

An Encounter-Based Multicast Scheme for Disruption Tolerant Networks

M. Chuah, Y. Xi
{chuah, yox205}@cse.lehigh.edu
CSE Department
Lehigh University
Bethlehem, PA 18015

Abstract—Some ad hoc network scenarios are characterized by frequent partitions and intermittent connectivity. Hence, existing adhoc routing schemes that assume that an end-to-end path exists from a source to a destination do not work in such challenging environment. A store-and-forward network architecture known as the disruption tolerant network (DTN) has been designed for such challenging network environments. Several unicast and multicast routing schemes have been designed for DTNs. However, the existing multicast routing schemes assume a route discovery process that is similar to the existing adhoc network routing approach. Thus, in this paper, we explore an encounter-based multicast scheme for DTNs. Our approach builds on top of an existing DTN unicast routing scheme called Prophet. Via simulations, we study the delivery performance of this encounter-based multicast scheme. Our simulation results indicate that high delivery ratio can be achieved with reasonable data transmission efficiency.

Keywords: multicast routing; disruption tolerant networks; redundancy.

I. INTRODUCTION

With the advancement of technology, we can find many wireless devices e.g. sensors, PDAs, laptops etc. Such devices can form wireless adhoc networks dynamically without any infrastructure support. Much work has been done in the past to design unicast routing schemes for adhoc networks [1],[2]. However, the unicast routing schemes often assume that an end-to-end path exists between a source/destination pair and hence are not suitable for challenging network environments where the nodes experience intermittent connectivity and frequent partitions. In addition, much design has also been done for delivering multicast traffic in adhoc networks e.g. [3],[4]. Again, such multicast routing schemes often assume a multicast tree can be maintained for delivering multicast traffic. It is difficult to maintain a multicast delivery tree in challenging network environments with frequent partitions and intermittent connectivity among the nodes.

Recently, a new network architecture [5] called the Disruption Tolerant Network (DTN) has been proposed to allow partitioned nodes or clusters of nodes to communicate with one another. Recent research interests in this area include network architecture design [5],[6], and different routing algorithms for

DTNs [7],[8],[9],[10],[11],[12]. Most of the routing schemes designed are delivering unicast traffic. However, there are applications that need multicasting support e.g. distributing mission planning information in the battlefield. Several DTN multicast routing schemes have been designed for DTNs e.g. [13],[14]. These DTN multicast routing schemes rely on a route discovery process that is similar to traditional adhoc routing schemes and hence may not be able to perform well when the network becomes very sparse. In [15], the authors overcome this problem by allowing nodes in very sparse environment to use high power transmission to complete the route discovery process and use message ferrying to deliver data packets. They show that such an adaptive scheme can significantly improve delivery performance. However, this approach assumes that mobility of the nodes can be controlled which may not be feasible in some network scenarios. Thus, we are interested in exploring a multicast delivery approach that does not require the nodes to have controlled mobility. Specifically, we explore an encounter-based multicast routing scheme that is built on top of the Prophet routing scheme.

The remainder of this paper is organized as follows. We provide a brief review of related work in Section 2. In Section 3 we present our encounter-based multicast routing scheme for DTNs. In Section 4, we describe our simulation setup and present our simulation results. We conclude in Section 5 with some discussions on future work.

II. RELATED WORK

A. Routing in Intermittently Connected Networks

Several routing schemes have been proposed for DTNs [10],[11],[12],[13],[14]. These different schemes can be grouped into three categories. The first category [9] uses special nodes called ferries to deliver messages between partitioned networks. Ferry routes have significant effect on the data delivery performance, hence they need to be designed efficiently. The second category [11],[12] uses multihop routing approach where contact history information is used to determine the next hop node to pass a message. For example, in [12], a probabilistic metric called delivery predictability is used to determine if a node needs to

pass any stored messages to a new contact that it comes across. The third category [13],[14] uses a two-hop routing approach where the intermediate nodes that receive messages from any source have to store the messages until they can deliver the messages when they come into contact with the destinations of the messages. Sometimes, erasure-coding is used to encode and divide the message into multiple blocks and these different blocks are sent to different relays to increase the chances of a destination receiving a particular message since the destination only needs to receive a certain fraction of the encoded blocks to reconstruct the original message.

B. Multicast Routing Schemes for DTNs

Several multicast routing schemes have been designed for DTNs, namely (a) DTBR [13], (b) OS-Multicast [14], and (c) Context-Aware Multicast Routing (CAMR) [15]. DTBR is a tree-based multicasting algorithm. DTBR assumes that each source node of the multicast group has complete knowledge or a summary of the link states in the network. During the lifetime of a multicast session, DTBR requires an upstream node to assign the receiver list for its downstream nodes based on its knowledge of the current network topology. The downstream nodes are allowed to forward bundles only to the receivers in the list. Custody transfer feature is enabled for those DTN nodes along the multicast tree. However, since the network topology changes frequently, it is not easy to maintain the multicast delivery tree. In addition, the receiver list cannot be adjusted by intermediate nodes once it is decided by upstream nodes, which means newly discovered delivery opportunities cannot be used by intermediate nodes.

