Efficient Broadcasting in Delay Tolerant Networks

Appu Goundan
Department of Computer Science
University of Southern California
Email: goundan@usc.edu

Eric Coe
Ming Hsieh Dept. of Electrical Eng.
University of Southern California
Email: ecoe@usc.edu

Cauligi Raghavendra
Ming Hsieh Dept. of Electrical Eng.
University of Southern California
Email: raghu@usc.edu

***Abstract*—Delay Tolerant Networks are a class of networks characterized by intermittent connectivity, long delays, and noncontemporaneous end-to-end paths between nodes. Standard Internet protocols do not fare well in these situations and special protocols have been developed to handle routing, broadcasting and other mechanisms for transporting data. In this paper, we discuss a mechanism for energy efficient broadcast - the k-neighbor broadcast scheme. Nodes do not broadcast all the time, but wait for an opportunity to reach multiple nodes with one transmission, thereby reducing the number of transmissions overall. We performed a simulation based study and found our scheme performs efficiently in a predictable manner.**

I. Introduction

Increasingly networks are being employed in challenged environments, such as army battlefields, space exploration, sensor networks with limited power budgets, etc. These networks will have intermittent connectivity, long delays, and exhibit noncontemporaneous end-to-end paths between nodes. Therefore, traditional Internet protocols will not perform at all or perform very poorly. Such networks are called delay/disruption tolerant networks (DTN) and there is ongoing research to bring Internet like services to these networks. Communication schemes in DTNs are specifically designed for situations where full network connectivity is unavailable and they implement the necessary mechanisms for effective transfer of data over these networks.

Delay tolerant networks research has progressed recently with an architecture [6] proposed by The Delay Tolerant Networks Research Group (DTNRG) [1] to support these new networks. This architecture discusses the bundling layer, custody transfer, naming and late binding. There is also a reference implementation of the DTN architecture [2]. One of the problems extensively worked on in DTNs is routing [4], [13], [15], [16]. With intermittent connectivity and long delays between contacts in DTNs, efficient routing schemes are developed based on utility functions and an understanding of movement of nodes. In general these networks are wireless networks and the objective is to reduce delays and energy consumption for communication. There is limited work on multicasting [18] and broadcasting schemes for DTNs. Broadcasting mechanisms are important in DTNs for disseminating critical information to most or all nodes in the network. With long delays between contacts, the goal for a broadcasting algorithm is to reach as many nodes as possible with minimum delay and minimal energy consumption.

This research was funded in part by NSF grant number 0520017 and NSF grant number 0626788

Efficient broadcasting has been well studied in the context of Mobile Ad Hoc Networks (MANETs) which are similar to DTNs but MANETs assume end-to-end paths and low delays between contacts. [5], [7]–[9], [14], [17] provide mechanisms for efficiently broadcasting messages throughout an ad-hoc network. Some work is focused on energy efficiency in broadcasting [14], while others introduce mechanisms to prevent nodes from receiving broadcast messages redundantly [17]. In wireless networks the broadcast medium can help reduce the total number of transmissions by sending data to multiple nodes in range at the same time. MANET broadcast algorithms will not work at all or will perform poorly when applied directly to DTNs where there are long delays between contacts and no end-to-end paths.

There is limited work on broadcasting in DTNs. [10] discusses broadcasting of content and proposes an architecture for users to selectively subscribe to broadcast channels. The architecture allows users to download data on different broadcast channels like a television or a radio. Nodes were also required to carry and forward data for other channels in order to maximize coverage for all content. The paper discusses a mechanism to provide maximum coverage for all data while optimizing for storage space and energy, but suggests no general mechanism for energy efficiency. [11] discusses a mechanism called hypergossiping for MANETs that is tolerant of delays and partitions in the network. Hypergossiping can adapt to varying network conditions, broadcast messages across partitions and reduce the severity of broadcast storms, but does not address energy efficiency specifically.

In this paper, we develop an efficient broadcasting scheme for DTNs where nodes are mobile and can communicate wirelessly when they are in range of other nodes. The problem is to send a broadcast message from a source to all nodes while trying to minimize energy. If we want to minimize delay, the best approach is to use some form of flooding but that will waste energy as nodes will receive messages redundantly. On the otherhand, to minimize energy, each node should transmit when multiple neighbors can receive the message. One can also eliminate redundant receptions if nodes can turn themselves off during redundant transmissions. There is a tradeoff between time delay and energy reduction which is explored in this paper. We develop a general k-neighbor broadcast scheme to effectively explore energy efficient broadcasting in DTNs

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

and use simulations to study and evaluate the performance of our algorithm.

