Local Broadcast Algorithms in Wireless Ad Hoc Networks: Reducing the Number of Transmissions

Majid Khabbazian, Ian F. Blake *Fellow, IEEE*, Vijay K. Bhargava, *Fellow, IEEE*

***Abstract*— There are two main approaches, static and dynamic, to broadcast algorithms in wireless ad hoc networks. In the static approach, local algorithms determine the status (forwarding/non-forwarding) of each node proactively based on local topology information and a globally known priority function. In this paper, we first show that local broadcast algorithms based on the static approach cannot achieve a good approximation factor to the optimum solution (an NP-hard problem). However, we show that a constant approximation factor is achievable if (relative) position information is available. In the dynamic approach, local algorithms determine the status of each node "on-the-fly" based on local topology information and broadcast state information. Using the dynamic approach, it was recently shown that local broadcast algorithms can achieve a constant approximation factor to the optimum solution when (approximate) position information is available. However, using position information can simplify the problem. Also, in some applications it may not be practical to have position information. Therefore, we wish to know whether local broadcast algorithms based on the dynamic approach can achieve a constant approximation factor without using position information. We answer this question in the positive - we design a local broadcast algorithm in which the status of each node is decided "on-the-fly" and prove that the algorithm can achieve both full delivery and a constant approximation to the optimum solution.**

***Index Terms*— mobile ad hoc networks, distributed algorithms, broadcasting, connected dominating set, constant approximation.**

I. Introduction

Wireless ad hoc networks have emerged to support applications, in which it is required/desired to have wireless communications among a variety of devices without relying on any infrastructure or central management. In ad hoc networks, wireless devices, simply called nodes, have limited transmission range. Therefore, each node can directly communicate with only those within its transmission range (i.e., its neighbors) and requires other nodes to act as routers in order to communicate with out-of-range destinations.

One of the fundamental operations in wireless ad hoc networks is broadcasting, where a node disseminates a message to all other nodes in the network. This can be achieved through flooding, in which every node transmits the message after receiving it for the first time. However, flooding can impose a large number of redundant transmissions, which can result in significant waste of constrained resources such as bandwidth and power. In general, not every node is required to forward/transmit the message in order to deliver it to all nodes in the network. A set of nodes form a Dominating Set (DS) if every node in the network is either in the set or has a neighbor in the set. A DS is called a Connected Dominating Set (CDS) if the subgraph induced by its nodes is connected. Clearly, the forwarding nodes, together with the source node, form a CDS. On the other hand, any CDS can be used for broadcasting a message (only nodes in the set are required to forward). Therefore, the problems of finding the minimum number of required transmissions (or forwarding nodes) and finding a Minimum Connected Dominating Set (MCDS) can be reduced to each other. Unfortunately, finding a MCDS (and hence minimum number of forwarding nodes) was proven to be NP hard even when the whole network topology is known [1], [2]. A desired objective of many efficient broadcast algorithms is to reduce the total number of transmissions to preferably within a constant factor of its optimum. For local algorithms and in the absence of global network topology information, this is commonly believed to be very difficult or impossible [3], [4].

The existing local broadcast algorithms can be classified based on whether the forwarding nodes are determined statically (based on only local topology information) or dynamically (based on both local topology and broadcast state information) [5]. In the static approach, the distinguishing feature of local algorithms over other broadcast algorithms is that using local algorithms any local topology changes can affect only the status of those nodes in the vicinity. Therefore, local algorithms can provide scalability as the constructed CDS can be updated, efficiently. The existing local algorithms in this category typically use a priority function known by all nodes in order to determine the status of each node [5]. In this paper we show that, using only local topology information and a globally known priority function, the local broadcast algorithms based on the static approach are not able to guarantee a good approximation factor to the optimum solution (i.e., MCDS). On the other hand, we show that local algorithms based on the static approach can achieve interesting results such as a constant approximation factor and shortest path preservation if the nodes are provided with position information.

In the dynamic approach, the status of each node (hence the CDS) is determined "on-the-fly" during the broadcast progress. Using this approach, the constructed CDS may vary from one broadcast instance to another even when the whole network topology and the source node remain unchanged. Consequently, the broadcast algorithms based on the dynamic approach typically have small maintenance cost and are expected to be robust against node failures and network topology changes. Many local broadcast algorithms in this category use local neighbor information to reduce the total number of transmissions and to guarantee full delivery (assuming no

Digital Object Indentifier 10.1109/TMC.2011.67

loss at the MAC/PHY layer). Others, such as probability-based and counter-based algorithms [6]–[8], do not rely on neighbor information. These algorithms typically cannot guarantee full delivery but eliminate the overhead imposed by broadcasting "Hello" messages or exchanging neighbor information.

Many of the existing neighbor-information-based broadcast algorithms in this category can be further classified as neighbor-designating and self-pruning algorithms. In neighbor-designating algorithms [9]–[11], each forwarding node selects some of its local neighbors to forward the message. Only the selected nodes are then required to forward the message in the next step. For example, a forwarding node u may select a subset of its 1-hop neighbors such that any 2-hop neighbor of u is a neighbor of at least one of the selected nodes [9]. In self-pruning algorithms [3], [12], [13], on the other hand, each node decides by itself whether or not to forward a message. The decision is made based on a self-pruning condition. For example, a simple self-pruning condition employed in [12] is whether all neighbors have been covered by previous transmissions. In other words, a node can avoid forwarding/rebroadcasting a message if all of its neighbors have received the message by previous transmissions.

In [14], it was shown that neither neighbor-designating nor self-pruning algorithms can guarantee both full delivery and a constant approximation if they use only 1-hop neighbor information and do not piggyback information into the broadcast packets. The authors then proposed a self-pruning algorithm based on partial 2-hop neighbor information and proved that the algorithm achieves a constant approximation to the optimum solution and guarantees full delivery. However, in their proposed algorithm, each node was assumed to have its (approximate) position information, which is not practical in some applications/scenarios. Also, having position information can provide non-trivial information in wireless ad hoc networks and can greatly simplify the problem. As such, we wish to know whether similar results can be obtained without using position information. In this paper, we answer this remaining question in the positive - we propose a local broadcast algorithm based on 2-hop neighbor information and prove that it guarantees a constant approximation to the optimum solution. The proposed algorithm is both neighbor-designating and self-pruning, i.e. the status of each node is determined by itself and/or other nodes. In particular, using our proposed algorithm, each broadcasting node selects at most one of its neighbors to forward the message. If a node is not selected to forward, it has to decide, on its own, whether or not to forward the message.

The rest of the paper is organized as follows. In Section II, we describe our system model and assumptions. In Sections III and IV we analyze the power of local broadcast algorithms based on the static and dynamic approach, respectively. In Section V, we use simulation to confirm the analytical results presented in Section IV. Finally, we conclude the paper in Section VI. Most of the proofs are placed in the appendix.

II. System Model and Assumptions

We assume that the network consists of a set of nodes V, $|V| = N$. Each node is equipped with omnidirectional antennas. Every node $u \in V$ has a unique *id*, denoted $id(u)$, and every packet is stamped by the *id* of its source node and a nonce, a randomly generated number by the source node. For simplicity, we assume that all nodes are located in two-dimensional space. However, all the results presented in this paper can be readily extended to three-dimensional ad hoc networks.

To model the network, we assume two different nodes $u \in V$ and $v \in V$ are connected by an edge if and only if $|uv| \leq R$, where $|uv|$ denotes the Euclidean distance between nodes u and v and R is the transmission range of the nodes. Thus, we can represent the communication graph by $G(V, R)$, where V is the set of nodes and R is the transmission range. This model is, up to scaling, identical to the unit disk graph model, which is a typical model for two-dimensional ad hoc networks. In reality, however, the transmission range can be of arbitrary shape as the wireless signal propagation can be affected by many unpredictable factors. Finally, we assume that the network is connected and static during the broadcast and that there is no loss at the MAC/PHY layer. These assumptions are necessary in order to prove whether or not a broadcast algorithm can guarantee full delivery. Note that without these assumptions even flooding cannot guarantee full delivery.

III. Broadcasting Using the Static Approach

Let the *k-neighborhood* of a node u, denoted $G_k(u)$, be the subgraph induced by u and nodes at most k hops away from u. Suppose each node is given a globally fixed priority function $Pr(id(w), G_{h'}(w))$ which gets a node's *id*, $id(w)$, and its local topology information, $G_{h'}(w)$, as inputs and returns a real number that determines the priority of w. For example, priority of a node can be determined by its *id*, by its degree (i.e. the number of its 1-hop neighbors) or by its neighbor connectivity ratio (i.e. ratio of pairs of neighbors that are not directly connected to all possible pairs of neighbors).