Due to the limitations of DTBR, OS-multicast [14] was proposed. OS-multicast relies on a DSR-like routing to build a knowledge base of the link state and network topology. Unlike DTBR, OS-multicast let each intermediate node maintain a tree rooted at itself to all the receivers and adjust the receiver list according to local knowledge of the network topology. Via simulations, the authors [14] show that OS-multicast achieves good performance when the probability of the link unavailability is high. However, all simulations are based on a network of 25 nodes deployed in a 1000×1000 m² area, which is still quite well-connected. The authors in [15] show that the performance of OS-multicast degrades when the network becomes sparser. Moreover, OS-multicast relies on a DSR-like route discovery process to build a

knowledge base of the current network topology. Such a process will not work in a very sparse network environment.

In [15], the authors propose a context-aware multicast routing (CAMR) scheme where nodes are allowed to use high power transmissions when the locally observed node density drops below a certain threshold. Each node maintains a 2-hop neighborhood information, and hence can deliver traffic without invoking a route discovery process if all receivers are within its two-hop neighborhood. In addition, the nodes are allowed to act as message ferries when they discover they are in a very sparse neighborhood. The combined high-power route discovery process and message ferrying features allow CAMR to achieve much higher multicast delivery ratio than DTBR and OS-multicast schemes. However, CAMR still relies on a route discovery process that is similar to the traditional adhoc routing approach and also rely on the ability to control node movement. Thus, we are motivated to design a scheme that does not rely on the traditional route discovery process but purely based on node encounters. We describe our encounter-based multicast routing scheme in the next section.

III. ENCOUNTER-BASED MULTICAST ROUTING (EBMR) SCHEME

Before we describe our EBMR scheme, we first give a summary on how the original Prophet [10] works. Prophet uses the history information of encounters and transitivity to select a next-hop node for message delivery. Specifically, Prophet establishes a probabilistic metric called delivery predictability at every node A for each known destination B. This metric indicates how likely it is that node A will be able to deliver a message to that destination. The delivery predictability ages with time and also has a transitive property, i.e., a node A that encounters node B which encounters node C allows node A to update its delivery predictability to node C based on its (A's) delivery predictability to node B and node B's delivery predictability to node C. In Prophet, a node will forward a message to another node it encounters if that node has higher delivery predictability to the destination than itself. Such a scheme was shown to produce superior performance than epidemic routing [10]. The three equations used for updating the delivery predictability are as follow:

$$P(a,b) = P(a,b)_{old} + (1 - P(a,b)_{old}) * \alpha$$

$$P(a,b) = P(a,b)_{old} \times \gamma^k$$

$$P(a,c) = P(a,c)_{old} + (1 - P(a,c)_{old}) * P(a,b) * P(b,c) * \beta$$

In [10], α is set to 0.75, β is set to 0.25 and γ is set to 0.98. Each node broadcasts a beacon periodically. The beacon contains the delivery predictability values from this node to all other nodes. Such delivery predictability values are updated upon receiving beacons from other nodes.

Next, we describe how our encounter-based multicast routing (EBMR) scheme works. In Figure 1, we have divided the geographical area into four different cells. Cell 1, where the multicast source S is located, is the area with the highest node density (each node may have 6-8 neighbors). Cells 2 and 3 are the areas with medium node density (each node may have 2-4 neighbors), and Cell 4 is the area with the lowest node density where each node may have 0-2 neighbors. There is a multicast source S and eight multicast receivers marked as R₁ to R₈ in the picture.

Our EBMR scheme is built on top of the Prophet [10] scheme. Several enhancements are made in EBMR to improve the delivery performance. First, each node will not pass the bundle to a next-hop node unless that next-hop node has a delivery predictability higher than a delivery threshold (P_{thresh}). For multicast delivery, each node in the EBMR scheme will pick as many nodes as needed with the highest delivery predictability to each of the multicast receivers. A node will cache the data if no such next-hop node is found until a wait timer expires. If the wait timer (WT) expires, then the node will simply pick a node with the highest delivery predictability to a multicast receiver whose next-hop node has not been selected. Such enhancement improves delivery performance. For example, in Figure 1, node n₉ holds onto the multicast bundle until it can hear a node in Cell 4 that can be selected as a next-hop node. Similarly, node n₁₀ waits till it can hear node n₁₁ before n₁₀ passes the multicast bundle to n₁₁. The second enhancement we make is to allow nodes in the boundary region of each cell to use directional antenna to find nodes in other cells if they cannot hear any such node using omnidirectional antenna. This enhancement allows a node to discover more nodes in a sparse environment. A third enhancement added to further improve the delivery performance is that the source is allowed to select K next-hop nodes for each multicast receiver.