II. k-Neighbor Broadcast Scheme

A. Motivation

Consider broadcasting a message from a source node to all nodes in a DTN. A simple scheme to accomplish this is to send this message from each node via unicast using some DTN routing protocol. This, however, will take a significant amount of energy. Alternatively, a node with this message can transmit to any other node that comes in contact as in flooding and this will take the least amount of time at the expense of energy. Flooding by nature tends to overload the network with transmissions, possibly crippling network resources by taking over communication channels [12]. A more efficient broadcast scheme would address these issues to reduce power consumption, network contention and traffic load.

B. Mechanism

One mechanism to address these problems is for a node to wait until more nodes are in range and broadcast to those nodes in one transmission. This is easily done within the wireless domain where nodes communicate over a shared medium, thus all nodes within range of a transmitting node can *hear* its transmissions. This scheme will be energy efficient because waiting for more nodes to come into range increases the number of nodes reached per transmission, thus the total number of transmissions required to reach all nodes decreases. One challenge is determining how long to wait or how many nodes to wait for before broadcasting. A solution is to set a static number k, where k represents the number of nodes in range that have not received the message and transmit only when that criteria is met. This solution can lead to deadlock though, in that after the message has reached a large percentage of the network, it becomes more difficult for nodes to satisfy this k "new" nodes requirement as the number of possible "new" nodes decreases below the k threshold.

We avoid this deadlock by relaxing the above constraint, now requiring that k nodes need to be in range and at least one has not received the broadcast message. This allows for the broadcast to continue even as the number of nodes that have not received the message decreases and also sets an upper bound on the number of transmissions to the number of nodes in the network.

The broadcast scheme depends on a threshold number of nodes in range, k, to initiate transmission. A node with a broadcast message will only send the message if it sees k other nodes in range and at least one of those has not received the message. The motivation is to reduce broadcast transmissions. If more nodes are in range then we can reach more nodes per transmission and so we expect, but do not guarantee, to make fewer transmissions to reach all nodes in the network for each broadcast message.

To find the nodes that have not received the message, we propose sending out a short request to broadcast (RTB) with the broadcast message's identification number. This message will be extremely small compared to the size of broadcast messages and will not add significant overhead. Once a single "yes" is received a transmission can be initiated and whoever needs the message can receive it. A node will respond "yes" if it needs the message, and could respond "no" if it already has the message. The transmitting node can use a timeout to decide when to stop waiting for replies.

A node can also decide to wait for "no" messages from nodes that already have the message so it can mark off internally all the nodes that the message has already visited. This can be done using a bit-array in the message header, which would be n bits long, where n is the number of nodes in the network. This can help avoid potentially wasteful RTBs, by checking the bit array against all neighbors and only making a RTB if all the neighbors are not marked off in the array. This scheme is appropriate for smaller networks since overhead will be insignificant compared to the size of the broadcast message. In larger networks where a bit-array of destinations might mean significant overhead, using a TTL is a viable alternative and is discussed at the end of this section. After this exchange of "yes" and "no" messages has occurred, the message can be broadcasted and as seen in [14], nodes that do not need the message can turn off their radios to avoid the cost of receiving redundant messages.

There are two cases to handle in our broadcasting scheme: When a message is added to the queue and when a node comes into range. Case 1 handles the broadcasting of any message just added to the queue, either from the application layer or from other nodes. If a message can immediately be broadcast, then it is, otherwise it remains in the queue and is handled later. Any messages remaining in the queue are handled by case 2 and are only in the queue because there were not appropriate neighbors to meet our broadcast criteria. When a new node comes into range, messages in the queue may be able to satisfy the condition to broadcast, so the algorithm checks and broadcasts if possible. When a node's neighbors are not changing it can stay idle, waiting to be activated by a new neighbor event.

1) A new message is generated at the node or a message is received at the node for re-broadcast:

if nodes in range $\geq k$ **then**
  make RTB for $message$
  if 'yes' response to announcement ≥ 1 **then**
    broadcast $message$
  end if
end if

2) A new node comes into range:

for all $message$ in message queue **do**
  if nodes in range $\geq k$ **then**
    make RTB for $message$
    if 'yes' response to announcement ≥ 1 **then**
      broadcast $message$
    end if
  end if
end for

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.

