A Generic Distributed Broadcast Scheme in Ad Hoc Wireless Networks

Jie Wu, *Senior Member*, *IEEE*, and Fei Dai, *Student Member*, *IEEE*

Abstract—We propose a generic framework for distributed broadcasting in ad hoc wireless networks. The approach is based on selecting a small subset of nodes to form a forward node set to carry out a broadcast process. The status of each node, forward or nonforward, is determined either by the node itself or by other nodes. Node status can be determined at different snapshots of network state along time (called views) without causing problems in broadcast coverage. Therefore, the forward node set can be constructed and maintained through either a proactive process (i.e., "up-to-date") before the broadcast process or a reactive process (i.e., "on-the-fly") during the broadcast process. A sufficient condition, called *coverage condition*, is given for a node to take the nonforward status. Such a condition can be easily checked locally around the node. Several existing broadcast algorithms can be viewed as special cases of the generic framework with k-hop neighborhood information. A comprehensive comparison among existing algorithms is conducted. Simulation results show that new algorithms, which are more efficient than existing ones, can be derived from the generic framework.

Index Terms—Ad hoc wireless networks, broadcasting, distributed algorithms, pruning.

1 Introduction

Ad hoc wireless networks (or simply ad hoc networks) are dynamic in nature. Due to this dynamic nature, global information/infrastructure, such as link state and routing table, which are obtained through global information exchanges are no longer suitable to support routing in ad hoc networks. Broadcasting is a special routing process of transmitting a packet so that each node in a network receives a copy of this packet. *Flooding* is a simple approach to broadcasting with no use of global information/infrastructure; in flooding, a broadcast packet is forwarded by every node in the network exactly once. Simple flooding ensures the coverage; the broadcast packet is guaranteed to be received by every node in the network providing there is no packet loss caused by collision in the MAC layer and there is no high speed movement of nodes during the broadcast process. Fig. 1a shows a network with six nodes. When node v broadcasts a packet as shown in Fig. 1b, all neighboring nodes, u, w, x, and y, receive the packet due to the broadcast nature of wireless communication media. All neighbors will then forward the packet to each other. Apparently, the two transmissions from nodes u and x are unnecessary. Redundant transmissions may cause the *broadcast storm problem* [17] in which redundant packets cause contention and collision.

Both deterministic and probabilistic approaches can be used to select a *forward node set* (i.e., a small set of nodes that forward the broadcast packet). The probabilistic approach [6], [17] normally offers a simple scheme in which each node, upon receiving a broadcast packet, forwards the packet with probability p. The value p is determined by relevant information gathered at each node. When p is carefully selected, a high delivery ratio can be achieved. However, the probabilistic approach cannot guarantee full coverage, with or without mobility and collision. In order to achieve a reasonably high delivery ratio, the selection of p is usually conservative and yields a relatively large forward node set. In the deterministic approach, the forward node set can be selected statically (based on topology information only) [3], [4], [13], [19] or dynamically (based on both topology and broadcast state information) [7], [9], [11], [12], [15], [16]. The forward node set forms a *connected dominating set* (CDS) and guarantees full coverage. A connected dominating set is a subset of connected nodes in the network where every node is either in the subset or a neighbor of a node in the subset. It has been proven that the task of finding a minimal CDS with global network information is NP-complete. The problem is even more challenging in the absence of global network information/infrastructure. Heuristic methods are normally used to balance cost (in collecting network information and in decision making) and effectiveness (in deriving a small forward node set).

Many distributed broadcast algorithms with no use of global information/infrastructure have been proposed for use in ad hoc networks and can be further divided into *self-pruning* and *neighbor-designating* algorithms. In self-pruning algorithms [3], [4], [11], [15], [16], [19], each node makes its local decision on *forward status* (i.e., whether it is a forward node or nonforward node). In neighbor-designating methods [7], [9], [12], [13], the forward status of each node is determined by its neighbors. Different assumptions and models have been used. So far, no generic framework can capture a large body of distributed broadcast algorithms. The only exception is our generic self-pruning scheme proposed in [18]. However, this scheme does not cover neighbor-designating algorithms and its correctness under a dynamic situation is yet to be proven.