In local broadcast algorithms based on the static approach, the status (forwarding/non-forwarding) of each node u, $Stat(u)$, is a function of $id(u)$, $G_h(u)$ and $Pr(id(v), G_{h'}(v))$, where $v \in G_h(u)$ and the parameters h and h' are fixed constant numbers[1]. Note that the status of each node does not depend on that of other nodes. Therefore, any local topology change can only affect the status of the nodes in the vicinity. In designing local broadcast algorithms, we are looking for status functions that not only guarantee constructing a CDS (hence full delivery) but also ensure that the constructed CDS has small size, preferably within a constant factor of the optimum. In the following, we show that no such status function exists. The idea is to find a graph in which any status function fails either in constructing a CDS or finding a CDS whose size is smaller than $\Omega(N)$, where N is the total number of nodes in the network. Our approach is to construct a graph with a large number of nodes for which both the local topology, $G_h(.)$, and the relative priority of the nodes in $G_h(.)$ are the same.

Without loss of generality, we can assume $R = 1$. As shown in Figure 1, let us distribute N nodes on the x-axis between

[1]For the sake of generality, we consider two separate parameters h and h'.

the coordinates 0 and $2(h+h')+1$ such that the coordinate of the ith node is $\frac{(i-1)}{(N-1)} \times (2(h+h')+1)$, where $1 \leq i \leq N$ and $N \gg 2(h+h')+1$. It is easy to see that $G_{h'}(u)$ and $G_{h'}(v)$ are isomorphic if $u, v \in [h', 2h+h'+1]$. Based on the definition of priority function, the relative priority of two nodes u and v only depends on their *id*s if $G_{h'}(u)$ and $G_{h'}(v)$ are isomorphic. Therefore, we can distribute all nodes in the interval $[h', 2h+h'+1]$ such that their priorities increase as their coordinates (their distance to the origin) increase. Using this distribution, all nodes in the unit interval $[h'+h, h'+h+1]$ will have similar views of the local topology $G_h(.)$ and priority relationship between the nodes in $G_h(.)$. Therefore, the output of the status function is the same for all nodes in this unit interval, i.e., either all or none of the nodes in the unit interval will be selected. Clearly, the latter case is not possible since at least one node in every unit interval has to be selected (otherwise the graph induced by the selected nodes will be disconnected). On the other hand, selecting all nodes in the interval $[h'+h, h'+h+1]$ will result in a large set of selected nodes. Note that every MCDS consists of at most two nodes in each unit interval. Therefore, for the simple network shown in Figure 1, we have $|MCDS| \leq 2 \times (2(h+h')+1)$, where $|MCDS|$ denotes the size of a MCDS. [2] If all nodes in a unit interval are selected, the size of the obtained CDS would be at least

$$\lfloor \frac{N-1}{2(h+h')+1} \rfloor + 2(h+h'-1).$$

Since h and h' are constant, the approximation factor would be at least

$$\frac{\lfloor \frac{N-1}{2(h+h')+1} \rfloor + 2(h+h'-1)}{2 \times (2(h+h')+1)} \in \Omega(N).$$

Consequently, local broadcast algorithms based on the static approach are not able to guarantee a good approximation factor in the worst case. Note that this result does not imply that local broadcast algorithms cannot achieve a good bound on average.

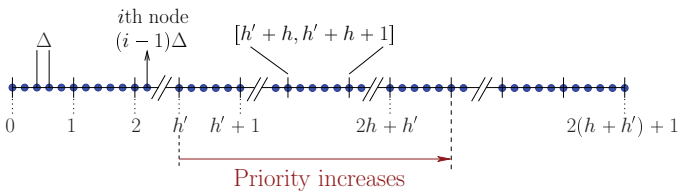

Fig. 1. Distributing nodes on a line segment with length $2(h+h')+1$.

A. Using Position Information

In the context of broadcast algorithms based on the static approach, we may wish to know whether using position information can help us to yield a small approximation factor. In this section, we show that a constant approximation factor is achievable if position information is available. To show this, we partition the network area into square cells as illustrated in Figure 2 and present simple local algorithms that select a constant number of nodes in each cell.

The approach of network partitioning to use geographical information has been used for different purposes including saving energy in sensor networks, and handling mobility in broadcast algorithms for ad hoc networks. In the primary work by Xu et al. [15], the authors propose an algorithm that partitions the network area into square cells. To save energy, at each time, the algorithm selects one node in each cell as active and puts other nodes in the cell to sleep. The set of active nodes must form a CDS in order to maintain network connectivity and coverage. To guarantee this, the proposed algorithm in [15] assumes that the side length of each square cell is $\frac{R}{\sqrt{5}}$, where R is the radio transmission range. However, this is not sufficient to guarantee connectivity, because some square cells may not contain any node[3]. In this section, we show under what conditions the set of nodes constructed by selecting one node in each non-empty set form a CDS. We also show that these conditions can be relaxed at the price of selecting more than one yet a constant number of nodes in some non-empty cells. Therefore, the result of this section can be used to extend the work in [15] to the case where some cells may contain no (working) sensor due to, for example, a non-uniform distribution of sensors, sensor mobility or sensor failure.

In [16], authors use a network partitioning approach in designing broadcast protocols for *dense* mobile ad hoc networks. To handle mobility, the proposed broadcast protocols in [16] rely on the distribution of nodes as opposed to the majority of existing broadcast protocols that rely on the frequently changing communication topology. In this section, we focus on reducing the number of transmissions in a general static ad hoc network. Readers may refer to [16] for a discussion on how to handle mobility when network partitioning is used.

Theorem 1: Let $\alpha > 0$ and $m \geq 1$ be constant numbers. Let $\mathcal{S}$ be a CDS. Suppose there are at most m nodes of $\mathcal{S}$ in any square of size $\alpha R \times \alpha R$. Then, we have

$$|\mathcal{S}| \leq (m\lceil \frac{2}{\alpha} \rceil^2)|MCDS|,$$

that is, the size of $\mathcal{S}$ is within a constant factor of $|MCDS|$. For proof see Appendix.)

As shown in Figure 2, assume that all nodes in the network are located in a square area of size $L \times L$. Considering Theorem 1, it is natural to partition the network area into small square cells of size $\alpha R \times \alpha R$, where α is a positive real number, and search for algorithms that guarantee a constant number of selected nodes in each cell. One simple algorithm is to select exactly one node in each non-empty cell. A node can be selected using local neighbor information as follows: Suppose each node knows the *id* and position of itself and its 1-hop neighbors. Also, assume that $\alpha \leq \frac{\sqrt{2}}{2}$. Therefore, the maximum distance between any two points in a cell is at most R, hence all the nodes in a cell are 1-hop neighbors of each other. Thus, a node can verify whether or not it has the minimum *id* in the cell in which it is located. Consequently,

[2] To be exact, $|MCDS| = \left\lceil \frac{l-1}{\lfloor \frac{N-1}{l} \rfloor \times \frac{l}{N-1}} \right\rceil$, where $l = 2(h+h')+1$.

[3] In [15], the authors implicitly assume that there exists at least one node in each square cell. This assumption guarantees connectivity.

in every cell, using local neighbor information, the node with minimum *id* can select itself. Note that the set of selected nodes form a dominating set as every node in the network is either in the set or within the transmission range of the selected node in its cell.

Let α be a positive real number. Let $S_{sq}(\alpha)$ denote a set constructed by dividing the network area into small square cells of size $\alpha R \times \alpha R$ (as shown in Figure 2) and selecting exactly one node in each non-empty cell. As explained earlier, the set $S_{sq}(\alpha)$ is a dominating set (DS) if $\alpha \leq \frac{\sqrt{2}}{2}$. Therefore, we use the notation $DS_{sq}(\alpha)$ instead of $S_{sq}(\alpha)$, wherever $\alpha \leq \frac{\sqrt{2}}{2}$. In general, the graph induced by $DS_{sq}(\alpha)$ is not connected even for small values of α. Nevertheless, it can be proven that the graph induced by $DS_{sq}(\alpha)$ is connected (hence, $DS_{sq}(\alpha)$ is a CDS) if the network satisfies any of the two conditions that we call *high-connectivity* condition and *high-transmission* condition.

