Approximation Algorithms for Data Broadcast in Wireless Networks

Rajiv Gandhi[*], Yoo-Ah Kim[†], Seungjoon Lee[‡], Jiho Ryu[§], and Peng-Jun Wan[¶]

[*]Department of Computer Science, Rutgers University – Camden, rajivg@camden.rutgers.edu
[†]Computer Science and Engineering Department, University of Connecticut, ykim@engr.uconn.edu
[‡]AT&T Labs – Research, slee@research.att.com
[§]School of Computer Science and Engineering, Seoul National University, jhryu@mmlab.snu.ac.kr
[¶]Department of Computer Science, Illinois Institute of Technology, wan@cs.iit.edu

Abstract—Broadcasting is a fundamental operation in wireless networks and plays an important role in the communication protocol design. In multihop wireless networks, however, interference at a node due to simultaneous transmissions from its neighbors makes it non-trivial to design a minimum-latency broadcast algorithm, which is known to be NP-complete. We present a simple 12-approximation algorithm for the one-to-all broadcast problem that improves all previously known guarantees for this problem. We then consider the all-to-all broadcast problem where each node sends its own message to all other nodes. For the all-to-all broadcast problem, we present two algorithms with approximation ratios of 20 and 34, improving the best result available in the literature. Finally, we report experimental evaluation of our algorithms. Our studies indicate that our algorithms perform much better in practice than the worst-case guarantees provided in the theoretical analysis and achieve up to 37% performance improvement over existing schemes.

I. INTRODUCTION

Network-wide broadcasting is a fundamental operation in wireless networks. The goal of broadcasting is to transmit a message from a source to all the other nodes in the network. Several network protocols rely on broadcasting, for example, information dissemination, service/resource discovery, or routing in multihop wireless networks. Given that key applications of multihop wireless networks include disaster relief and rescue operations, military communication, and prompt object detection using sensors, the design of low-latency broadcasting scheme is essential to meeting stringent end-to-end delay requirements for higher-level applications.

Interference is a fundamental limiting factor in wireless networks. When two or more nodes transmit a message to a common neighbor at the same time, the common node will not receive any of these messages. In such a case, we say that collision has occurred at the common node. Any communication protocol for wireless networks should contend with the issue of interference in the wireless medium.

One of the earliest broadcast mechanisms proposed in the literature is flooding [1, 2], where every node in the network transmits a message to its neighbors after receiving it. Although flooding is extremely simple and easy to implement, Ni *et al.* [3] show that flooding can be very costly and

can lead to serious redundancy, bandwidth contention, and collision: a situation known as *broadcast storm*. Since then, a large amount of research has been directed towards designing broadcast protocols which are collision-free and which reduce redundancy by reducing the number of transmissions. In this paper, we revisit the data broadcast problem and present improved algorithms that guarantee collision-free delivery and achieve low latency.

A. Our Contributions

We present algorithms for ONE-TO-ALL and ALL-TO-ALL broadcasting problems. In one-to-all broadcast, there is a source that sends a message to all other nodes in the network. In all-to-all broadcast each node sends its own message to all other nodes. Even the one-to-all broadcasting problem is known to be NP-complete [4]. For both problems, we develop approximation algorithms, which improve the previous results.

- For ONE-TO-ALL BROADCAST problem, we present a simple approximation algorithm (Section IV) that achieves a 12-approximate solution, thereby improving the approximation guarantee of 16 due to Huang *et al.* [5].
- We then consider the ALL-TO-ALL BROADCAST problem and present two algorithms (called CDA and ICDA) with approximation guarantees of 20 and 34 respectively (Section V), thereby improving the approximation guarantee of 27 by Huang et al. [6].
- We study the performance of our broadcast algorithms through simulations under various conditions. Our results indicate that our algorithms perform much better in practice than the worst case guarantees provided.