IV. PERFORMANCE EVALUATION

A. Simulation Setup

In order to evaluate the EBMR scheme, we implement this scheme using NS-2 simulator [17].The

performance metrics we used in our evaluation are: (1) *Delivery Ratio*, which is the ratio of the number of successfully received multicast messages by all receivers over the total number of multicast messages that have been sent. (2) *Average Delivery Latency*, which is defined as the average end-to-end delay incurred by the delivered messages, (3) *Data Efficiency*, which is the number of transmissions used to deliver multicast messages over the total number of received multicast messages, and (4) the average buffer usage which is the average number of buffers used for storing data.

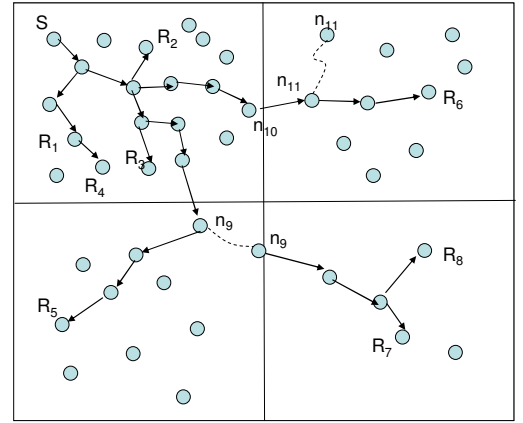


Figure 1: EBMR scheme

In our simulation, 40 nodes are randomly distributed over a certain geographical area e.g. 3000x3000m². By default, the nodes move according to the random waypoint model [16]. Unless otherwise stated, we set the pause time to be 10 seconds, and the maximum node speed to be 5 m/s.

For the traffic model, we sometimes use (a) one multicast session with one source and multiple receivers, (b) multiple multicast sessions with one source and multiple receivers in each session. Each multicast source generates CBR traffic with a packet size of 512 bytes. We let the source generates traffic after 1000 seconds of warming up period and the traffic generation lasts for 2000 seconds but the simulation will run until 10,000 seconds. Each reported data point is the average of 10 runs.

B. Comparison of different multicast routing schemes.

In this section, we first illustrate the performance improvement one can get by the two enhancements we introduced in EBMR: (a) allowing nodes to withhold the message passing until it can meet a node with a

delivery predictability higher than $P_{\text{threshold}}$ or until a wait timer (WT) expires, (b) allowing each source to select K next-hop node for each multicast receiver. We compare the following multicast delivery schemes: (a) brute-force unicast delivery using the original Prophet scheme where the source duplicates one message for each multicast receiver and delivers each message using the original Prophet, (b) multicast delivery using the original Prophet scheme where the source only duplicates as many messages as needed for the different next-hop nodes, (c) EBMR delivery with $K=1$, $P_{\text{threshold}}=0.5$ and $WT=500$ seconds, (d) EBMR delivery with $K=2$, $P_{\text{threshold}}=0.5$ and $WT=500$ seconds. We use a network scenario with 40 nodes randomly distributed over $3000 \times 3000 \text{m}^2$. Each multicast session has one source and 4 receivers. Each multicast source generates 1 bundle every 10 seconds. We vary the number of multicast sessions. The results for the delivery ratio, the average delay, average number of hops it takes to reach a receiver, and the data efficiency are plotted in Figures 2(a), 2(b), 2(c) and 2(d) respectively.

Figure 2(a) show that the EBMR with $K=2$ scheme achieves the best delivery performance: it has the highest delivery ratio and lowest average delay. The lower average delay for the unicast approach in Figure 2(b) is misleading since the delivery ratio drops significantly with increasing number of sessions using the unicast approach, and only messages that can easily be delivered will reach the various multicast destinations. Figure 2(c) shows that the average number of hops it takes to reach the different multicast receivers is significantly reduced with our enhanced EBMR scheme. Since $K=2$ incurs extra data redundancy, it is not surprising that the EBMR approach with $K=1$ achieves the best data efficiency as shown in Figure 2(d).