In this paper, we provide a generic framework that covers most deterministic distributed broadcast schemes in ad hoc networks, including self-pruning, neighbor-designating, and

• *The authors are with the Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431. E-mail: Email: {jie, fdai}@cse.fau.edu.*

Manuscript received 11 Sept. 2003; revised 30 Jan. 2004; accepted 17 Mar. 2004.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-0153-0903.

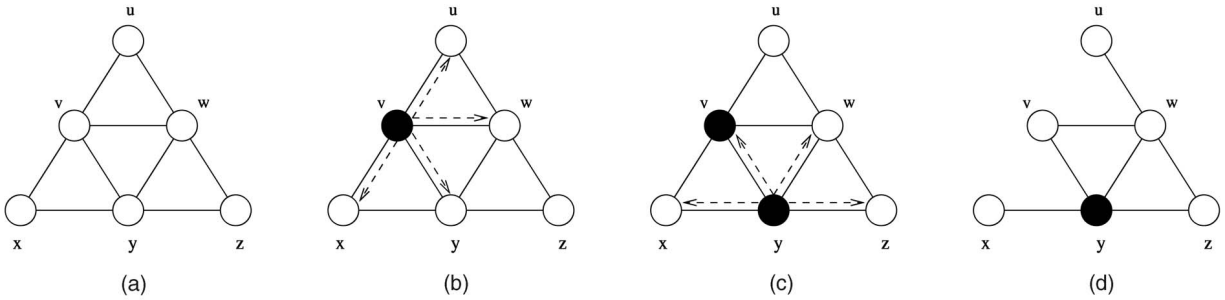

Fig. 1. A sample ad hoc network with three different views.

new hybrid algorithms. The correctness of this generic protocol is proven under both static and dynamic views. In this framework, the status of each node, forward or nonforward, is determined locally based on k-hop neighborhood information. The broadcast packet can carry a small amount of broadcast state information such as recently visited nodes. Broadcast algorithms based on global network topology [5] or pseudoglobal network topology (that exhibits "sequentialized propagation") [2] do not provide scalability and are not included for further discussion. A comprehensive classification of broadcast schemes in ad hoc networks can be found in [20]. Under the objective of selecting a small CDS, the status of each node is decided in a decentralized manner based on a particular *view*, which is a snapshot of the network state, including network topology and broadcast state, along time. Views can be sampled at different times and they can be *local views* that include the connectivity and broadcast state of only nodes in the vicinity. In the generic framework, the status of a node can be decided by itself or by other nodes (say neighbors). Each node has the forward status by default, like in flooding, and the status can be changed to nonforward if the proposed sufficient condition, called *coverage condition*, is met. In addition, such a condition can be easily checked locally around the node. Several existing broadcast algorithms can be viewed as special cases of the generic framework under local views with 2 or 3-hop neighborhood information.

Note that our goal is not to provide an ultimate solution for deterministic broadcasting in ad hoc networks, but rather a solid foundation for future work. The benefit of having such a generic framework is multifold. First, it simplifies the correctness proof, which is a tedious task in previous work. Under the generic framework, one only needs to show that the coverage condition is satisfied. Second, it is easier to apply new techniques to existing broadcast protocols. A new enhancement (e.g., mobility management) developed for the coverage condition can also used in its special cases. Finally, by fine tuning of implementation options, the generic framework can derive new protocols that are more efficient than any single existing protocol. In this paper, various implementation options and their potential overheads are discussed in detail and their impacts on efficiency are evaluated via a comprehensive simulation study. Simulation results confirm that the generic broadcast protocol outperforms most existing protocols in all circumstances.

2 Preliminaries

For each broadcasting, the status of each node can be determined proactively based on neighborhood information only (called the *static approach*) or reactively based on both neighborhood and broadcast state information (called the *dynamic approach*). The dynamic approach is usually more efficient in reducing the size of a CDS. On the other hand, the static approach produces a relatively stable CDS, which forms a virtual backbone that facilitates both broadcasting and unicasting.