II. RELATED WORK

Several techniques have been proposed for broadcasting in wireless networks. In order to reduce the broadcast redundancy and contentions, they make use of nodes' neighborhood information and determine whether a particular node needs to transmit a message [7–14]. Gandhi *et al.* [4] show that minimizing broadcast latency in wireless networks is NP-complete and then present an approximation algorithm for one-to-all broadcasting. Their algorithm simultaneously achieves a constant approximation both for the latency as well as the number of transmissions. However, the approximation guarantee for the latency of their algorithm is greater tan 400. In this

work, we modify their algorithm to obtain a 12-approximation ratio, thereby improving their result significantly. Huang *et al.* [5] obtained a 16-approximation algorithm for one-to-all broadcasting problem. They also present an algorithm with latency at most $R + O(\log R)$ where R is the maximum Euclidean hop distance from the source to any node. However, the hidden constant in $O(\log R)$ is not small (> 150). Chen *et al.* [15] also address the problem of minimizing broadcast latency when the interference range is strictly larger than the transmission range. If α is the ratio of the interference range to the transmission range, then for $\alpha > 1$, they give an $O(\alpha^2)$-approximation algorithm. In particular, when $\alpha = 2$, their algorithm achieves a 26-approximation. However, it is not clear how their algorithm behaves when $\alpha = 1$. For all-to-all broadcast problem, Gandhi et al. [4] present a constant approximation algorithm where the constant factor is quite large (> 1000). The algorithm by Huang et al. [6] achieves the approximation factor of 27. In this work, we further further improves the approximation guarantee for the all-to-all broadcasting.

Hung *et al.* [16] provide centralized and distributed algorithms for broadcasting and experimentally study their algorithms with respect to collision-free delivery, number of transmissions and broadcast latency. While their centralized algorithm is guaranteed to be collision-free, their distributed algorithm is not. They do not provide any guarantees with respect to the number of transmissions and latency of the broadcast schedule. Williams and Camp [17] survey many wireless broadcast protocols discussed above. They provide a neat characterization and experimental evaluation of many of these protocols under a wide range of network conditions.

III. Preliminaries

A. Network Model

We model a wireless network using a unit disk graph, $G = (V, E)$. The nodes in V are embedded in the plane. Each node $u \in V$ has a unit transmission range. Let $|u, v|$ denote the Euclidean distance between u and v. Let $D(u)$ denote the neighbors of u in G. A node $v \in D(u)$ iff $|u, v| \leq 1$. We assume that time is discrete and an antenna is omni-directional. If node w hears a message transmitted by two or more neighbors at the same time, we say that there is a collision and that the transmissions interfere at w. Node w receives a message collision-free iff w hears the message without any collision.

B. Problem Statement

We are given a disk graph $G = (V, E)$ and a set of messages $M = \{1, 2, \ldots, m\}$. We also have a set of sources for these messages: $sources = \{s_j | s_j$ is the source of message $j\}$. A node can transmit message j only after it receives message j collision-free. A *schedule* specifies, for each message j and each node i, the time at which node i receives message j collision-free and the time at which it transmits message j. If a node does not transmit a message then its transmit time for that message is 0. The latency of the broadcast schedule is the first time at which every node receives all messages. The number of transmissions is the total number of times every node transmits any message. Our goal is to compute a schedule in which the latency is minimized.

IV. One-to-All Broadcast Algorithm

The algorithm takes as input a UDG $G = (V, E)$ and a source node s. The algorithm first constructs a *broadcast tree*, T_b, rooted at s in which if a node u is a parent of a node w then u is responsible for transmitting the message to w without any collision at w. It then schedules the transmissions so that every node receives the message collision-free. The two key differences from the algorithm in [4] that lead to a significantly improved approximation guarantee are:

(i) Processing the nodes in a greedy manner while constructing the broadcast tree.

(ii) Allowing a node to transmit more than once.

Both these properties are crucial to showing that our algorithm yields a 12-approximate solution. Note that in [4] the analysis of their algorithm gives an approximation ratio of at least 400.