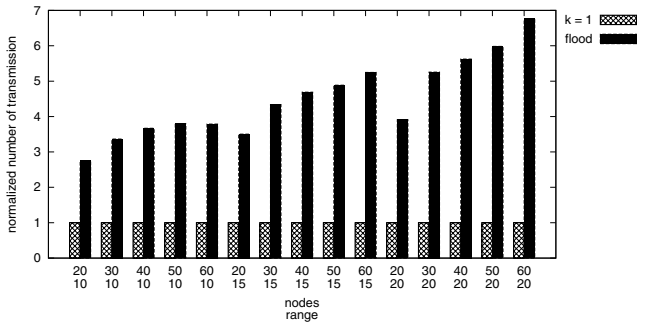


Fig. 1. Transmissions to reach 100% of network for k = 1 vs. flooding

In case 2, if more than one message needs to be broadcast, they can be queued up and broadcast sequentially.

This protocol has a natural dampening effect on broadcast messages as nodes will only receive the message once. Once all nodes have the broadcast message, there will be no more broadcasting of the message, as the condition that at least one node has not received the message can never be satisfied. A challenge within the broadcast domain is knowing when to stop broadcasting the message and when to remove it from message queues.

In the k-broadcast scheme, nodes will eventually fill up the broadcast message's known destinations bit-array to the extent required by the application (100%, 50%, etc). At this point a cleanup can be initiated. Broadcast message cleanup can proceed in various ways, one method is to send out a probe packet to clean up broadcast messages from the queues of all nodes, but that is beyond the scope of this paper. Another method is to assign a time to live (TTL), which will enforce deletion of broadcast messages from queues as data becomes invalid irrespective of percent completion.

III. Simulation

A. Set Up

We developed a discrete event simulator in Java to evaluate the performance of the k-neighbor broadcast scheme. Networks of mobile nodes were generated using a random waypoint mobility model over a set of ranges and different node populations. Care was taken to ensure the networks generated exhibited DTN characteristics of intermittent connectivity, large delays between contacts and no end-to-end paths. We evaluated how the protocol performed in a variety of network conditions. Running simulations for $k = 1, 2, 3, 4, 5$ over node counts of 20, 30, 40, 50 and 60 with ranges of 10, 15 and 20 units. Node densities were kept at a constant 0.002 nodes/unit area, while varying ranges were used to adjust connectivity. We obtained total number of transmissions and delay data for reaching 70%, 80%, 90% and 100% of the network and performed multiple runs for each network type. We took averages over 50 runs and have presented the data graphically.

B. Results

When flooding, a node will transmit to any node in range but when running our scheme with $k = 1$, a node will transmit to any node in range that has not yet received the message. This gives our scheme with $k = 1$ delay performance equal to flooding but better transmission performance as shown in Fig. 1. It is clear that our scheme performs better than flooding. It can also be seen that as networks become more dense, flooding performance gets even worse and puts more load on the network due to increase in contacts and message transmissions.

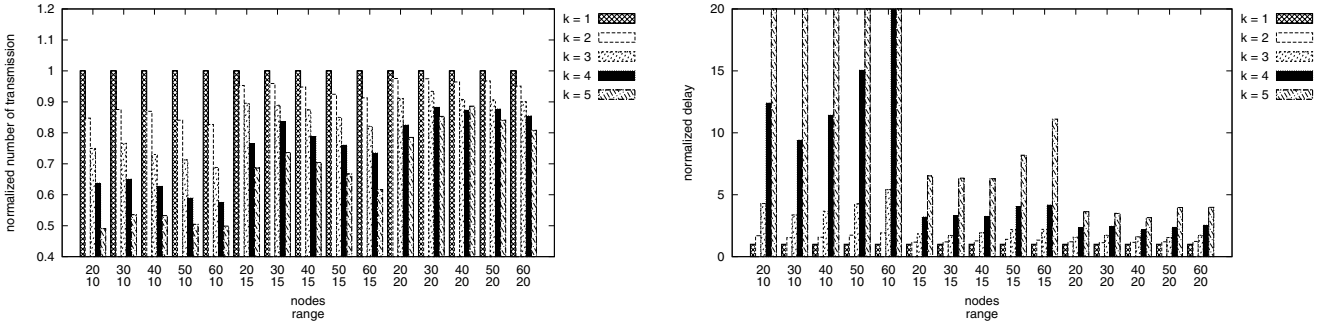
Fig. 2a shows the variation in transmissions with k. Results in the figure are normalized to the $k = 1$ case for each network. As expected, transmissions decrease as k increases. This is true regardless of the network conditions. Fig. 2b shows the variation in delay with k, again normalized to $k = 1$, as expected delay increases as k increases. Delay increases because the protocol is waiting for k neighbors to come into range. Note that networks with a range of 10 and $k = 5$ take substantially longer to reach completion, a little more than 100 times our base case when $k = 1$, and they are capped at 20 on the graphs.