In formal terms, an ad hoc network is represented by a unit disk graph $G(t) = (V, E)$, where two vertices (nodes) are connected if their geographical distance is within a given transmission range r. Note that $G(t)$ is a function on time t. A *global view* with respect to a particular broadcast process is a snapshot of network topology and broadcast state. More formally, $View(t) = (G(t), Pr(V, t))$, where $Pr(V, t)$ is a priority vector of nodes in V at time t. The status of each node is determined based on a particular view $View(t)$. We assume the network topology does not change during the broadcast period, so $G(t)$ can be simply represented as G. The priority of each node $v \in V$, $Pr(v, t)$, is a tuple $(S(v, t), id(v))$, where $S(v, t)$ represents the forward status of v under $View(t)$ and $id(v)$ is the distinct identifier of node v (other parameters such as node degree can be used in place of node id).

A node that has forwarded the broadcast packet is called a *visited node*. In neighbor-designating protocols, a node designated by its neighbors to forward the broadcast packet is also viewed as a visited node. In a global view, $S(v, t) = 2$ is reserved for a visited node v and $S(v, t) = 1$ for an unvisited node (i.e., a visited node has a higher priority than an unvisited node under the lexicographical order). In this case, $Pr(v, t)$ is a monotonically increasing function along the time. In the subsequent discussion, t is omitted with an understanding that all terms are with respect to a particular view. An unvisited node is called a *forward node* if it is temporarily determined to forward the broadcast packet under the current view; otherwise, it is called a *nonforward node*. Both visited/unvisited and forward/nonforward status are time sensitive. The difference is that the visited/unvisited status is declared among neighbors, while the forward/nonforward status is the internal state of each node and does not appear in local views of other nodes. For each broadcasting, every node is initially a forward node. A forward node under the current view may become a visited node (if it has forwarded the broadcast packet) or a nonforward node (if a certain condition is satisfied) in the next view. On the other hand, a visited node or a nonforward node never becomes a forward node in the next view. The broadcast process completes when all nodes are either visited nodes or nonforward nodes. We say a

broadcast algorithm ensures coverage if the visited nodes form a CDS in the end of every broadcasting.

Fig. 1 shows an example of view changes during a broadcast process initiated from v, where the network topology remains unchanged and visited nodes are colored black. The priority vectors are the following:

$$\begin{aligned} Pr(V) &= (Pr(u), Pr(v), Pr(w), Pr(x), Pr(y), Pr(z)) \\ &= ((1,u),(1,v),(1,w),(1,x),(1,y),(1,z)) \end{aligned}$$

for Fig. 1a, $Pr(V) = ((1,u),(2,v),(1,w),(1,x),(1,y),(1,z))$ for Fig. 1b, and $Pr(V) = ((1,u),(2,v),(1,w),(1,x),(2,y),(1,z))$ for Fig. 1c. The lexicographical order can be used to order nodes based on their priorities, e.g., $(1,w) > (1,v)$ and $(2,v) > (1,w)$.

In ad hoc networks, a *local view* at node v_i is a more realistic model to determine the node status of v_i. A view is local at node v if node v_i can only capture part of a view within its vicinity. Specifically, a local view, $View_i = (G_i, Pr_i(V))$, of $View = (G, Pr(V))$ meets the following conditions: $G_i = (V(v_i), E(v_i))$ is a subgraph of $G = (V, E)$ and $Pr_i(V) \leq Pr(V)$, that is, each element $Pr_i(v)$ is no more than the one in $Pr(v)$. $Pr_i(v)$ is defined as follows: $Pr_i(v) = Pr(v)$ if $v \in V(v_i)$; otherwise, $Pr_i(v) = (S(v) = 0, id(v))$ (i.e., an invisible node under the local view has the lowest priority). Fig. 1d shows a local view at node z of the global view in Fig. 1c. Here, we assume that, for some reason, links (u,v) and (v,x) and the visited status of node v are unknown at node z.

A *static view* is a view without any visited node. Applying the static approach (i.e., with static views only) to the example in Fig. 1a, any two of nodes v, w, and y can be selected to form a forward node set. Node priority (i.e., node id) can be used to break a tie. Suppose w and y (the highest ids among the six) are selected. When the source is w, w and y form a forward node set. When the source is v, v, w, and y form a forward node set. In the dynamic approach, not only topology information but also the distribution of visited nodes are used to select forward nodes.