The broadcast tree T_b, is constructed as follows. The set of nodes V is partitioned into *primary nodes* P and *secondary nodes* S (These nodes are also referred to as *dominators* and *connectors* in literature [5]). Let T_{BFS} be the breadth-first search tree rooted at s. Let $L_i, i = 0, 1, 2, \ldots, \ell$, be the set of nodes at level i in the BFS tree. P is a maximal independent set in G constructed by considering one level at a time starting from L_0 in T_{BFS}. The nodes in P form a dominating set in G, i.e., each node in S is within the transmission range of some node in P. The parent-child relationships in T_b are determined as follows. Let $P_i = P \cap L_i$ and $S_i = S \cap L_i$ be the set of primary nodes and secondary nodes respectively at level i in the BFS tree. At any level i, the algorithm first considers each node $u \in P_i$ in *non-increasing order of the number of nodes in $D(u)$* that do not have a parent yet (in [4] the nodes were processed in an arbitrary manner). The children of u in T_b are all the secondary nodes in $D(u)$ that do not have a parent when u is considered. After considering all nodes in P_i, the secondary nodes are considered in the same way (i.e., non-increasing order of the number of nodes in P_{i+1} that do not have a parent) and assigned its children. This algorithm runs in linear time, dominated by the time required to compute the BFS tree.

BroadcastTree($G = (V, E), s$)

```
1    P ← P₀ ← {s} // P is the set of primary nodes.
2    T_BFS ← BFS tree in G with root s
3    ℓ ← maximum number of levels in T_BFS // s belongs to level 0
4    for i ← 1 to ℓ do
5        L_i ← set of all nodes at level i in T_BFS; P_i ← ∅
6        for each w ∈ L_i do
7            if (P ∩ D(w) = ∅) then
8                P_i ← P_i ∪ {w}; P ← P ∪ {w}
9        S_i ← L_i \ P_i
10   P_{ℓ+1} ← ∅; S ← V \ P
11   for each node u ∈ V do
12       parent(u) ← Nil
13   for i ← 0 to ℓ do
14       P'_i ← P_i
15       while (P'_i ≠ ∅) do
```

```
16        u ←node in P'_i with maximum
              |{w|w ∈ D(u) and parent(w) = NIL}|
17        C(u) ← {w|w ∈ D(u) and parent(w) = NIL}
18        for each w ∈ C(u) do
19            parent(w) ← u
20        P'_i ← P'_i \ {u}
21     while (∃ w ∈ P_{i+1} s.t. parent(w) =NIL) do
22        u ←node in S_i with maximum
              |{w|w ∈ D(u) ∩ P_{i+1} and parent(w) = NIL}|
23        C(u) ← {w| w ∈ D(u)∩ P_{i+1} and parent(w) = NIL}
24        for each w ∈ C(u) do
25            parent(w) ← u
26     V_b ← V; E_b ← {(u,w)|u = parent(w)}
27     return T_b = (V_b, E_b)
```

In ONE-TO-ALL BROADCAST, the transmissions are scheduled in two phases. In Phase 1, the algorithm schedules transmissions only to the nodes in set (denoted by X) which contains all primary nodes and non-leaf secondary nodes in T_b. In Phase 2, transmissions are scheduled to send the message to all other nodes. *Note that this leads to some nodes transmitting more than once* which is again a significant departure from the algorithm in [4] in which each node transmits the message at most once. The intuition behind this is that it is not necessary to send a message to terminal nodes early as they are not responsible for relaying the message further. On the other hand, by reducing the number of recipients in the first phase, a node need to avoid less conflicts thus sending a message to nonterminal nodes quickly.

In Phase 1, nodes are considered one level at a time starting from L_0. Only those primary nodes that have a child in X will transmit the message in this phase. Clearly, for any primary node u if $C(u) \neq \emptyset$ and $C(u) \cap X = \emptyset$ then u will transmit the message in Phase 2. At any level L_i, the secondary nodes are scheduled for transmission only after all transmissions of primary nodes in P_i have been scheduled. While scheduling transmissions, the nodes in P_i as well as $S_i \cap X$ are considered in non-increasing order of the number of their children in T_b. The algorithm then follows a greedy strategy to schedule the collision-free transmissions to nodes in X. Any transmitting node, u, transmits at the minimum time t that satisfies the following collision-free constraints – (i) u must have received the message collision-free before time t, (ii) no node in $C(u) \cap X$ is hearing any transmissions at time t, (iii) no node in $D(u) \cap X$ is receiving the message collision-free at time t.