It can also be seen from Fig. 2a and Fig. 2b that variation in the number of transmissions and delay is greater in less connected networks (range = 10) than in more MANET-like networks (range = 20). This is expected because with less connectivity, running the algorithm with a higher k value forces nodes to wait till enough nodes come into range. This limits transmissions in the networks with large k values. The networks with higher connectivity show less variation because nodes may have k or more neighbors at any time even with higher values of k. This higher connectivity means nodes may be transmitting to more than k nodes with a larger probability, thereby covering more nodes at once and pushing transmission performance of lower k values close to the performance of higher k values. The higher connectivity allows higher k values to transmit more often and wait less, reducing the time required to cover the network and increasing the number of transmissions to be comparable to lower values of k.

Fig. 2a reveals some less obvious results from the data. For a fixed density of nodes and fixed range, the normalized transmission performance for each k remains almost constant over all networks. This is a very useful result in determining an optimum value of k to use based on previous results for a new network. Analytical or empirical expected meeting times to see k nodes should be taken into account, if the values are too high delays may be unacceptable even though transmission efficiency increases.

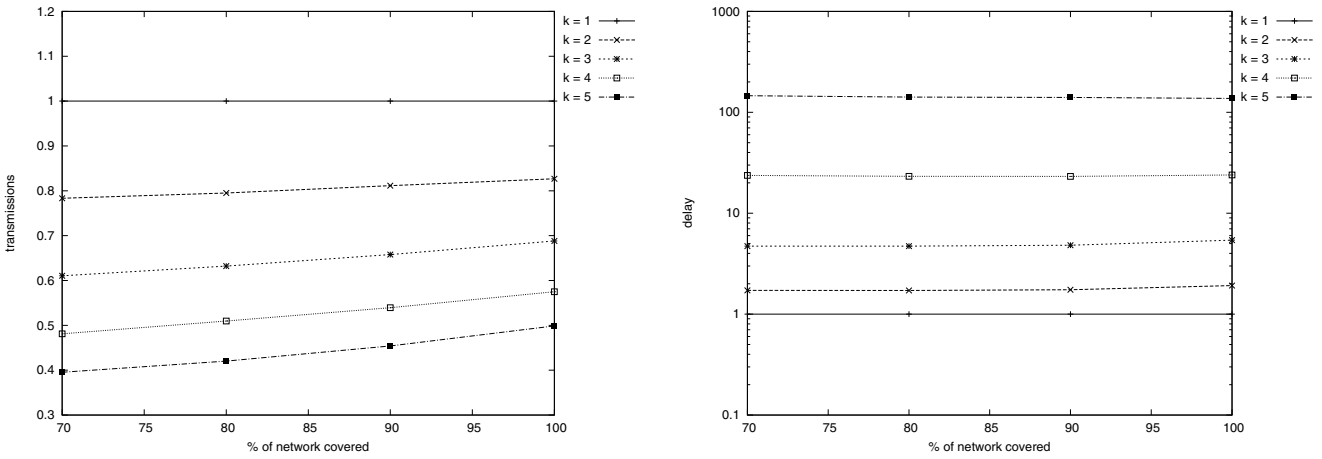
Fig. 3a and Fig. 3b show that the normalized time for broadcast over a particular network remains fairly constant for all completion percentages, but normalized number of transmissions increases for each k as the percentage of network coverage increases. The importance of this observation is in determining the value of k given a particular network coverage requirement. Smaller percentage covered requirements lead to

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.



(a) transmissions (b) delay

Fig. 2. Normalized data over all networks for all values of k



(a) transmissions (b) delay

Fig. 3. Normalized data for network with 60 nodes and range 10 for all values of k

a smaller k value which results in increased power savings.