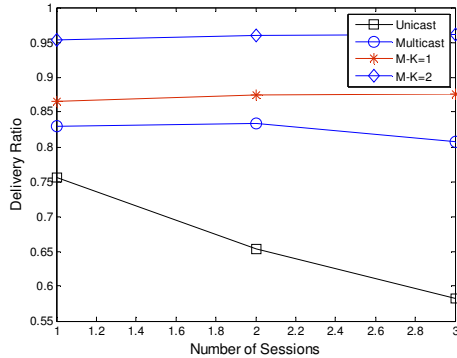


Figure 2(a) Delivery Ratio vs Number of Sessions

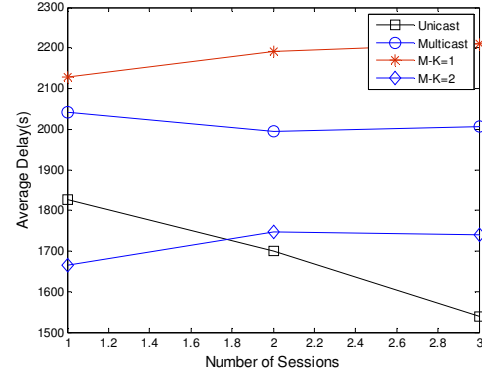


Figure 2(b): Average Delay vs Number of Sessions

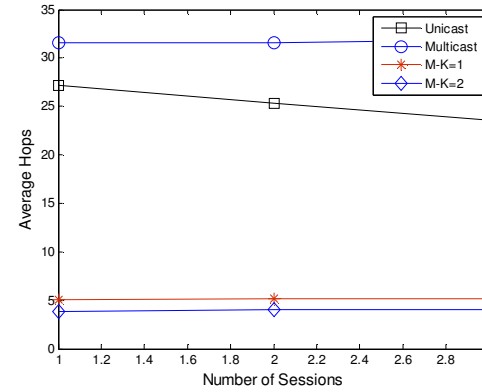


Figure 2(c): Avg Number of Hops vs Avg Number of Sessions.

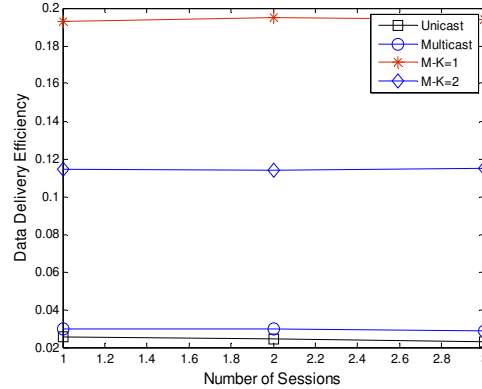


Figure 2(d): Average data efficiency vs Number of Sessions

C. Sensitivity Analysis of P_{thresh} and WT

In this section, we performed some sensitivity analysis to see how different P_{thresh} and WT values affect the delivery performance. In this experiment, we also use 40 nodes distributed over $3000 \times 3000 \text{m}^2$. We use two multicast sessions with each session having one multicast source and 4 receivers. Each multicast source generates 1 bundle every 10 seconds. The results for the delivery ratio, the average delivery latency, the average data efficiency and the average buffer usage are plotted in Figures 3(a), 3(b), 3(c) and 3(d) respectively.