In Phase 2, transmissions are scheduled so that the nodes in $Y = V \setminus X$ receive the message. Nodes are considered one level at a time. For each $v \in Y$, $parent(v)$ is responsible for transmitting the message collision-free to v. Since $P \cap Y = \emptyset$, the secondary nodes do not transmit in Phase 2. Any transmitting node, u, transmits at the minimum time t that satisfies the above three collision-free constraints.

ONE-TO-ALL BROADCAST(G, T_{BFS}, T_b, s)

```
1    for each node u ∈ V do
2        trTime_1(u) ← 0; trTime_2(u) ← 0
3    X ← P ∪ {w ∈ S | C(w) ≠ ∅} // set of transmitters
4    Y ← V \ X // set of terminals
5    // Phase 1 - transmitters will receive the message
6    for i ← 0 to ℓ - 1 do
7        P'_i ← {u ∈ P_i | ∃ w ∈ C(u) ∩ X}
8        X_i ←nodes in P'_i ∪ (X ∩ S_i) with all primary nodes ordered
```

```
             before the secondary nodes and the primary and secondary
             nodes listed in the order they were chosen in lines 19 and
             25 resp. in BROADCASTTREE.
9        while (X_i ≠ ∅) do
10           u ← first node in X_i
11           I_1(u) ← {t| ∃ w ∈ C(u) \ Y that hears a message at time t}
12           I_2(u) ← {t| ∃ w ∈ D(u) \ Y that receives
                          a message coll-free at time t}
13           I(u) ← I_1(u) ∪ I_2(u)
14           trTime_1(u) ← min{t| t > rcvTime(u) and t ∉ I(u)}
15           for each w ∈ C(u) \ Y do
16               rcvTime(w) ← trTime_1(u)
17           X_i ← X_i \ {u}
     // Phase 2 - the terminals will receive the message
18   Y' = Y
19   for i ← 0 to ℓ do
20       for each u ∈ S_i ∩ Y' do
21           v ← parent(u)
22           I_1(v) ← {t| ∃ w ∈ C(v) ∩ Y' that hears a message at time t}
23           I_2(v) ← {t| ∃ w ∈ D(v) that receives
                          a message coll-free at time t}
24           I(v) ← I_1(v) ∪ I_2(v)
25           trTime_2(v) ← min{t| t > rcvTime(v) and t ∉ I(v)}
26           for each u ∈ C(v) ∩ Y' do
27               rcvTime(u) ← trTime_2(v)
28       Y' ← Y' \ C(v)
29   return trTime_1, trTime_2
```

Theorem 4.1: Our algorithm gives a 12-approximate solution for the latency. The number of transmissions in our algorithm is at most 21 times those in an optimal solution. For complete proofs, we refer to the full version [18].

V. ALL-TO-ALL BROADCAST ALGORITHM

We now consider all-to-all broadcasting, in which each node v has a message $m(v)$ to send to all other nodes. In this section, we present two algorithms. The first one achieves an approximation guarantee of 20 by obtaining a tighter lower bound and adopting efficient scheduling algorithms for data collection and pipelined broadcasting. This improves the best known guarantee of 27 in the literature [6]. The second algorithm achieves an approximation factor of 34, which performs well in our experiments (Section VI).

A Lower Bound. Let G be a unit-disk graph with n nodes. Denote by γ_c the connected domination number of G. That is, $\gamma_c = |C(G)|$ where $C(G)$ is a minimum size connected dominating set of G. Then we have the following *lower bound* on the latency of all-to-all broadcasting.

Lemma 5.1: The minimum latency of all-to-all broadcasting in G is at least $n - 1 + \gamma_c$.

A. Collect-and-Distribute Algorithm (CDA)

The *graph radius* of G with respect to a node v is the maximum depth of the BFS tree rooted at v. A *graph center* of G is a node in G with respect to which the graph radius of G is the smallest. Let s be a graph center of G, and R be the graph radius of G with respect to s. Clearly, $\gamma_c \geq R$. We call transmissions of message m from a node v *upward* if the message m is from the descendant of v. Otherwise, a transmission is called *downward*. Our schedule consists of two phases. In Phase 1, s collects all the packets by performing upward transmissions. In the Phase 2, s broadcasts all the n packets to all other nodes via downward transmissions.

