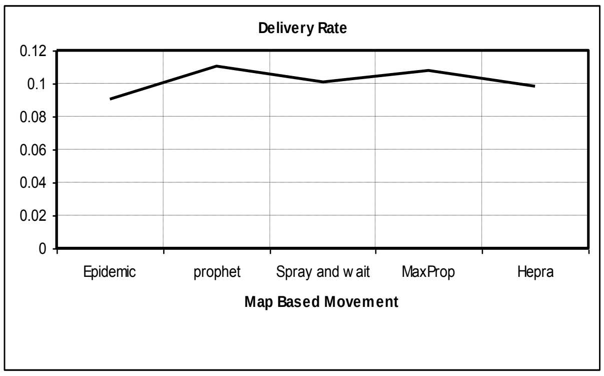


Figure 11.7 PROPHET delivers messages than others in the map based movement

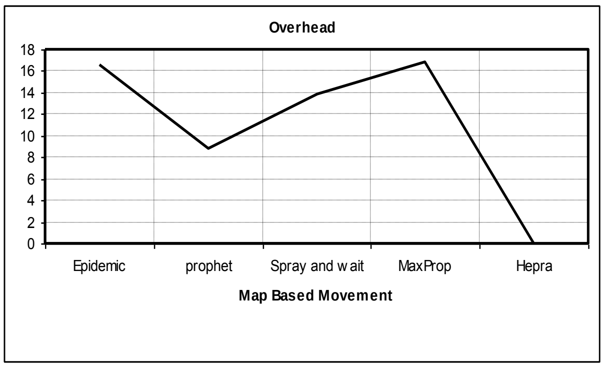


Figure 11.8 HEPRA has the lowest overhead in the map based movement

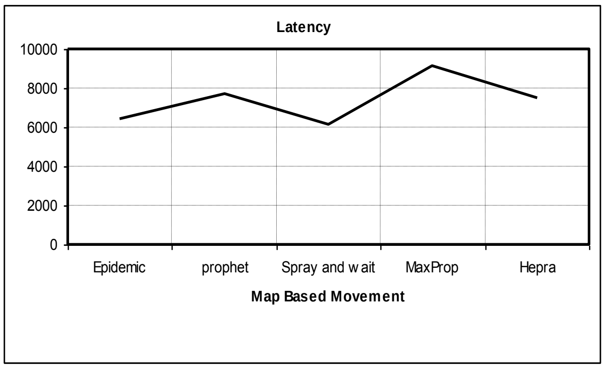


Figure 11.9 Delay of SnW is lower than others in the map based movement

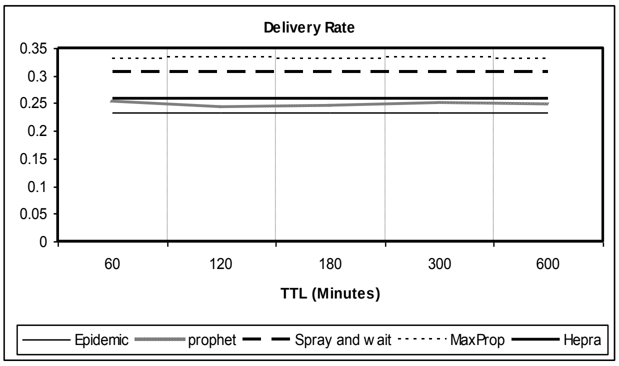


Figure 12.1 MaxProp delivers more messages but fixed when TTL increases

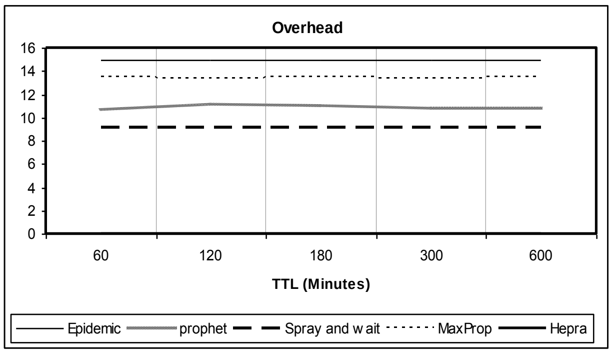


Figure 12.2 HEPRA's overhead is perfect and fixed when TTL increases

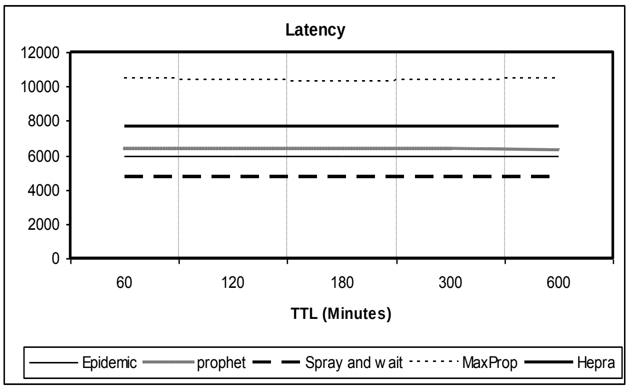


Figure 12.3 MaxProp's delay is lower than others when TTL increases

9. Discussion and Conclusion

We provide in this article an evaluation of the performance of the routing protocols in DTN. We illustrate the behaviors of common DTN routing protocols in terms of various parameters and variables. Each routing protocol has its own advantages and disadvantages in terms of parameters. MaxProp and Spray and Wait perform better in our environment model. The main contributions of this article are as followings:

• Evaluation of performance of HEPRA and propose the Time-Slots HEPRA that illustrate how HEPRA works with this novel approach.

• Analysis of performance of common DTN routing protocols in terms of different parameters in the MANET environment. The article will aid researchers who are new to the field to have a better overall understanding of the performance of those routing protocols.

The challenge was to find a routing algorithm that can deal with dynamic environment causing networks to split and merge, considering nodes mobility, transmission range, buffer size, movement models, and packet TTL. Authors in this article encourage researchers DTN research group to continue developing their routing protocols to achieve high quality performance.

References:

[1] John Seguí and Esther Jennings, Delay Tolerant Networking – Bundle Protocol Simulation. Proc. 2nd IEEE International Conference on Space Mission Challenges for Information Technology, WA, DC, 2006.

[2] DTNRG website (www.dtnrg.org).

[3] DARPA website (http://www.darpa.mil/ATO/solicit/DTN/index.htm)

[4] Forrest Warthman, Delay-Tolerant Networks (DTNs): A Tutorial v1.1, Mar 2003

[5] Jorg Ott, Dirk Kutscher, Christoph Dwertmann, Integrating DTN and MANET Routing, SIGCOMM Workshops, Pisa, Italy, September 11-15, 2006.

[6] Mooi-Choo Chuah, Peng Yang, Brian D. Davison, Liang Cheng, Store-and-Forward Performance in a DTN, Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd.

[7] Michael Demmer, Eric Brewer, Kevin Fall, Sushant Jain, Melissa Ho, and Rabin Patra, Intel Research Berkeley Technical Report IRB-TR-04-020, December 2004

[8] Anders Lindgren, Avri Doria, and Olov Schel´en, Poster, Probabilistic routing in intermittently connected networks, Proc. The Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003), June 2003.

[9] Jeremie Leguay, Timur Friedman, Vania Conan, DTN Routing in a Mobility Pattern Space, SIGCOMM'05 Workshops, Philadelphia, PA, USA., August 22–26, 2005.

[10] Sushant Jain, Michael Demmer, Rabin Patra, and Kevin Fall, Using Redundancy to Cope with Failures in a Delay Tolerant Network, SIGCOMM 2005. August 21–26, 2005, Philadelphia, Pennsylvania, USA.

[11] Fuad Alnajjar, Tarek Saadawi, HEPRA: History of Encounters Probabilistic Routing Algorithm in Delay-Tolerant Network, Parallel and Distributed Computing and Networks (PDCN 2010), Innsbruck, Austria, February 16–18, 2010

[12] Amin Vahdat and David Becker, Epidemic routing for partially connected ad hoc networks, Technical Report CS-200006, Duke University, April 2000.

[13] Anders Lindgren, Avri Doria, and Olov Schel´en, Poster, Probabilistic routing in intermittently connected networks, Proc. The Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003), June 2003.

[14] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi 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, 2005

[15] John Burgess, Brian Gallagher, David Jensen, and Brian Neil Levine. MaxProp: Routing for vehicle-based disruption-tolerant networks. In Proc. IEEE INFOCOM, April 2006.

[16] Aruna Balasubramanian, Brian Neil Levine, and Arun Venkataramani. DTN routing as a resource allocation problem. In Proc. ACM SIGCOMM, August 2007.

[17] Michael Demmer and Kevin Fall, DTLSR: Delay Tolerant Routing for Developing Regions ACM SIGCOMM Workshop on Networked Systems in Developing Regions (NSDR), August 27, 2007, Kyoto, Japan.

[18] Paolo Costa, Cecilia Mascolo, Mirco Musolesi, and Gian Pietro Picco, Socially-Aware Routing for Publish-Subscribe in Delay-Tolerant Mobile Ad Hoc Networks, IEEE Journal on selected areas in communications, VOL. 26, No. 5, June 2008.

[19] Qun Li and Daniela Rus, Communication in disconnected ad-hoc networks using message relay, Journal of Parallel and Distributed Computing, 2003.

[20] Ari Keranen and Jorg Ott: Increasing Reality for DTN Protocol Simulations, Technical Report, Helsinki University of Technology, Networking Laboratory, July 2007.