Throughout the paper, it is assumed that each node only captures a local view, which is a subgraph of the original graph that consists of at least its entire 1-hop neighbor set, and its priority vector is no more than that of the global view (i.e., an unvisited node will not be treated as a visited node). Note that any visited nodes are assumed to be connected under any local view since they are all connected to the source. Five additional assumptions are used:

1. There is no error in packet transmission, that is, each message (broadcast packet or network state message) sent from a node will eventually reach its neighbors.
2. Location information of each node is not available. Location-based broadcasting has been extensively studied as in [15].
3. Network topology is a connected graph without unidirectional links. A sublayer can be added [14] to provide a bidirectional abstraction for unidirectional ad hoc networks.
4. All nodes have fresh topology information in their local views at the beginning of the broadcast period and the network topology does not change during the broadcast period. Note that, if the network topology changes during the broadcast period, no broadcast algorithm (including flooding) can ensure full coverage. Although some existing reliable broadcast protocols [1], [10] can be applied to guarantee full coverage through transmission redundancy and confirmation, they are beyond the scope of this paper.
5. The network is relatively sparse. For a dense ad hoc network, the *clustering approach* [8], [20] can be used to convert the dense graph to a sparse one.

3 The Generic Coverage Condition

In the generic distributed broadcast protocol, each node has the forward status by default, like in flooding. However, the status of a node can be nonforward if the following sufficient condition, called *coverage condition*, is met: We start with the coverage condition where the status of all nodes is determined under one single view. The result is then extended to the case where the status of each node is determined under a distinct local view.

Coverage Condition. Node v has a nonforward status if, for any two neighbors u and w, a *replacement path* exists that connects u and w via several intermediate nodes (if any) with higher priorities than that of v.

The coverage condition indicates that, when every pair of neighbors of v can be connected through nodes with higher priorities, node v, as the connecting node for its neighbors, can be replaced (i.e., can take the nonforward status). A replacement path may include some visited nodes that have the highest priorities. Note that "replacement" can be applied iteratively. To avoid possible "cyclic dependency" among replacement paths, $Pr(v)$ is used to establish a total order among different replacement paths. Intermediate nodes may not exist. In this case, u and w are directly connected. In formal terms, assume that v is a nonforward node. Let $N(v)$ be the open neighbor set of node v (i.e., v's neighbor set excluding v). For any $u, w \in N(v)$, a replacement path $(u, u_1, u_2, \ldots, u_l, w)$ exists such that $Pr(u_i) > Pr(v)$. Next, we define a special replacement path, called a *maximal replacement path*, such that all intermediate nodes (if any) are either forward nodes or visited nodes. That is, none of the nodes in the maximal replacement path can be replaced under the current view.

Definition 1. Max-min node for (u, w, v): *A* min node *in a path is a node with the lowest priority. Assume $\{P_i\}$ is the set of paths connecting u and w and each node in a path in the set has a higher priority than that of v. A* max-min node *in $\{P_i\}$ is a node with the highest priority among all min nodes in $\{P_i\}$.*

Next, we define a procedure called MAXMIN to construct a maximal replacement path for v connecting u and w.

MAXMIN(u, w, v):

1. **if** u and w are directly connected **then return** $\emptyset$.
2. Find the max-min node x for (u, w, v).
3. **return** path (MAXMIN$(u, x, v), x,$ MAXMIN(x, w, v)).

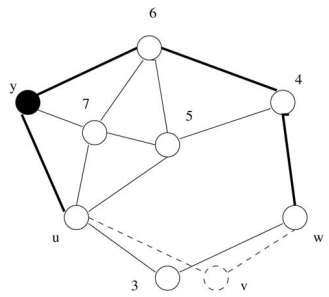

Fig. 2. A maximal replacement path for v connecting u and w.