Phase 1. Node s collects all messages by modifying a data dissemination algorithm by Florens and McEliece [19]. In the data dissemination problem, a source node sends a different message $m(v)$ to each node v in the network, and they present a 3-approximation algorithm. To collect messages in Phase 1, we can simply schedule transmissions in a reverse order of data dissemination schedule and also reverse the direction of the transmissions. It requires at most 3 times the optimal collection time.

Phase 2. We construct a broadcast tree T_b using BROAD-CASTTREE in Section IV. Next, we describe transmission scheduling algorithm. In the algorithm by Gandhi *et al.* [4], the root node collects all messages and perform one-to-all broadcasting for each message. The root node needs to wait until the previous message reaches level 3 (L_3) before initiating a broadcast for another message to make sure there are no conflicts in their algorithm. In our algorithm, we find a schedule by a vertex coloring, which make sure that all the nodes with the same color can broadcast a message without conflicts, and show that 17 colors are enough to obtain a conflict-free schedule.

Let H_1 (resp., H_2) be the graph over the primaries (resp., secondaries) in which there is an edge between two primaries (resp., secondaries) if and only if one of them has a child adjacent to the other in G.

The scheduling for H_1 can be done by computing a vertex coloring of H_1 in the first-fit manner in the smallest-degree-last ordering. By proper renumbering of the colors, we assume that s has the first color. Let k_1 be the number of colors used by this coloring. Then, $k_1 \leq 12$ [5].

We compute a vertex coloring of H_2 in the first-fit manner by considering nodes in the same order as used when computing the broadcast tree T_b, and let k_2 be the number of colors. Then, $k_2 \leq 5$ [5].

Let $k = k_1 + k_2$. We define a *superstep* to be a group of consecutive k time slots. In each superstep, the first k_1 slots are for scheduling transmissions from primaries, and the remaining k_2 slots will be for secondaries. Each primaries (resp., secondaries) with color i is only allowed to transmit in the i-th slot of a primaries (resp., secondaries) slot in a superstep. The source node s transmits one packet in each superstep. Each connector receiving a packet in a superstep transmits the received packet in the corresponding connector slot in the *same superstep*. Each primaries receiving a packet in a connector slot transmits the received packet in a primary slot of the *subsequent superstep*. Note that any message that the primaries at level i received in a given superstep will be forwarded to the primaries at level $i + 1$ or $i + 2$ in the next superstep. Therefore, a message which has been sent from a source will be broadcast to all nodes within R supersteps where R is the number of levels in the broadcast tree.

Theorem 5.2: Our all-to-all broadcasting algorithm gives a 20-approximation.

Proof Sketch: Recall that the first phase takes at most $3(n - 1)$ time slots. The second phase takes no more than $17(n - 1 + R)$ time steps as in $n - 1 + R$ supersteps, all n messages are broadcast and each superstep consists of 17 time slots. Therefore, the total latency of our all-to-all broadcast schedule is at most $3OPT + 17(n - 1 + R) < 20OPT$.

B. Interleaved Collect-and-Distribute Algorithm (ICDA)

We now describe an algorithm in which all nodes participate in broadcasting as soon as possible so as to minimize the broadcast time. The main idea is as follows. Suppose that a node v receives a message $m(x)$ forwarded originally from its descendant x in the broadcast tree and relays it to its parent to deliver the message to the root node s. Note that the children of v can also receive the message while v broadcast it and therefore, they can initiate broadcasting $m(x)$ in their own subtrees in parallel without waiting for the message forwarded from s.

We schedule transmissions for each node as follows. As in CDA, we define a *superstep* but in a slightly different way. That is, in each superstep, every node transmits at most one message (either upward or downward) if there is any message that the node received but not sent. Instead of finishing all upward transmissions first, we mix upward or downward transmissions in each superstep with preferences given to upward transmissions. Also for an upward transmission, a node should make sure that its parent and *all of its children* (except the one which sent the message to v) receive the message. For a downward transmission, v is responsible for sending the message to all of its children. We find a schedule using the following rules.