An interesting trend is for fixed density of nodes and number of nodes, as range increases, the number of transmissions for each k tend to converge as seen in Fig. 4. Time tends to converge to the lowest value of k as expected because $k = 1$ will always be optimal with respect to delay. This is useful because if we know what kind of performance we need in a degraded state, we can set that value of k early. Initially, when a network is in a better connectivity state k will have little affect on broadcast performance, but if nodes start to fail and the network degrades, our chosen k will give us desired performance in a degraded system.

C. Anomalies

In a small percentage of runs on dense networks an interesting anomaly is observed, a strictly decreasing transmission count with strict increase in k is not seen. This can be attributed to missed opportunities in nodes waiting for enough nodes to come into range in order to meet the k-neighbor requirement. Networks with nodes *in waiting* sometime miss future opportunities to broadcast to large number of nodes than a smaller k would have reached and may also end up doing multiple single destination transmissions as the network

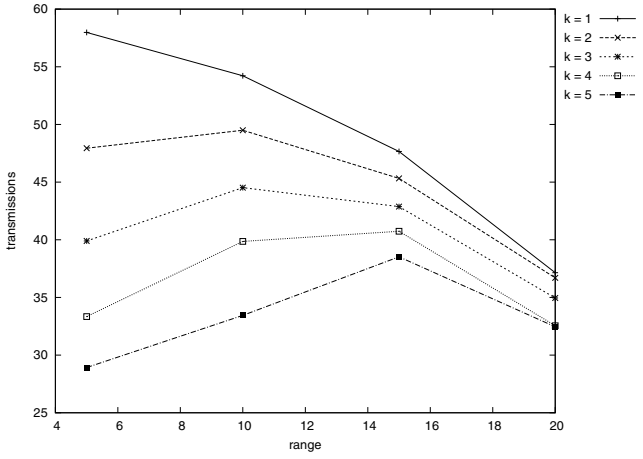


Fig. 4. Transmissions in a 60 node network for multiple ranges and all values of k

density allows for it. Fig. 5 shows how this may happen. At $t = 1$, when $k = 2$, node 1 transmits to 2 and 7, but when $k = 3$, node 1 must wait for other nodes to come into range so it does not transmit. In the next time step $t = 2$, when

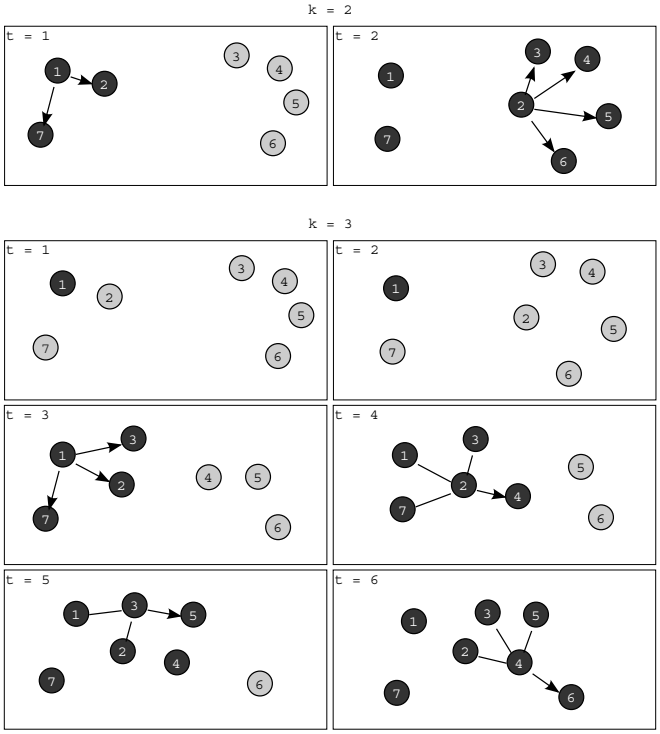


Fig. 5. The missed opportunity problem

$k = 2$, node 2 now comes into range of 4 new nodes and can transmit to all 4, but when $k = 3$, node 2 cannot make the transmission because it does not have the message. At $t = 2$ for $k = 2$, everyone in the network has the message, but for $k = 3$, only node 1 has the message. For $k = 3$, at $t = 3$, 3 nodes come into range of 1 so it transmits to 3 new destinations. For $t = 4, 5, 6$ when $k = 3$, transmitting nodes have enough neighbors to transmit but only one new neighbor to transmit to, leading the network to make one transmission at each step. This is why on occasion a network with a lower k may perform better in transmissions compared to a network with higher k.