We say a network satisfies the *high-connectivity* condition if there exists a constant c, $0 < c < 1$, such that the network remains connected if the transmission ranges of all nodes is reduced from R to cR. Dense networks typically satisfy the high-connectivity condition. For example, when the density of nodes is high, it is expected that the network remains connected if all nodes reduce their transmission range to, say, 90% of the original. Notice that the high-connectivity condition does not require nodes to reduce their transmission ranges. It just states that the network remains connected if the transmission range of nodes is reduced by a constant.

The *high-transmission* condition is an alternative condition that can guarantee the constructed $DS_{sq}(\alpha)$ is a CDS. We say a network satisfies the *high-transmission* condition if there exists a constant c, $c > 0$, such that every selected node by the algorithm can increase its transmission range from R to at least $(1+c)R$ by increasing its transmission power. In the following we show that, if α is selected carefully, then high-connectivity and high-transmission conditions can guarantee that the constructed $DS_{sq}(\alpha)$ (by, for example, the simple local algorithm explained above) is a CDS. The following lemma is useful in proving these results.

Lemma 1: Let α be a positive real number. Let $u \in \mathcal{C}_i$ and $v \in \mathcal{C}_j$ be two nodes located in two different square cells $\mathcal{C}_i$ and $\mathcal{C}_j$, where the size of each cell is $\alpha R \times \alpha R$. For any pair of nodes $u' \in \mathcal{C}_i$ and $v' \in \mathcal{C}_j$ we have

$$|u'v'| \leq |uv| + 2\sqrt{2}\alpha R.$$

For the set of nodes V and any positive real number r, let $G(V, r)$ denote the graph constructed by connecting two nodes in V if and only if their Euclidean distance is at most r. Using this notation, the high-connectivity condition implies that $G(V, cR)$ is connected for some constant c, $0 < c < 1$. The following theorem shows that, for some range of values of α, $DS_{sq}(\alpha)$ is a CDS if the network satisfies the high-connectivity condition.

Theorem 2: Let α, $0 < \alpha \leq \frac{1-c}{2\sqrt{2}}$, be a real number. Note that $\alpha \leq \frac{\sqrt{2}}{2}$. Let $DS_{sq}(\alpha)$ be a constructed DS by selecting one node in each non-empty set. If $G(V, cR)$ is connected, then $DS_{sq}(\alpha)$ is a CDS whose size is at most $\lceil \frac{2}{\alpha} \rceil^2 |MCDS|$. (For proof see Appendix.)

The next theorem proves that, for some range of α, the constructed $DS_{sq}(\alpha)$ is a CDS whose size is within a constant factor of $|MCDS|$ if the selected nodes increase their transmission range by a constant.

Theorem 3: Let α, $0 < \alpha \leq \frac{\min(2,c)}{2\sqrt{2}}$, and $c > 0$ be real numbers. Note that $\alpha \leq \frac{\sqrt{2}}{2}$. Let $DS_{sq}(\alpha)$ be a constructed DS by selecting one node in each non-empty set. If every node in $DS_{sq}(\alpha)$ increases its transmission range to at least $(1+c)R$, then $DS_{sq}(\alpha)$ will be CDS whose size is at most $\lceil \frac{2}{\alpha} \rceil^2 |MCDS|$.
(For proof see Appendix.)

There are optimization techniques to further reduce the size of a constructed $DS_{sq}(\alpha)$ or to relax the requirement of transmitting at higher power for many selected nodes. For example, suppose that node u_s is the selected node in cell $\mathcal{C}_i$. The node u_s is not required to increase its transmission power if every node v within transmission range of any node $u \in \mathcal{C}_i$ is within the transmission range of either u_s or a selected neighbor of u_s. This condition can be formally expressed as

$$\begin{aligned} &\forall u, v \text{ s.t. } u \in \mathcal{C}_i \wedge |uv| \leq R : \\ &\exists w_s \in DS_{sq}(\alpha) \text{ s.t. } |u_s w_s| \leq R \wedge |w_s v| \leq R. \end{aligned} \tag{1}$$

Both the high-connectivity and high-transmission conditions can be relaxed if we allow selecting more than one yet a constant number of nodes in each non-empty cell. In the following, we describe a simple algorithm that achieves a constant approximation factor in any network $G(V, R)$ without relying on any condition.

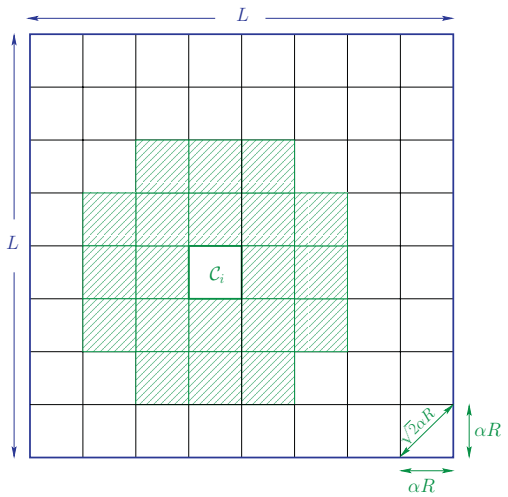

Fig. 2. Network partitioning and possible neighbors of a cell $\mathcal{C}_i$ for the case $\alpha = \frac{\sqrt{2}}{2}$.

Suppose we divide the network into square cells with size $\frac{\sqrt{2}}{2}R \times \frac{\sqrt{2}}{2}R$. All nodes in a cell are 1-hop neighbors of each other as the maximum distance between any two point in the cell is R. Two different cells $\mathcal{C}_i$ and $\mathcal{C}_j$ are called neighbors if and only if

$$\exists u \in \mathcal{C}_i, v \in \mathcal{C}_j \text{ s.t. } |uv| \leq R.$$

In this case, nodes u and v are called connectors of the neighbor cells $\mathcal{C}_i$ and $\mathcal{C}_j$, and u is referred to as a connector

to the cell $\mathcal{C}_j$ through the node v. As shown in Figure 2, each cell is a neighbor of at most 20 other cells if the side of each square cell is set to $\frac{\sqrt{2}}{2}R$. Assume that each node has a list of its 2-hop neighbors together with their positions. Suppose u is a node located in the cell $\mathcal{C}_i$. The node u selects itself as a member of CDS if, based on a criteria, it is the selected connector to a neighboring cell $\mathcal{C}_j$ through a node $v \in \mathcal{C}_j$. The designed criteria must be symmetric in the sense that u selects itself as a connector to the cell $\mathcal{C}_j$ through the node $v \in \mathcal{C}_j$ if and only if v selects itself as a connector to the cell $\mathcal{C}_i$ through the node $u \in \mathcal{C}_j$. As an example of a symmetric criteria, a node $u \in \mathcal{C}_i$ can select itself if and only if it has a neighbor v in a neighboring cell $\mathcal{C}_j$ such that $|uv|$ is minimum among all the possible connectors of the cells $\mathcal{C}_i$ and $\mathcal{C}_j$. Any tie can be broken using, for example, nodes' *id*s. Note that node u has a list of its 2-hop neighbors (as well as their positions) and therefore it can compute the set of all possible connectors between its own cell and any neighboring cell. By Theorem 1, the constructed set is a CDS whose size is at most $20 \times \lceil \frac{2}{\alpha} \rceil^2 = 180$ times of its optimum. Note that, to construct a smaller CDS, many of the selected nodes can be pruned using similar conditions as (1).

An important application of constructing a CDS is to employ it as a backbone for routing. When a CDS is constructed, only the nodes in the set are required to forward packets towards the destination. Therefore, each path between the source node s and destination node d can be represented as

$$s, w_1, w_2, \ldots, w_k, d,$$

where w_i are in the CDS. Let $l(s,d)$ and $l_{CDS}(s,d)$ denote the length of shortest paths between s and d in the original graph and the graph induced by CDS $\cup \{s,d\}$, respectively. In general, the ratio $\frac{l_{CDS}(s,d)}{l(s,d)}$ can be very large, in the worst case. For example, consider a network with $N > 3$ nodes located on a circle with radius $\frac{R}{2\sin(\frac{\pi}{N})}$ such that the distance between any two neighbors is R. In the corresponding cycle graph, every MCDS is a simple path of length $N-3$. Let s and d be the nodes at the two ends of the path. Nodes s and d are only 3 hops away from each other. However, $l_{MCDS}(s,d)$, the length of the shortest path between s and d in the graph induced by the nodes in $MCDS$, is $N-3$. Thus,

$$\frac{l_{MCDS}(s,d)}{l(s,d)} = \frac{N}{3} - 1.$$

Consequently, even a MCDS cannot provide a good approximation of a shortest path in the original graph. Theorem 4, on the other hand, shows that the length of a shortest path in $CDS_{sq}(\alpha)$ constructed based on the high-transmission condition is at most one more than that of the original shortest path.