(1) All terminal nodes X send messages to their parents first.
(2) In each superstep, primaries are scheduled before secondaries.
(3) Transmissions from primaries are scheduled based on a vertex-coloring of H_1. and the order to process nodes are the same as in CDA.
(4) Each node can receive at most one upward message from its children. Therefore, a node can perform an upward transmission only if its parent has not received an upward message in the superstep. Otherwise, it performs a downward transmission.
(5) For secondaries, the nodes are considered in the same order as the broadcast tree is constructed. Upward transmissions are scheduled before downward transmissions.
(6) Once all upward transmissions for nodes are scheduled, the remaining nodes perform downward transmissions if there is any message to be sent.

This algorithm yields a 34-approximation [18]. Even though the theoretical bound of ICDA is weaker than that of CDA, the experimental results (Section VI) show that it provides comparable performance as CDA. In fact, for large networks (300 nodes or more), ICDA performs better than CDA.

VI. EXPERIMENTAL EVALUATION

A. Simulation Setup

We place wireless nodes in a square (e.g., 1000m by 1000m) uniformly at random, while varying the number of nodes and the size of square. We use a fixed transmission range of 200m and assume two nodes can communicate if they are within the transmission range of each other. For one-to-all broadcast experiments, we select a source uniformly at random, and the
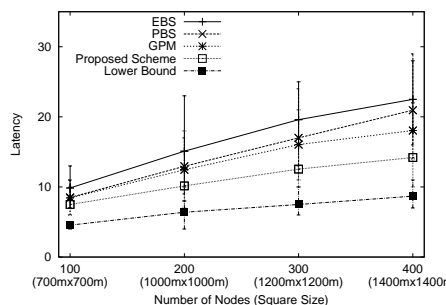
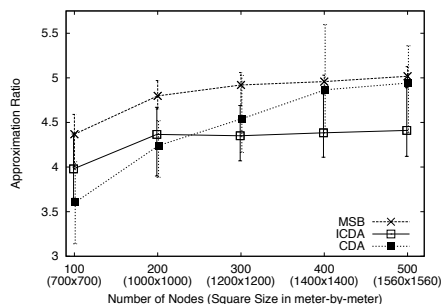**Figure 1**. Average latency of one-to-all broadcast.

Figure 2. Average approximation ratio of all-to-all broadcast.

source becomes the root of broadcast tree. For each scenario, we use 20 runs with different random seeds and present the average along with an error bar displaying the minimum and maximum. We focus on the latency metric although we briefly report the results on the number of transmissions.

We compare our proposed algorithms with existing schemes. Gandhi, Parthasarathy, and Mishra [4] present a one-to-all algorithm (which we call GPM) and an all-to-all algorithm (which we call MSB). We also compare our one-to-all algorithm with Enhanced Broadcast Scheduling (EBS) and Pipelined Broadcast Scheduling (PBS) by Huang *et al.* [5]. Although we have experimented with various settings, we only report a set of representative results due to space constraint.

B. Results for One-to-All Broadcast

In Figure 1, we present the average latencies for broadcast algorithms as well as the height of BFS tree (i.e., lower bound), when we vary both the number of nodes and the square size to keep the node density similar. We observe that our proposed algorithm consistently outperforms existing schemes by 11–37%. While the BFS tree height increases as the square size increases, our proposed scheme maintains the approximation ratio around 1.65. Although our analytical bound is 12, the performance of our scheme in practice is much closer to optimal. In terms of packet transmissions, our proposed scheme performs very similar to GPM and EBS and consistently outperforms PBS (by up to 22%).

C. Results for All-to-All Broadcast

In Figure 2, we present the average approximation ratio of our all-to-all broadcast schemes (CDA and ICDA) and MSB

when we vary the number of nodes and the square size. In all schemes, the ratio increases as the network becomes larger. CDA performs well when the network size is small (e.g., around 17% better than MSB in the 100-node case). However, the performance difference between CDA and MSB becomes smaller in larger networks. This is because both schemes first send all packets to the root before distributing them, and the initial latency increases with the network size growth. Although ICDA does not perform as well as CDA for small networks, it consistently outperforms MSB by 7–11% in all cases. In fact, the performance gap between ICDA and MSB becomes larger as the network size grows, which indicates the benefit of interleaved transmissions. Again, the performance of CDA and ICDA in practice is much better than the analytical bound (20 and 34).