IV. Conclusion and Future Work

In this paper an efficient broadcast scheme for DTNs is explored, where a node transmits a message when it has k other nodes in range with at least one node that has not yet received the message. We have simulated this broadcast scheme for different network sizes and k values. Our simulation results show that this k-neighbor broadcasting scheme is energy efficient. In DTNs, the objective is to reduce the number of transmissions to reach all nodes (in dense networks) or a certain percentage of nodes (in sparse networks). Intuitively for maximum power savings, setting a high k would be best, but delays may be so high that they will be unacceptable. Currently we can only estimate a good k from previous experiments.

We are working on a mechanism for varying the value of k dynamically based on the density of the network from each node's point of view thereby eliminating the need to determine the value of k beforehand. Each node would set its own k based on a previous history of node sightings. A node that sees more nodes would set its k higher than a node that does not sees fewer nodes. Since k would vary as node densities vary, the broadcast protocol would be able to automatically adapt to degrading networks where node density decreases, and it can select a k that satisfies its delay and power requirements.

References

[1] http://dtnrg.org.

[2] http://www.dtnrg.org/wiki/Code.

[3] Azzedine Boukerche. *Handbook of Algorithms for Wireless Networking and Mobile Computing (Chapman & Hall/Crc Computer & Information Science)*. Chapman & Hall/CRC, 2005.

[4] Brendan Burns, Oliver Brock, and Brian Neil Levine. Mora routing and capacity building in disruption-tolerant networks. *Ad Hoc Netw.*, 6(4):600–620, 2008.

[5] N.B. Chang and Mingyan Liu. Optimal controlled flooding search in a large wireless network. *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, 2005. WIOPT 2005. Third International Symposium on*, pages 229–237, 3-7 April 2005.

[6] Kevin 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.

[7] L. Gruenwald, M. Javed, and Meng Gu. Energy-efficient data broadcasting in mobile ad-hoc networks. *Database Engineering and Applications Symposium, 2002. Proceedings. International*, pages 64–73, 2002.

[8] Chih-Shun Hsu, Yu-Chee Tseng, and Jang-Ping Sheu. An efficient reliable broadcasting protocol for wireless mobile ad hoc networks. *Ad Hoc Netw.*, 5(3):299–312, 2007.

[9] F. Ingelrest and D. Simplot-Ryl. Localized broadcast incremental power protocol for wireless ad hoc networks. *Computers and Communications, 2005. ISCC 2005. Proceedings. 10th IEEE Symposium on*, pages 28–33, 27-30 June 2005.

[10] Gunnar Karlsson, Vincent Lenders, and Martin May. Delay-tolerant broadcasting. In *CHANTS '06: Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, pages 197–204, New York, NY, USA, 2006. ACM.

[11] Abdelmajid Khelil, Pedro José Marrón, Christian Becker, and Kurt Rothermel. Hypergossiping: A generalized broadcast strategy for mobile ad hoc networks. *Ad Hoc Netw.*, 5(5):531–546, 2007.

[12] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 151–162, New York, NY, USA, 1999. ACM.

[13] Ram Ramanathan, Richard Hansen, Prithwish Basu, Regina Rosales-Hain, and Rajesh Krishnan. Prioritized epidemic routing for opportunistic networks. In *MobiOpp '07: Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pages 62–66, New York, NY, USA, 2007. ACM.

[14] S. Singh, C. Raghavendra, and J. Stepanek. Power-aware broadcasting in mobile ad hoc networks, 1999.

[15] T. Spyropoulos, K. Psounis, and C.S. Raghavendra. Efficient routing in intermittently connected mobile networks: The single-copy case. *Networking, IEEE/ACM Transactions on*, 16(1):63–76, Feb. 2008.

[16] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259, New York, NY, USA, 2005. ACM.

[17] Ivan Stojmenovic, Mahtab Seddigh, and Jovisa Zunic. Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks. *IEEE Trans. Parallel Distrib. Syst.*, 13(1):14–25, 2002.

[18] Wenrui Zhao, Mostafa Ammar, and Ellen Zegura. Multicasting in delay tolerant networks: semantic models and routing algorithms. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 268–275, New York, NY, USA, 2005. ACM.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.