Theorem 4: For the CDS constructed based on the high-transmission condition we have

$$l_{CDS}(s,d) \leq l(s,d) + 1.$$

Employing a symmetric criteria to construct the CDS or using Condition (1) will change the shortest path approximation to

$$l_{CDS}(s,d) \leq 2 \times l(s,d) + 1.$$

(For proof see Appendix.)

IV. Broadcasting Using The Dynamic Approach

Using the dynamic approach, the status (forwarding/non-forwarding) of each node is determined "on-the-fly" as the broadcasting message propagates in the network. In particular, in neighbor-designating broadcast algorithms, each forwarding node selects a subset of its neighbors to forward the packet and in self-pruning algorithms each node determines its own status based on a self-pruning condition after receiving the first or several copies of the message. It was recently proved that self-pruning broadcast algorithms (hence broadcast algorithms based on the dynamic approach) are able to guarantee both full delivery and a constant approximation factor to the optimum solution (MCDS) [14]. However, the proposed algorithm in [14] uses position information in order to design a strong self-pruning condition. In the previous section, we observed that position information can simplify the problem of reducing the total number of broadcasting nodes. Moreover, having position information may not be practical in some applications. Therefore, it is interesting to know if both full delivery and a constant approximation factor can be achieved when position information is not available. In this section, we design a hybrid (i.e., both neighbor-designating and self-pruning) broadcast algorithm and show that the algorithm can achieve both full delivery and constant approximation only using connectivity information.

A. The Proposed Local Broadcast Algorithm

Suppose each node has a list of its 2-hop neighbors (i.e., nodes that are at most 2 hops away). This can be achieved in two rounds of information exchange. In the first round, each node broadcasts its *id* to its 1-hop neighbors (simply called neighbors). Thus, at the end of the first round, each node has a list of its neighbors. In the second round, each node transmits its *id* together with the list of its neighbors.

The proposed broadcast algorithm is a hybrid algorithm, hence every node that broadcasts the message may select some of its neighbors to forward the message. In our proposed broadcast algorithm, every broadcasting node selects at most one of its neighbors. A node has to broadcast the message if it is selected to forward. Other nodes that are not selected have to decide whether or not to broadcast on their own. This decision is made based on a *self-pruning condition* called the *coverage condition*.

To evaluate the coverage condition, every node u maintains a list $List_u^{cov}(m)$ for every unique message m. Upon receiving a message m for the first time, $List_u^{cov}(m)$ is created and filled with the *id*s of all neighbors of u and then updated as follows. Suppose u receives m from its neighbor v and assume that v selects $w \neq u$ to forward the message. Note that w may not be a neighbor of u. However, since w is a neighbor of v, it is at most 2 hops away from u. Having *id*'s of v and w (included in the message), node u updates $List_u^{cov}(m)$ by removing all nodes in $List_u^{cov}(m)$ that are a neighbor of either v or w. This update can be done because u has a list of its 2-hop neighbors. Since w will eventually broadcast the message, by updating the list, u removes those neighbors that have received the message or will receive it, eventually. Every time u receives a copy

of message m it updates $List_u^{cov}(m)$ as explained earlier. If $w = u$ (i.e., u is selected by v to forward the message), node u updates $List_u^{cov}(m)$ by removing only neighbors of v from the list. Note that in this case, u must broadcast the message. However, u has to update $List_u^{cov}(m)$ as it needs to select one of its neighbors from the updated list (if it is not empty) to forward the message.

*Definition 1 (**coverage condition**):* We say the *coverage condition* for node u is satisfied at time t if $List_u^{cov}(m) = \emptyset$ at time t.

Algorithm 1 shows our proposed hybrid broadcast algorithm. When a node u receives a message m, it creates a list $List_u^{cov}(m)$ if it is not created yet and updates the list as explained earlier. Then, based on whether u was selected to forward or whether the coverage condition is satisfied, u may schedule a broadcast by placing a copy of m in its MAC layer queue. There are at least two sources of delay in the MAC layer. First, a message may not be at the head of the queue so it has to wait for other packets to be transmitted. Second, in contention based channel access mechanisms such as CSMA/CA, to avoid collision, a packet at the head of the queue has to wait for a random amount of time before getting transmitted. In this paper, we assume that a packet can be removed from the MAC layer queue if it is no longer required to be transmitted. Therefore, the broadcast algorithm has access to two functions to manipulate the MAC layer queue. The first function is the scheduling/placing function, which is used to place a message in the MAC layer queue. We assume that the scheduling function handles duplicate packets, i.e., it does not place the packet in the queue if a copy of it is already in the queue. The second function is called to remove a packet form the queue (it does not do anything if the packet is not in the queue). Note that to remove a packet from the queue, the algorithm needs to have access to the MAC layer queue. This requires a cross-layer design. In the absence of any cross-layer design, the broadcast algorithm can use a timer at the network layer. We refer the reader to [17] (Section IV) for a discussion on how to use a timer to avoid this cross-layer problem.

The proposed algorithm obeys the following:

1) u discards a received message m if it has broadcast m before.
2) If u is selected to forward the message, it schedules a broadcast (regardless of the coverage condition) and never removes the messages from the queue in future. However, u may change or remove the selected node's *id* from the scheduled message every time it receives a new copy of the message and updates $List_u^{cov}(m)$.
3) Suppose u has not been selected to forward the message by time t and the $List_u^{cov}(m)$ becomes empty at time t after an update. Then at time t, it removes the message from the MAC layer queue (if the message has been scheduled before and is still in the queue).
4) If $List_u^{cov}(m) \neq \emptyset$ then u selects a node from $List_u^{cov}(m) \neq \emptyset$ to forward the message and adds the *id* of the selected node in the message. The selection can be done randomly or based on a criteria. For example, u can select the node with the minimum *id* or the one with maximum battery life-time.
5) If u has been selected to forward and $List_u^{cov}(m) = \emptyset$ it does not select any node to forward the message. This is the only case where a broadcasting node does not select any of its neighbors to forward the message.

Algorithm 1 The proposed hybrid algorithm executed by u

```
Extract id's of the broadcasting node and the selected node from the received message m
if u has broadcast the message m before then
    Discard the message
    Return
end if
if u receives m for the first time then
    Create and fill the list List_u^cov(m)
end if
Update the list List_u^cov(m)
Remove the information added to the message by the previous broadcasting node
if List_u^cov(m) ≠ ∅ then
    Select an id from List_u^cov(m) and add it to the message
    Schedule the message {(*only update the selected id if m is already in the queue*)}
else {(*List_u^cov(m) = ∅ in this case*)}
    if u was selected then
        Schedule the message {(*only remove the id of the selected neighbor if m is already in the queue*)}
    else
        Remove the message form the queue if u has not been selected by any node before
    end if
end if
```

B. Analysis of the Proposed Broadcast Algorithm

In this section, we prove that the proposed broadcast algorithm guarantees full delivery as well as a constant approximation to the optimum solution irrespective of the forwarding node selection criteria and the random delay in the MAC layer (hence, the random sequence of the broadcasting nodes). In order to prove these properties, we assume that nodes are static during the broadcast, the network is connected and there is no loss at the MAC/PHY layer. Note that even flooding cannot guarantee full delivery without these assumptions.

Theorem 5: Algorithm 1 guarantees full delivery.

Proof: Every node broadcasts a message at most once. Therefore, the broadcast process eventually terminates. By contradiction, assume that node d has not received the message by the broadcast termination. Since the network is connected, there is a path from the source node s (the node that initiates the broadcast) to node d. Clearly, we can find two nodes u and v on this path such that u and v are neighbors, u has received the message and v has not received it. The node u has not broadcast the message since v has not received it. Therefore, u has not been selected to broadcast; thus the coverage condition must have been satisfied for u. As the result, v must have a neighbor w, which has broadcast the message or was selected

to broadcast. Note that all the selected nodes will eventually broadcast the message. This is a contradiction as, based on the assumption, v cannot have a broadcasting neighbor. ■

Lemma 2: Using Algorithm 1, the number of broadcasting nodes inside any disk $D_{O,\frac{R}{2}}$ centered at an arbitrary point O and with a radius $\frac{R}{2}$ is bounded by 33.