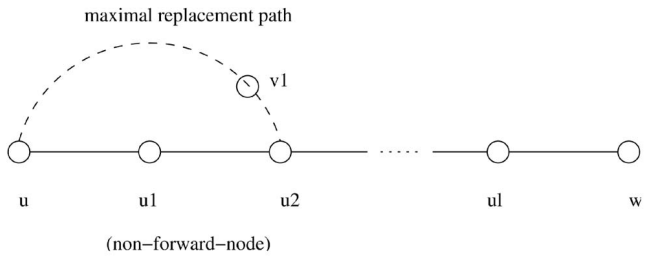

Fig. 3. Maximal replacement path for u_1.

Lemma 1. *The procedure MAXMIN(u, w, v) will complete in a finite number of steps and generate a maximal replacement path.*

Proof. First, we show that all nodes generated by MAXMIN(u, w, v) are distinct. Based on the min node definition, x has the lowest priority among all nodes in replacement paths connecting u and x (and ones connecting x and w). Therefore, x will not be selected as the max-min node in either MAXMIN(u, x, v) or MAXMIN(x, w, v). To show that MAXMIN(u, x, v) and MAXMIN(x, w, v) have no common element, we assume that MAXMIN$(u, x, v) = u_1, u_2, \ldots, u_l$ and MAXMIN$(x, w, v) = x_1, x_2, \ldots, x_m$. Suppose $u_i = x_j$, then $(u, u_1, \ldots, u_i, x_{j+1}, \ldots, x_m, w)$ is a replacement path for v connecting u and w. The fact that all the nodes in this path have a higher priority than x contradicts the fact that x is a max-min node. Since each recursive call of the max-min procedure selects a distinct node, this process will complete in finite steps.

Next, we show that x cannot be further replaced (i.e., x will be a forward node under the current view if it is not a visited node). If x is replaced by path P, then (MAXMIN(u, x, v), P, MAXMIN(x, w, v)) is another replacement path for v that connects u and w (if it is a *walk* with multiple occurrences of some nodes, multiple occurrences can be easily removed to form a path). Clearly, all the nodes in this path have higher priorities than x, which contradicts the fact that x is a max-min node. □

Fig. 2 shows a sample maximal replacement path constructed from the max-min procedure by including u and v at the two ends. In this example, $id(v) = 2$. Nodes with priorities lower than the one of v are not shown. Node 4 is the max-min node for (u, w, v). Node 6 is the max-min node for $(u, 4, v)$ and visited node y is the max-min node for $(u, 6, v)$. Therefore, the maximal replacement path is $(u, y, 6, 4, w)$.

Theorem 1. *Given a graph $G = (V, E)$ that is connected but not a complete graph, the forward node set V' (including forward nodes and visited nodes), derived based on the coverage condition, forms a connected dominating set of G.*

Proof. We first show that V' forms a dominating set. Randomly select a node $v \in V$. We show that v is either in V' or adjacent to a node in V'. If v is a visited node or a neighbor of a visited node, the theorem holds; otherwise, if v is a forward node under the current view, the theorem also holds. For the remaining case, we will show that there exists a forward neighbor. Since v is a nonforward node, for any two neighbors of v, there is a replacement path for v connecting these two neighbors. There exists at least one neighbor u of v such that there is $w \in N(u)$, but $w \notin N(v) \cup \{v\}$ (otherwise, G is a complete graph). Let u be such a neighbor with the largest id. Clearly, there is no replacement path for u connecting v and w and, hence, u is a forward node.

Next, we show that V' is connected. Randomly select two nodes u and w in V'. Assume that $(u, u_1, u_2, \ldots, u_l, w)$ is a path in G connecting u and w. Find a maximal replacement path for u_1 that connects u to u_2. Assume that v_1 is the last intermediate node of the maximal replacement path. Note that $v_1 = u_1$ if u_1 is not replaceable. Repeat the above process on $(v_1, u_2, \ldots, u_l, w)$ to replace u_2 (see Fig. 3). Eventually, $u_1, u_2, \ldots,$ and u_l are all replaced or skipped and the resultant path connects u and w with forward nodes and visited nodes only (if it is a walk with multiple occurrences of some nodes, multiple occurrences can be easily removed to form a path).□

Note that, when the network is a complete graph, there is no need of a forward node. One transmission from the source reaches all the nodes. Theorem 1 shows the result under one particular view, i.e., each node takes the same view in deciding its status. Suppose each node $v_i \in V$ decides its status under a distinct local view, $View_i$. The following theorem shows that Theorem 1 still holds.