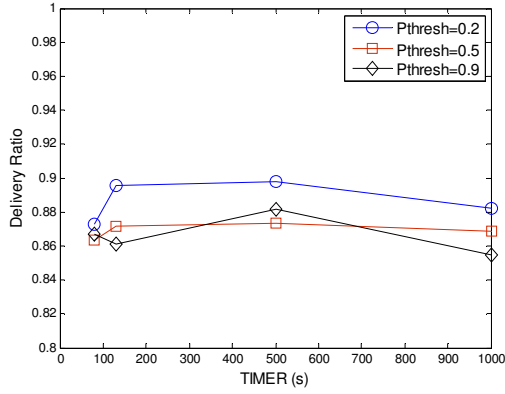


Figure 3(a): Avg Delivery Ratio vs Timer

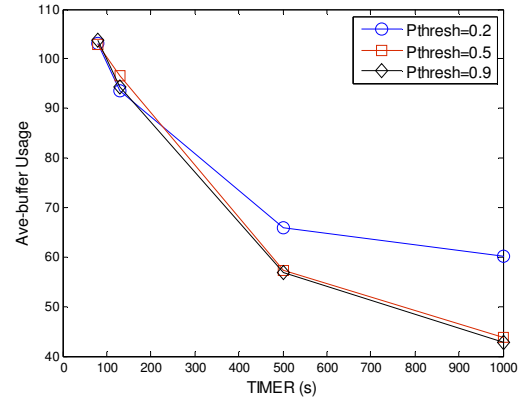


Figure 3(d): Avg buffer usage vs Timer

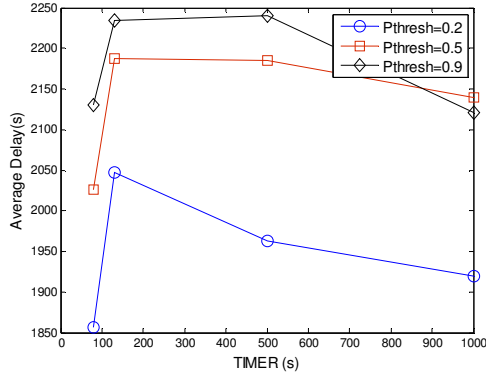


Figure 3(b): Avg Delay vs Timer

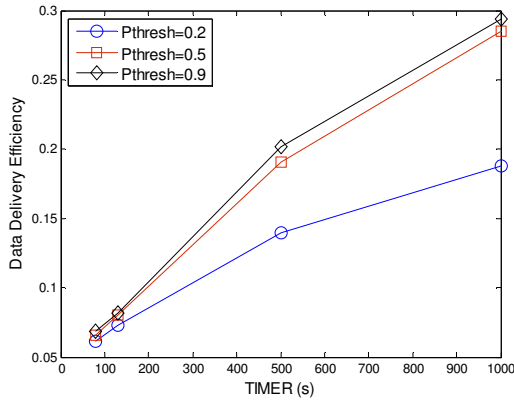


Figure 3(c): Avg data efficiency vs Timer

From the plots, we see that setting $P_{\text{thresh}}=0.2$ gives high delivery ratio, low average delay but also lower data efficiency and higher buffer usage. With increasing WT values, the average delay and the data efficiency improves since the nodes can find better next-hop node and hence the average number of hops to reach a multicast destination decreases with increasing WT. From the results, it seems that setting $P_{\text{thresh}}=0.5$ and timer=500 seconds is a good trade off between achieving high delivery ratio, high data efficiency, and lower buffer usage.

D. Enhanced EBMR scheme

We analyzed our traces to understand the reason for the failed delivery. We noticed that most of the failed delivery is due to the IFQ overflows in the NS-2 simulator. The queued messages are released to the MAC level immediately after a suitable next-hop node is selected and hence caused IFQ overflows. Thus, we modified the simulator to allow such messages to be released one by one after an average time of 0.5 seconds to avoid IFQ overflow. In addition, we also include an optimization where a next-to-last hop node caches the acknowledgements of the multicast destinations so that it does not forward a multicast message to any destination twice. This enhancement improves the data efficiency of the EBMR scheme with $K=2$. In this section, we show the performance improvements we get after these two enhancements. Again, we use 40 nodes over $3000 \times 3000 \text{ m}^2$ scenario. One multicast session with one multicast source and 8 receivers is used. The chosen parameters for EBMR are $K=2$, $P_{\text{threshold}}=0.5$ and $WT=500$ seconds. We vary the multicast source generation rate. We also include the simulation results for the CAMR protocol [15] for comparison purposes. The results are plotted in Figures 4(a), 4(b), 4(c) and 4(d) respectively.