Proof: All nodes inside $D_{O,\frac{R}{2}}$ are neighbors of each other, thus they receive each others messages. The broadcasting nodes can be divided into two types based on whether or not the coverage condition was satisfied for them just before they broadcast the message. Recall that the coverage condition may be satisfied for a broadcasting node if the node has been selected to forward the message. It is because a selected node has to broadcast the message irrespective of the coverage condition. Consider two disks centered at O with radii $\frac{R}{2}$ and $\frac{3R}{2}$, respectively. Suppose k is the minimum number such that for every set of k nodes $w_i \in D_{O,\frac{3R}{2}}$, $1 \leq i \leq k$, we have

$$\exists i, j \neq i : \quad |w_i w_j| \leq R.$$

Following, we find an upper bound on k. By the minimality of k, there must exist $k-1$ nodes $w_i \in D_{O,\frac{3R}{2}}$, $1 \leq i \leq k-1$, such that

$$\forall i, j \neq i : \quad |w_i w_j| > R. \tag{2}$$

Consider $k-1$ disks $D_1, \ldots, D_{k-1}$ with radius $\frac{R}{2}$ centered at w_i, $1 \leq i \leq k-1$, respectively. By (2), $D_1, \ldots, D_{k-1}$ are non-overlapping disks. Also, every disk D_i, $1 \leq i \leq k-1$, resides in $D_{O,2R}$, that is the disk centered at O with radius $2R$. It is because, the center of every Disk D_i, $1 \leq i \leq k-1$, is inside $D_{O,\frac{3R}{2}}$. Thus, by an area argument, we get

$$(k-1)\left(\pi(\frac{R}{2})^2\right) \leq \pi(2R)^2.$$

Hence, $k \leq 17$.

We first prove that the number of broadcasting nodes inside $\mathcal{D}_{O,\frac{R}{2}}$ for which the coverage condition is not satisfied is at most $k-1$. we then prove the same upper bound for the number of broadcasting nodes inside $\mathcal{D}_{O,\frac{R}{2}}$ for which the coverage condition is satisfied. Consequently, the total number of broadcasting nodes inside $\mathcal{D}_{O,\frac{R}{2}}$ is bounded by $2k-2 \leq 32$. By contradiction, suppose that there are more than $k-1$ broadcasting nodes inside $D_{O,\frac{R}{2}}$ for which the coverage condition is not satisfied. Let $u_1, \ldots u_k$ be the first k broadcasting nodes ordered chronologically based on their broadcast time, and $a_1, \ldots a_k$ the corresponding selected neighbor. Thus, for every i, $1 \leq i \leq k$, we have $a_i \in List^{cov}_{u_i}(m)$, where $List^{cov}_{u_i}(m)$ is the list of node u_i at the time it broadcasts the message. Since $u_1, \ldots u_k$ are all in $D_{O,\frac{R}{2}}$ and for every i, $1 \leq i \leq k$, $|u_i a_i| \leq R$, we get:

$$\forall i, 1 \leq i \leq k : \quad a_i \in D_{O,\frac{3R}{2}}.$$

Thus, by the definition of k, there are two nodes a_i, a_j, $i < j$ such that $|a_i a_j| \leq R$. The node u_i is broadcast before u_j and is a neighbor of it. Therefore, u_j is aware of u_i's selected neighbor a_i and removes a_j from $List^{cov}_{u_j}(m)$ as soon as it receives the message from u_i. This is a contradiction because $a_j \in List^{cov}_{u_j}(m)$ at the time u_j is broadcast.

It remains to prove that the number of broadcasting nodes inside $D_{O,\frac{R}{2}}$ for which the coverage condition is satisfied is at most $k-1$. By contradiction, suppose that there are at least k broadcasting nodes inside $D_{O,\frac{R}{2}}$ for which the coverage condition is satisfied. Let $v_1, \ldots v_k \in D_{O,\frac{R}{2}}$ be the first k broadcasting nodes, ordered chronologically based on their broadcast time. Note that a broadcasting node must have been selected (by another node) to forward the message if its coverage condition is satisfied. Let $b_1, b_2, \ldots b_k$ be the nodes that selected $v_1, \ldots v_k$ to forward the message. Clearly, for every i, $1 \leq i \leq k$, we have $b_i \in D_{O,\frac{3R}{2}}$. Also, for every i, $1 \leq i \leq k$ and every j, $1 \leq j \leq k$ and $j \neq i$, we get $b_i \neq b_j$, because each node can select at most one other node to forward. By the definition of k, there must exist two nodes b_i and b_j, $i < j$ such that $|b_i b_j| \leq R$. This is a contradiction because b_i and b_j are neighbors and b_j receives the b_i broadcast message, thus $v_j \notin List^{cov}_{v_j}(m)$ as v_i and v_j are neighbors. ■

Corollary 1: Let u be any node in the network. Using Algorithm 1, the number of broadcasting nodes within the transmission range of u is at most 224.

Proof: All the nodes within the transmission range of u (including u) are inside a disk with radius R. A disk with radius R can be covered with at most 7 disks with radius $\frac{R}{2}$ [18]. Thus, by Lemma 2, the number of broadcasting nodes within the transmission range of u is at most $7 * 32 = 224$. ■

Theorem 6: Algorithm 1 has a constant approximation factor to the optimal solution (MCDS). Moreover, the approximation factor is at most 224.

Proof: Let S_{MCDS} be a MCDS and S_{Alg} be the set of broadcasting nodes using Algorithm 1. Let u be any node in S_{MCDS}. By Corollary 1, the number of broadcasting nodes within the transmission range of u is at most 224. Note that every broadcasting node is within the transmission range of at least one node in S_{MCDS}, because S_{MCDS} is a dominating set. Therefore,

$$|S_{Alg}| \leq 224 \times |S_{MCDS}|.$$

■

C. The Strong Coverage Condition

As proven, the proposed broadcast algorithm guarantees that the total number of transmissions is always within a constant factor of the minimum number of required ones. However, the number of transmissions may be further reduced by slightly modifying the broadcast algorithm. As explained earlier, in the proposed algorithm, a selected node has to broadcast the message even if its coverage condition is satisfied. Nevertheless, in some cases, a selected node can avoid broadcasting. For example, a selected node u can abort transmission (by removing the message from the queue) at time t if by time t and based on its collected information, all its neighbors have received the message. This idea can be generalized as follows.

Suppose, for each unique message m, every node u maintains and updates an extra list $List^{str}_u(m)$. Similar to $List^{cov}_u(m)$, $List^{str}_u(m)$ is created and filled with the *id*s of u's neighbors upon the first reception of message m. Also,

every time u receives m, it updates $List_u^{str}(m)$ as follows. Let v be the broadcasting node and $w \neq u$ the selected node by v. Node u first removes the nodes in $List_u^{str}(m)$ that are neighbors of v. If the priority of w (e.g., its *id*) is higher than u, it also removes the nodes in $List_u^{str}(m)$ that are neighbors of w. To further reduce the number of redundant transmissions, a selected node can abort broadcasting m under the following *strong coverage condition*.

*Definition 2 (**strong coverage condition**):* We say the *strong coverage condition* is satisfied for node u at time t if $List_u^{str}(m) = \emptyset$ at time t.

Note that the strong coverage condition is only used by selected nodes to check whether they need to broadcast. Other nodes make a decision based on the previously-defined coverage condition (a weaker condition). The following theorem states that the full delivery is guaranteed if the selected nodes abort transmissions when the strong coverage condition is satisfied. Using a similar approach to that used in the proof of Lemma 2, it can be proven that this extension of the algorithm also achieves a constant approximation factor.

Theorem 7: Suppose Alg-str is a modified version of Algorithm 1 in which each node maintains two lists $List_u^{cov}(m)$ and $List_u^{str}(m)$ and selected nodes can avoid broadcasting under the strong coverage condition. Alg-str guarantees full delivery. (For proof see Appendix.)

D. Extending the Network Model

The results presented in the paper can be extended to the case where the nodes are distributed in three-dimensional space. In other words, when the nodes are distributed in three-dimensions it can be shown that local broadcast algorithms based on the static approach can provide a constant approximation if nodes have their position information. By replacing circles with balls, it can be similarly shown that Algorithm 1 can provide both full delivery and a constant approximation to the optimum solution.