REFERENCES

[1] C. Ho, K. Obraczka, G. Tsudik, and K. Viswanath, "Flooding for reliable multicast in multi-hop ad hoc networks," in *Proc. of the Int. Work. on Disc. Alg. and Meth. for Mobile Comp. and Comm.*, 1999, pp. 64–71.

[2] J. Jetcheva, Y. Hu, D. Maltz, and D. Johnson, "A simple protocol for multicast and broadcast in mobile ad hoc networks," July 2001.

[3] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proc. of the 5th Ann. ACM/IEEE Int. Conf. on Mobile Comp. and Net.*, 1999, pp. 151–162.

[4] R. Gandhi, A. Mishra, and S. Parthasarathy, "Minimizing broadcast latency and redundancy in ad hoc networks," *IEEE/ACM Transactions on Networking*, pp. 840–851, Aug 2008.

[5] S. Huang, P. Wan, X. Jia, H. Du, and W. Shang, "Minimum-latency broadcast scheduling in wireless ad hoc networks," in *Proceedings of INFOCOM*, 2007, pp. 733–739.

[6] S. C.-H. Huang, H. Du, and E.-K. Park, "Minimum-latency gossiping in multi-hop wireless networks," in *Proceedings of MobiHoc*. New York, NY, USA: ACM, 2008, pp. 323–330.

[7] H. Lim and C. Kim, "Multicast tree construction and flooding in wireless ad hoc networks," in *Proc. of the 3rd ACM Int. Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Sys*, 2000, pp. 61–68.

[8] W. Peng and X.-C. Lu, "On the reduction of broadcast redundancy in mobile adhoc networks," in *Proceedings of MobiHoc*, Aug. 11, 2000.

[9] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying: An efficient technique for flooding in mobile wireless networks," INRIA, Tech. Rep. Research Report RR-3898, Feb. 2000.

[10] W. Peng and X. Lu, "Ahbp: An efficient broadcast protocol for mobile adhoc networks," *J. of Science and Tech. - Beijing, China*, 2000.

[11] J. Sucec and I. Marsic, "An efficient distributed network-wide broadcast algorithm for mobile adhoc networks," Rutgers U., Tech. Rep., 2000.

[12] W. Peng and X. Lu, "Efficient broadcast in mobile ad hoc networks using connected dominating sets," *J. of Software - Beijing, China*, 1999.

[13] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks," *IEEE Trans. on Parallel and Distributed Systems*, pp. 14–25, Jan 2002.

[14] L. Orecchia, A. Panconesi, C. Petrioli, and A. Vitaletti, "Localized techniques for broadcasting in wireless sensor networks," in *Proceedings of DIALM-POMC*, New York, NY, USA, 2004, pp. 41–51.

[15] Z. Chen, C. Qiao, J. Xu, and T. Lee, "A constant approximation algorithm for interference aware broadcast in wireless networks," in *Proceedings of INFOCOM*, 2007, pp. 740–748.

[16] P.-K. Hung, J.-P. Sheu, and C.-S. Hsu, "Scheduling of broadcasts in multihop wireless networks," in *European Wireless*, 2002.

[17] B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks," in *Proc. of the 3rd ACM Int. Symp. on Mobile Ad hoc Net. and Comp.*, 2002, pp. 194–205.

[18] R. Gandhi, Y.-A. Kim, S. Lee, J. Ryu, and P.-J. Wan, "Approximation algorithms for data broadcast in wireless networks," CSE Technical Report BECAT/CSE-TR-09-01, University of Connecticut, http://www.becat.uconn.edu/reports/BECAT-CSE-TR-09-01.pdf, 2009.

[19] C. Florens and R. McEliece, "Packets distribution algorithms for sensor networks," in *INFOCOM*, 200, pp. 1063 – 1072.