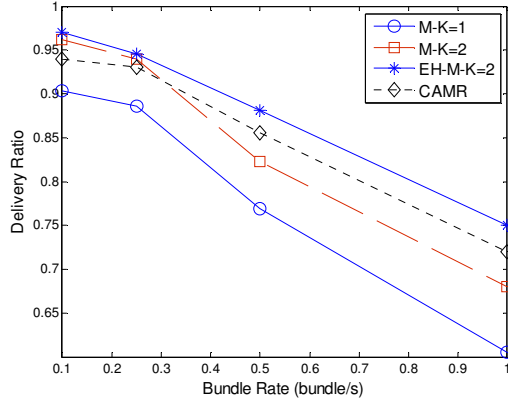


Figure 4(a): Avg Delivery Ratio vs Message Rate

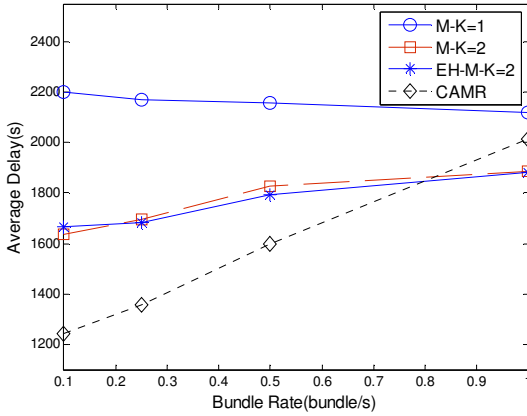


Figure 4(b): Avg Delay vs Message Rate

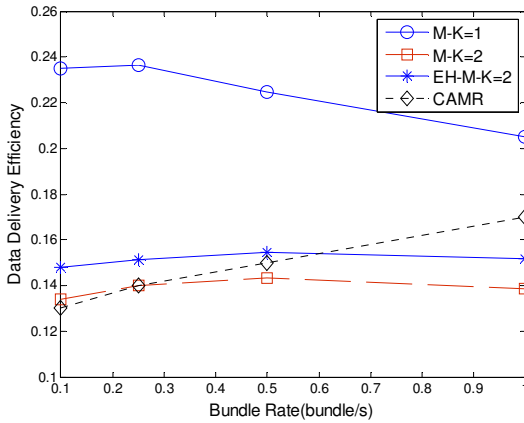


Figure 4(c): Avg data efficiency vs Message Rate

Figure 4(a) shows that the enhancement helps to increase the achievable delivery ratio for the EMBR scheme and lower the delivery latency. The price to pay is a slight increase in the buffer usage. The results also indicate that the delivery ratio drops with increasing multicast traffic load. Setting $K=2$ gives better delivery performance but at the price of lower data efficiency. The plots also indicate that the enhanced EMBR scheme can achieve slightly higher delivery ratio than CAMR scheme but at the expense

of higher buffer usage, lower efficiency at high load. For subsequent sections, we use the symbol EH-M-K=2 to indicate the EMBR scheme with such enhancements.

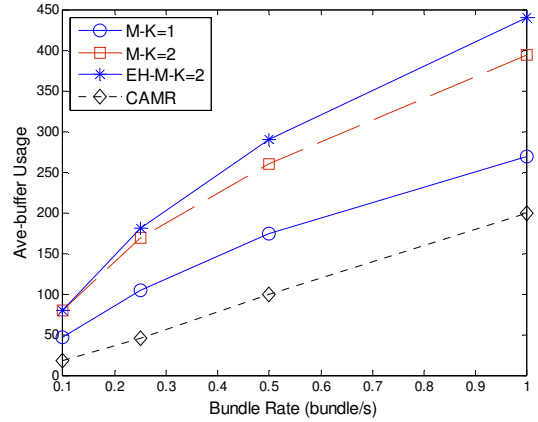


Figure 4(d): Avg buffer usage vs Message Rate

E. Impact of different multicast sessions

In this experiment, we further compare the EMBR scheme with $K=1$, $K=2$ and the enhanced EMBR (with the IFQ and the next-to-last-hop delivery enhancements discussed earlier) scheme when different number of multicast sessions are used. Each multicast session has one source and 4 receivers. We vary the number of multicast sessions. The source generates messages at a rate of 1 message/4 seconds. The results are plotted in Figures 5(a), 5(b), 5(c), and 5(d) respectively.

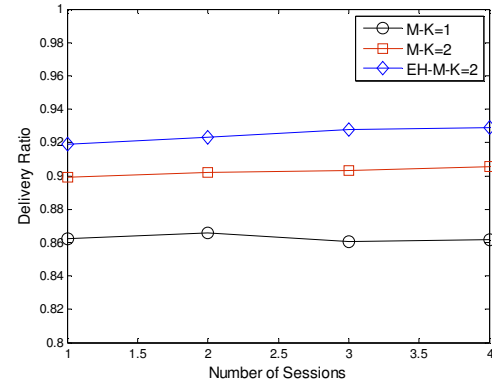


Figure 5(a): Avg Delivery ratio vs Number of Sessions

The delivery ratio remains almost the same but the average delay increases as the number of sessions increases. The average buffer usage also increases with more sessions. Setting $K=2$ improves the delivery performance but at the cost of lower data efficiency and increased buffer usage. Thus, a trade-off needs to be done when deploying EMBR in the field. If the delivery ratio is more important than energy usage,

then $K=2$ should be used. Otherwise, setting $K=1$ will suffice.