Algorithm 1 (the proposed algorithm based on the dynamic approach) can be extended to the case where the nodes have different transmission ranges. In this case, it can be proved that the algorithm guarantees a constant approximation factor if the ratio $\frac{R_{Max}}{R_{Min}}$ is constant, where R_{Max} is the maximum transmission range and R_{Min} is the minimum transmission range of the nodes in the network and two nodes have a link iff both are in transmission range of each other. Similar to the proof of Theorem 6, this can be proved by showing that the number of broadcasting nodes inside any disk $D_{O,\frac{R_{Min}}{2}}$ is constant. Also, we can use the quasi unit disk graph to model the network [19]. In this model, there is a link between two nodes if their Euclidean distance is less than γR, $0 < \gamma \leq 1$, and there is no link if the Euclidean distance is more than R. This model is closer to reality than the unit disk graph model. Using the quasi unit disk graphs model we can show that Algorithm 1 guarantees a constant approximation factor if γ is constant. Similarly, the proof is by showing that the number of broadcasting nodes in any disk $D_{O,\frac{\gamma R}{2}}$ is constant.

V. Experimental Results

One of the major contributions of this work is the design of a local broadcast algorithm based on the dynamic approach (Algorithm 1) that can achieve both full delivery and a constant approximation factor to the optimum solution without using position information. To confirm the analytical results, we implemented Algorithm 1, Liu's algorithm (a neighbor-designating algorithm) [9], edge forwarding algorithm (a self-pruning algorithm) [20] and flooding in the network simulator ns-2 and evaluated the ratio of broadcasting nodes (i.e., number of broadcasting nodes/total number of nodes) and end-to-end delay for each algorithm. Table I summarizes the parameters used in the ns-2 simulator. We also implemented the Wan-Alzoubi-Frieder algorithm [21] in C++ and used it as an approximation of the minimum number of broadcasting nodes required. Note that the Wan-Alzoubi-Frieder algorithm (referred to as ratio-6 approximation algorithm) is not a local algorithm and is only used as a benchmark as it has an approximation factor of at most 6 [22].

Both Liu's broadcast algorithm and the edge-forwarding algorithm use position information of nodes to reduce the total number of transmissions. Liu et al. showed that the number of broadcasting nodes using their algorithm is significantly lower that that of previous notable broadcast algorithms [20], [23]. In Liu's algorithm, each broadcasting node selects a smallest group of its 1-hop neighbors such that any potential 2-hop neighbor is within transmission range of at least one of the selected nodes. It then piggybacks the list of the selected nodes in the message before broadcasting it. Upon receiving a message for the first time, a node schedules a broadcast if and only if it is selected to forward; otherwise, it never broadcasts the message. Using the edge-forwarding algorithm, each node divides its transmission coverage into six equal size sectors. It then decides whether or not to broadcast based on the sector in which the broadcasting node is located, and the position of its 1-hop neighbors that have received the message.

To compute the number of broadcasting nodes, we uniformly distributed the nodes in a square of size $1000 \times 1000m^2$. We allowed only one network-wide broadcast at each simulation run, selected the next forwarding node randomly, and used the strong coverage condition in Algorithm 1 to further reduce the total number of transmissions.

Figures 3 and 4 show the average ratio of broadcasting nodes for over 500 runs for each given value of the input (i.e., the total number of nodes or the transmission range). To get the results shown in Figure 3, we set the transmission range to $250m$ and varied the total number of nodes from 25 to 1000. In Figure 4, the number of nodes was fixed to 1000 and the transmission range was varied from $50m$ to $300m$. The transmission range and the total number of nodes were selected from a large interval so that the simulation covers very sparse and very dense networks as well as the networks with large diameters. Both Figures 3 and 4 show that the ratio of broadcasting nodes using Algorithm 1 is significantly lower than that of Liu's algorithm and the edge-forwarding algorithm and is very close to the estimated minimum number of required transmissions (computed using the Wan-Alzoubi-

Frieder algorithm).

We computed another metric called the algorithm delay, which we define as the time between the first and the last transmission in a single network-wide broadcast. Figure 5, shows the average delay of our proposed algorithm, Liu's algorithm and the flooding algorithm. To get these results, we fixed the transmission range to $250m$ and varied the number of nodes from 25 to 1000. Based on the simulation results, the average delay of our broadcast algorithm is 86% to 54% of that of Liu's algorithm and 87% to 71% of that of the flooding algorithm.

Since Algorithm 1 requires nodes to collect their 2-hop neighbor information, it is mainly suitable for networks in which nodes have low mobility. The network-wide broadcast may take from a fraction of second to a few seconds depending on the MAC layer settings, the size of the network and the amount of contention. Therefore, if the average distance that nodes move during the network-wide broadcast is a small fraction of R, then we expect Algorithm 1 to achieve similar results as in the static networks. Otherwise, in networks with high mobility rates, Algorithm 1 cannot use the benefit of selecting nodes to forward the message. It is because, a node does not know at what time a selected node will transmit the message if the selected node is not within its transmission range. Therefore, in high mobility settings, a node may not know whether any of its neighbors is covered by the transmission of a selected node as it does not know the list of 1-hop neighbors of the selected node at the time the selected node transmits the message. If nodes do not select other nodes to forward, then Algorithm 1 becomes similar to the proposed algorithm in [12]. The proposed algorithm in [12] can provide full delivery in static networks, but it cannot guarantee a constant approximation factor. To evaluate the impact of low mobility on the number of transmissions required by Algorithm 1, we performed a simulation experiment. As before, we distributed the nodes in a square of size $1000 \times 1000m^2$ and allowed only one network-wide broadcast at each simulation run. We set the transmission range to $250m$ and the number of nodes to 50. At the beginning of each run, nodes start exchanging Hello messages. After a period of time, a random node initiates a broadcast. During the whole simulation run, nodes move according to the random waypoint mobility model [24]. In our previous simulations experiments, nodes do not exchange Hello messages after the broadcast initiation. Therefore, they do not take into consideration the overhead of the 2-hop neighbor discovery messages. However, in this setting, nodes exchange Hello messages during the whole simulation run in order to keep the list of neighbors up-to-date. Figure 6 shows the effect of mobility on the number of transmissions. The x-axis is the maximum speed set in the random waypoint mobility model and the y-axis is the average ratio of broadcasting nodes for at least 500 runs. As shown in Figure 6, the number of transmissions slightly decreases as mobility increases. Figure 7, shows the average delay of Algorithm 1 decreases as mobility increases.

We also evaluated the performance of Algorithm 1 in denser networks. Figure 8 shows the average number of transmissions when the total number of nodes varies from 50 to 1000. To obtain the results shown in Figure 8, we set the maximum speed to $10m/s$ and fixed the hello message transmission rate to one per second. The simulation results show that for low mobility rates, the performance (in terms of the number of transmissions) of both Algorithm 1 and Liu's algorithm are very close to the case where the network is static. Also, for both algorithms, the average number of nodes that do not receive a message in the aforementioned settings is less than 2.

As mentioned in Section IV, Algorithm 1 can use a back-off timer in the network layer in the absence of a cross layer design. Each time a node receives a message, it sets the timer with a random number and decides upon timer expiration whether to retransmit. Larger back-off delays decrease the probability of collisions and reduce the number of transmissions as nodes can hear more transmissions, hence make better decisions on whether or not to retransmit. However, as shown in Figure 9, algorithm's end-to-end delay linearly increases with the maximum duration of back-off timer. To obtain the results shown in Figure 9, we set the transmission range to $250m$ and considered the network static. The x-axis represents the maximum back-off delay in seconds and the y-axis represents the delay of Algorithm 1 in seconds.

Finally, figures 10 and 11, respectively, show an instance of using our proposed algorithm and Liu's algorithm for the same network where the total number of nodes is 400 and the transmission range is set to $250m$ (broadcasting nodes are shown by stars). As proven in [14], in neighbor-designating algorithms based on 1-hop neighbor information (and in Liu's algorithm, in particular) most of the nodes around the boundary of the network broadcast the message. This undesirable property of Liu's algorithm can be observed in Figure 11. Note that these snapshots of Algorithm 1 and Liu's algorithm are not necessarily representative of the overall performance of the two algorithms.