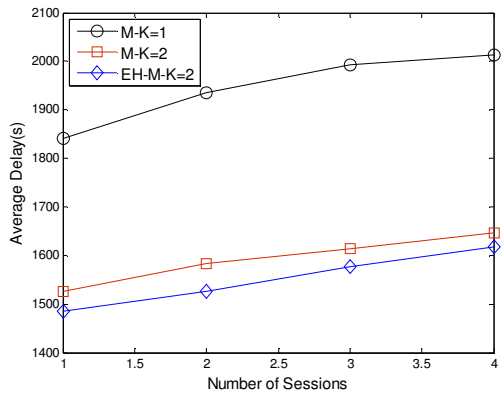


Figure 5(b): Avg Delay vs Number of Sessions

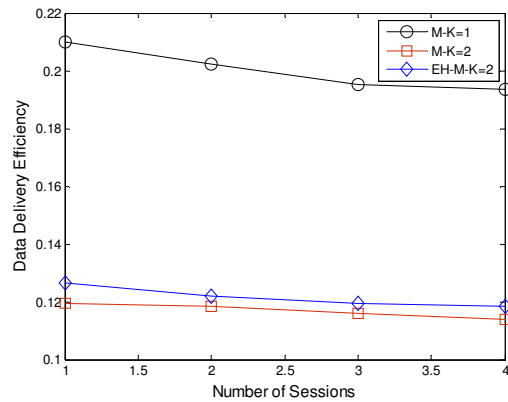


Figure 5(c): Avg data efficiency vs Number of Sessions

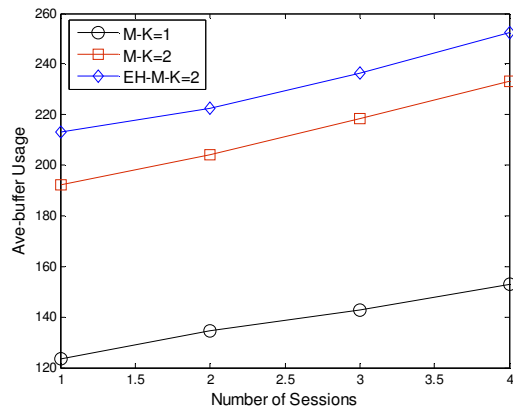


Figure 5(d): Avg buffer usage vs Number of Sessions

F. Impact of different number of receivers

Last but not least, we perform another set of experiment where we use one multicast session with one source but vary the number of receivers. We use the network scenario with 40 nodes distributed over

3000x3000 m². The source generates messages at a rate of 1 message/4 seconds. Figures 6(a), 6(b), 6(c) & 6(d) plot the average delay ratio, the average delivery latency, the average data efficiency and the average buffer usage results we obtained in this experiment respectively.

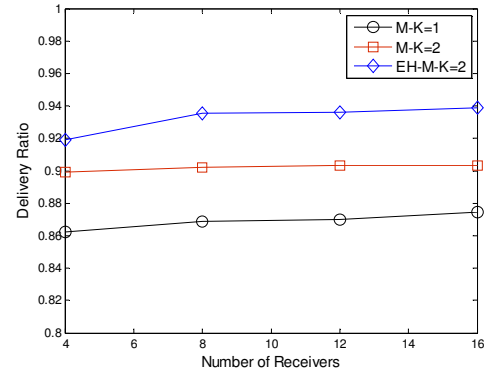


Figure 6(a) Delivery Ratio vs Number of Receivers

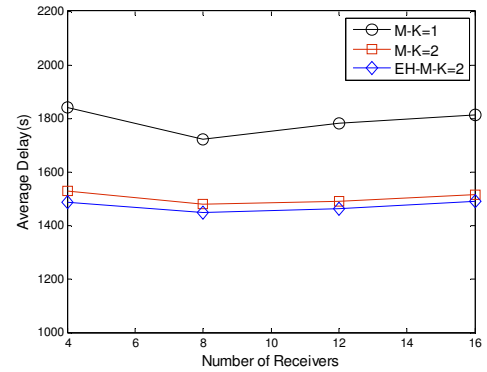


Figure 6(b): Avg Delay vs Number of Receivers

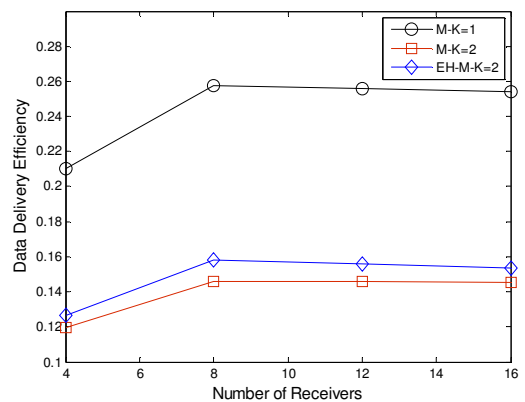


Figure 6(c): Avg data efficiency vs Number of Receivers

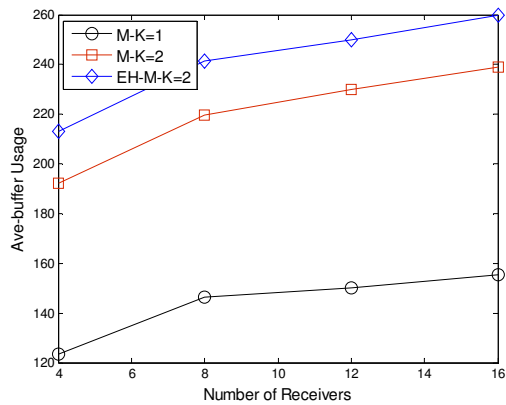


Figure 6(d): Avg buffer usage vs Number of Receivers

From the plots, one can see that the delivery ratio and the average delay remains almost the same with different number of receivers. The performance does not degrade with increasing number of receivers because the network is very sparse and the multicast source rate is not high enough to cause buffer overflows. The data efficiency increases initially with the number of receivers since the chances of using the same next-hop node to deliver multicast messages increase as the number of receivers increases. Beyond a certain point, no such gain can be observed so the average data efficiency then remains almost the same. The average buffer usage increases with increasing number of receivers.