Parameter	**Value**
Simulator	ns-2 (version 2.33)
MAC Layer	IEEE 802.11
Propagation Model	two-ray ground
Packet Size	256 bytes
Bandwidth	2 Mb/sec
Size of Square Area	1000×1000m^2
Average Forwarding Delay	1 ms
Transmission Range	50–300m
Number of Nodes	25–1000

TABLE I

SIMULATION PARAMETERS

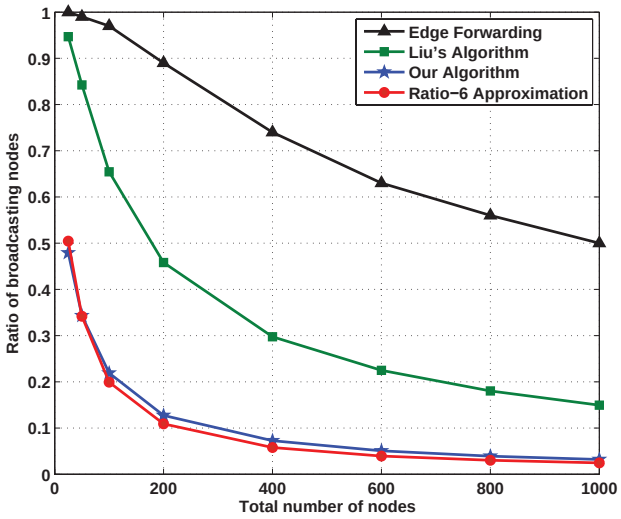

Fig. 3. Ratio of broadcasting nodes vs. total number of nodes.

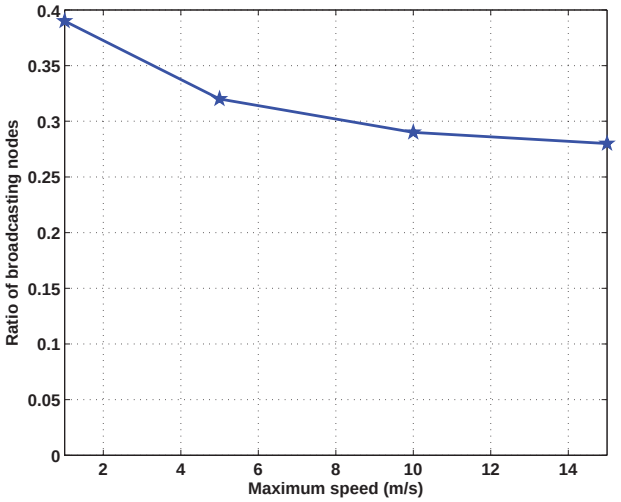

Fig. 6. Ratio of broadcasting nodes vs. maximum speed.

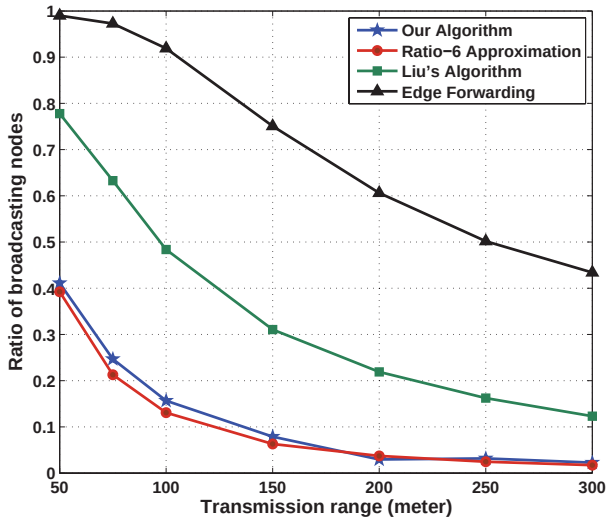

Fig. 4. Ratio of broadcasting nodes vs. transmission range.

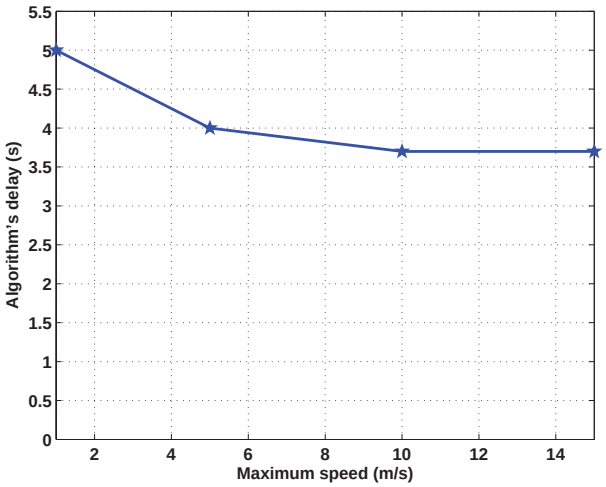

Fig. 7. Ratio of broadcasting nodes vs. maximum speed.

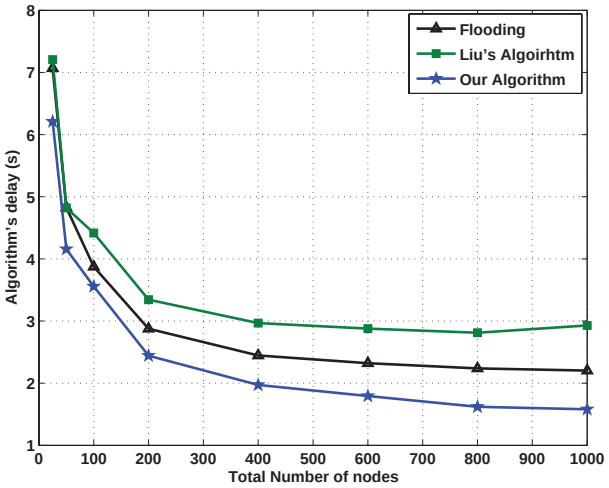

Fig. 5. Average delay vs. total number of nodes.

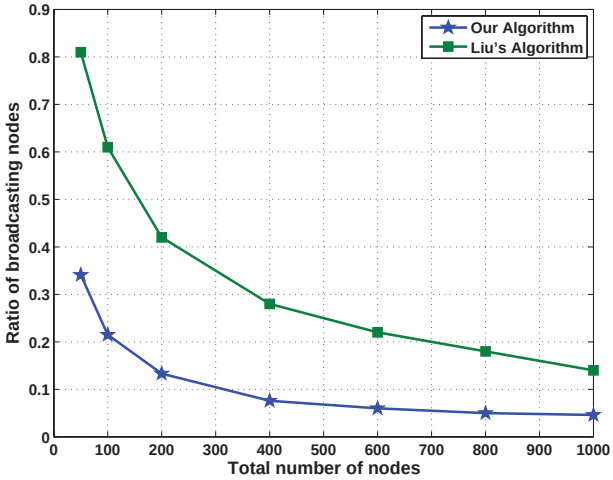

Fig. 8. Ratio of broadcasting nodes vs. total number of nodes; max. speed=10m/s

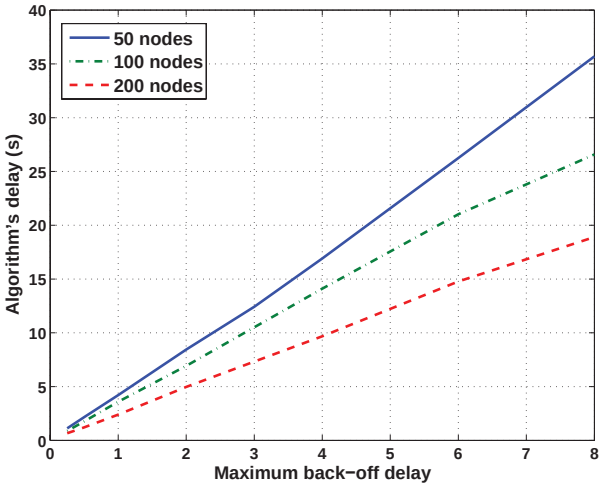

Fig. 9. Algorithm's delay vs. maximum back-off delay

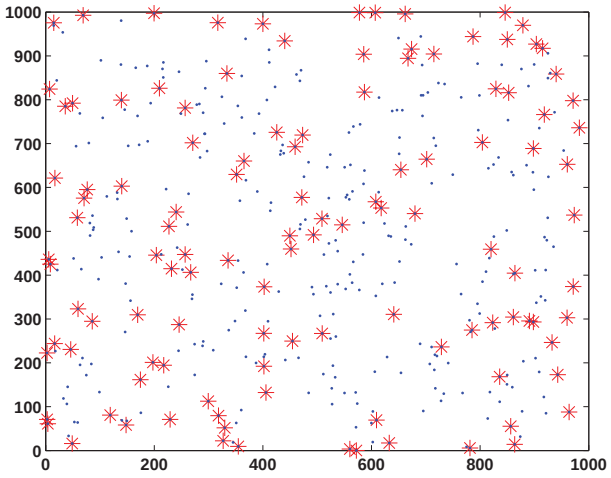

Fig. 11. An instance of using Liu's broadcast algorithm; Transmission range=250m, # nodes=400.

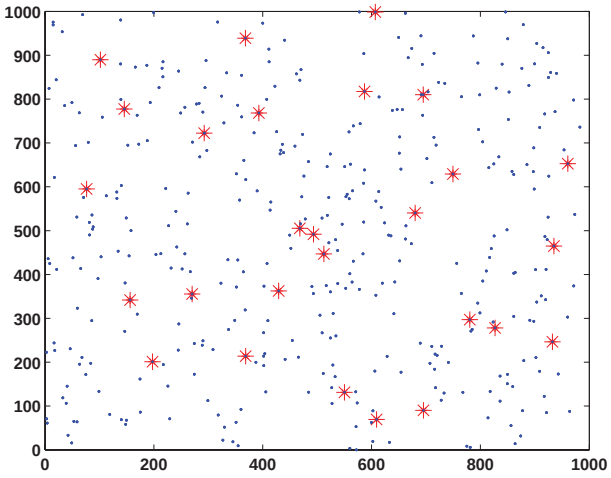

Fig. 10. An instance of using our broadcast algorithm; Transmission range=250m, # nodes=400.

VI. Conclusions

In this paper, we investigated capabilities of local broadcast algorithms in reducing the total number of transmissions that are required to achieve full delivery. As proven, local broadcast algorithms based on the static approach cannot guarantee a small sized CDS if the position information is not available. It was shown that having relative position information can greatly simplify the problem of reducing the total number of selected nodes using the static approach. In fact, we showed that a constant approximation factor is achievable using position information. Using the dynamic approach, it was recently shown that a constant approximation is possible using (approximate) position information [14]. In this paper, we showed that local broadcast algorithms based on the dynamic approach do not require position information to guarantee a constant approximation factor. The results presented in the paper can be extended to the case where nodes are distributed in three-dimensional space. Also, the proposed algorithm based on the dynamic approach can be extended to the case where nodes have different transmission ranges or when the network is modeled using the quasi unit disk graph model.

References

[1] M. Garey and D. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman & Co., 1990.
[2] B. Clark, C. Colbourn, and D. Johnson, "Unit disk graphs," *Discrete Mathematics*, vol. 86, pp. 165–177, 1990.
[3] J. Wu and F. Dai, "Broadcasting in ad hoc networks based on self-pruning," *In Proc. IEEE INFOCOM*, pp. 2240–2250, 2003.
[4] J. Wu and W. Lou, "Forward-node-set-based broadcast in clustered mobile ad hoc networks," *Wireless Communications and Mobile Computing*, vol. 3, no. 2, pp. 155–173, 2003.
[5] J. Wu and F. Dai, "A generic distributed broadcast scheme in ad hoc wireless networks," *IEEE Transactions on Computers*, vol. 53, no. 10, pp. 1343–1354, 2004.
[6] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, "The broadcast storm problem in a mobile ad hoc network," *In Proc. ACM International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 151–162, 1999.
[7] Z. Haas, J. Halpern, and L. Li, "Gossip-based ad hoc routing," *In Proc. IEEE INFOCOM*, pp. 1707–1716, 2002.
[8] D. Y. Sasson and A. Schiper, "Probabilistic broadcast for flooding in wireless mobile ad hoc networks," *In Proc. IEEE Wireless Communications and Networking Conference*, pp. 1124–1130, 2003.
[9] H. Liu, P. Wan, X. Jia, X. Liu, and F. Yao, "Efficient flooding scheme based on 1-hop information in mobile ad hoc networks," *In Proc. IEEE INFOCOM*, 2006.
[10] J. Wu, W. Lou, and F. Dai, "Extended multipoint relays to determine connected dominating sets in manets," *IEEE Trans. on Computers*, vol. 55, no. 3, pp. 334–347, 2006.
[11] M. Khabbazian and V. K. Bhargava, "Efficient broadcasting in mobile ad hoc networks," *IEEE Transactions on Mobile Computing: accepted for publication*, 2008.
[12] W. Peng and X. Lu, "On the reduction of broadcast redundancy in mobile ad hoc networks," *In Proc. ACM Interational Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pp. 129–130, 2000.
[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*, vol. 13, pp. 14–25, 2002.
[14] M. Khabbazian and V. K. Bhargava, "Localized broadcasting with guaranteed delivery and bounded transmission redundancy," *IEEE Transactions on Computers*, vol. 57, no. 8, pp. 1072–1086, 2008.

[15] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," *In Proc. ACM International Conference on Mobile Computing and Networking (MOBICOM)*, 2001.
[16] Y. Chen and J. L. Welch, "Location-based broadcasting for dense mobile ad hoc networks," *In Proc. ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pp. 63–70, 2005.
[17] A. Keshavarz-Haddad, V. Ribeiro, and R. Riedi, "DRB and DCCB: Efficient and robust dynamic broadcast for ad hoc and sensor networks," *In Proc. IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pp. 253 – 262, 2007.
[18] C. T. Zahn, "Black box maximization of circular coverage," *Journal of Research of the National Bureau of Standards B.*, vol. 66, pp. 181–216, 1962.
[19] L. Barriére, P. Fraigniaud, and L. Narayanan, "Robust position-based routing in wireless ad hoc networks with unstable transmission ranges," pp. 19–27, 2001.
[20] Y. Cai, K. Hua, and A. Phillips, "Leveraging 1-hop neighborhood knowledge for efficient flooding in wireless ad hoc networks," *In Proc. IEEE International Performance, Computing, and Communications Conference (IPCCC)*, pp. 347–354, 2005.
[21] P. Wan, K. Alzoubi, and O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," *In Proc. IEEE INFOCOM*, vol. 3, pp. 1597–1604, 2002.
[22] A. Vahdatpour, F. Dabiri, M. Moazeni, and M. Sarrafzadeh, "Theoretical bound and practical analysis of connected dominating set in ad hoc and sensor networks," *DISC*, pp. 481–495, 2008.
[23] J. Wu and H. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless networks," *In Proc. International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DiaLM)*, pp. 7–14, 1999.
[24] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *WIRELESS COMMUNICATIONS and MOBILE COMPUTING (WCMC)*, vol. 2, pp. 483–502, 2002.

PLACE PHOTO HERE

Majid Khabbazian received his undergraduate degree in computer engineering from Sharif University of Technology, Iran, and his Master's and Ph.D. degrees both in electrical and computer engineering from the University of Victoria and the University of British Columbia, Canada, respectively. From 2009 to 2010, he was a research fellow at the Computer Science And Artificial Intelligence Lab, MIT. Currently, Dr. Khabbazian is an assistant professor at the Department of Applied Computer Science, University of Winnipeg, Canada. His research interests include wireless networks, distributed algorithms, applied cryptography and network security.

PLACE PHOTO HERE

Ian F. Blake received his undergraduate degree from Queens University, Kingston, ON, Canada, and his Ph.D. degree from Princeton University, Princeton, NJ. From 1967 to 1969, he was a Research Associate with the Jet Propulsion Laboratory, Pasadena, CA. From 1969 to 1996, he was with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, where he was Chairman from 1978 to 1984 and Director of the Institute of Computer Research from 1990 to 1994. From 1996 to 1999, he was with the Hewlett-Packard Labs, Palo Alto, CA. He is currently with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, Canada. His research interests are in the areas of cryptography, algebraic coding theory, digital communications, and spread-spectrum systems. Dr. Blake is a Fellow of the Institute for Combinatorics and its Applications, the IEEE and the Canadian Academy of Engineering.

PLACE PHOTO HERE

Vijay K. Bhargava received his undergraduate, Master's, and Ph.D. degrees from Queen's University, Kingston, ON, Canada in 1970, 1972 and 1974, respectively. Currently, he is a Professor at the University of British Columbia, Vancouver,Canada. Previously he was with the University of Victoria (1984-2003) and with Concordia University in Montreal (1976-1984). He is a co-author of the book Digital Communications by Satellite (New York: Wiley, 1981), co-editor of Reed-Solomon Codes and Their Applications (New York: IEEE, 1994) and co-editor of Communications, Information and Network Security (Boston: Kluwer, 2003). His research interests are in wireless communications. Dr. Bhargava is a Fellow of the Engineering Institute of Canada (EIC), the IEEE, the Canadian Academy of Engineering and the Royal Society of Canada. He has served on the Board of the IEEE Information Theory Society and the IEEE Communications Society. He is a Past President of the IEEE Information Theory Society and a past Editor-in-Chief of the IEEE Transactions on Wireless Communications.