V. CONCLUSION

In this paper, we have presented an encounter-based multicast routing scheme for DTNs. Our scheme allows nodes to cache the data until a good next-hop node can be found to relay the messages to the destinations. Via simulation studies, we have demonstrated that this scheme can achieve high delivery ratio with reasonable data efficiency over different scenarios. There are several interesting issues we intend to explore further e.g. we wish to investigate the impact of other mobility models e.g. Zebranet mobility model [11], traffic patterns, the impact of power management on the multicast delivery performance. In addition, we intend to evaluate the EBMR scheme in scenarios with non-homogenous node distribution. We intend to compare this protocol with CAMR using more scenarios. We also intend to explore an adaptive scheme where data replication is invoked only when the delivery predictability is below a certain threshold. We intend to implement this scheme on a medium size testbed to evaluate its performance with real multicast applications running among the different DTN nodes.

Acknowledgements: This work has been supported by DARPA under Contract W15P7T-06-C-P430. Any opinions, findings, and conclusions or recommendations

expressed in this paper are those of the authors and do not necessarily reflect the views of the sponsor of this work.

REFERENCES

- [1] D. Johnson, D. Maltz, "Dynamic Source Routing in Ad-Hoc Wireless Networks", Proceedings of ACM Sigcomm, August 1996
- [2] S. Das, C. Perkins, E. Royer, "Performance Comparison of Two On-Demand Routing Protocols", Proceedings of IEEE Infocom, March, 2000.
- [3] E. Royer, C. Perkins, "Multicast Adhoc On-Demand (MAODV) Routing", draft-ietf-manet-maodv-00.txt", July, 2000.
- [4] J.J. Garcia-Luna-Aceves, "A multicast routing protocol for Adhoc Networks (CAMP)", Proceedings of IEEE Infocom, 1999 pp 784-792.
- [5] Forrest Warthman, "Delay-Tolerant Networks (DTNs) – A tutorial", based on DTN Research Group Internet Draft, March 2003.
- [6] K. Fall, "A delay-tolerant network architecture for challenged internets", in SIGCOMM, 2003.
- [7] M.M.B.Tariq, M. Ammar, and E. Zegura, "Message Ferry Route Design for Sparse Ad hoc Networks with Mobile Nodes", ACM MobiHoc, May22-25, 2006.
- [8] S.Jain, K.Fall, and R. Patra, "Routing in a Delay Tolerant Network", SIGCOMM'04, Aug. 30-Sept. 3, 2004.
- [9] J. Burgess, B. Gallagher, D. Jensen, and B.L.Levine, Maxprop: Routing for vehicle-based disruption-tolerant networks. In INFOCOM, 2006.
- [10] A. Lindgren, A.Doria, and O.Scheln, Probabilistic Routing in Intermittently Connected Networks. In Proc. Workshop on Service Assurance with Partial and Intermittent Resources, August 2004.
- [11] Y. Wang, S. Jan, M. Martonosi, and K. Fall, "Erasure-Coding Based Routing for Opportunistic Networks", Proceedings of ACM Sigcomm WDTN Workshop, August 2005..
- [12] S. Jain, M. Demmer, R. Patra, K. Fall, "Using Redundancy to cope with Failures in a Delay Tolerant Network", Proceedings of ACM Sigcomm, August, 2005.
- [13] W. Zhao, M. Ammar, E. Zegura, "Multicasting in delay tolerant networks: semantics models and routing algorithms", Proceedings of ACM workshop on Wireless DTN, August, 2005.
- [14] Q. Ye, L. Cheng, M. Chuah, B. Davison, "On-demand situation-aware multicasting in DTNs", Proceedings of IEEE Vehicular Technology Conference, May, 2006.
- [15] P. Yang, M. Chuah, "Context-Aware Multicast Routing Schemes for DTNs", Proceedings of ACM Workshop on PE-WASUN, Aug, 2006.
- [16] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research", Wireless Communications & Mobile Computing (WCMC), Vol .2, no. 5, pp. 483-502, 2002.
- [17] "The network simulator ns-2", [Online] at <http://www.isi.edu/nsnam/ns/>.