Theorem 2. *If each node v_i applies the coverage condition under a local view, $View_i$, Theorem 1 still holds.*

Proof. Let $f_i(v_i)$ be a Boolean variable representing the forward status of node v_i under $View_i = (G_i = (V(v_i), E(v_i)), Pr_i(V))$: 1 for forward and 0 for nonforward. Each G_i is a subgraph of G. $F = (f_1(v_1), f_2(v_2), \ldots, f_n(v_n))$ captures the forward status of all nodes in the network under their corresponding local views. Define $View_{super} = (G_{super}, Pr_{super}(V))$, where

$$\begin{aligned} G_{super} &= (V_{super}, E_{super}) \\ &= (V(v_1) \cup V(v_2) \cup \ldots \cup V(v_n), \\ &\quad E(v_1) \cup E(v_2) \cup \ldots \cup E(v_n)) \end{aligned}$$

and

$$Pr_{super}(v_i) = max\{Pr_1(v_i), Pr_2(v_i), \ldots, Pr_n(v_i)\}.$$

Note that each $Pr_l(v_i)$ has three potential forms/values: the priority of an invisible node $(0, id(v_i))$, the lowest priority; the priority of an unvisited node $(1, id(v_i))$; and the priority of a visited node $(2, id(v_i))$,

the highest priority. Suppose v_i is a nonforward node in $View_i$. In addition, when each node has the accurate k-hop information, all 1-hop neighbors of v_i in $View_{super}$ are also its neighbors in $View_i$. Based on the coverage condition, all neighbors of v_i are connected via replacement paths. Because $Pr_i(V) \leq Pr_{super}(V)$ and G_i is a subgraph of G_{super}, each replacement path in $View_i$ is also a replacement path in $View_{super}$. Therefore, v_i is also a nonforward node in $View_{super}$. That is, $f_{super}(v_i) \leq f_i(v_i)$ and

$$F_{super} = (f_{super}(v_1), f_{super}(v_2), \ldots, f_{super}(v_n)) \leq F = (f_1(v_1), f_2(v_2), \ldots, f_n(v_n)).$$

Applying Theorem 1 to $View_{super}$, the forward node set under F_{super} forms a connected dominating set. Clearly, the forward node set under F also forms a connected dominating set. □

A node that takes the forward status under a global view must also take the same status under a local view, but not vice versa. Therefore, the forward node set under different local views is a superset of the one under the global collective view of all local views.

4 Discussion

We elaborate more on the coverage condition based on four important aspects of its application: timing, selection, space, and priority. To simplify the discussion, under a particular view, each visited node is colored black (called a black node) and all the other nodes are colored white.

4.1 Timing

A broadcast protocol is called *static* if the forward/nonforward status of each node is determined on the *static view* (i.e., without visited node information) only; otherwise, it is *dynamic*. The static broadcast protocol is a special case of the dynamic one. The difference is that the forward node set derived from static views can be used in any broadcasting while the one derived from dynamic views is normally used in a specific broadcasting. There are two types of dynamic algorithms: 1) *First-receipt*: The status is determined right after the first receipt of the broadcast packet. 2) *First-receipt-with-backoff*: The status is determined after a *backoff delay* of the first receipt of the broadcast packet. A backoff delay is used so that a node can learn more about the broadcast state from its forward neighbors. However, this is done at the cost of prolonging the completion time of the broadcast process.

Fig. 4 shows two examples of forward node set on the same network: one without broadcast state based on the static version of the coverage condition (Fig. 4a) and one with broadcast state based on the dynamic version of the coverage condition (Fig. 4b). In the example with broadcast state, it is assumed that the upstream broadcast state is piggybacked with the broadcast packet. The forward node set derived from Fig. 4a can also be interpreted as the one for any broadcasting. In Fig. 4b, because nodes 2 (source) and 5 are visited nodes, node 3 can conclude that it can be a nonforward node since two of its neighbors can be connected using node 2 (a black node). Note that, if the status of node 3 is decided (as a forward node) before the broadcast process starts at node 2 or before it learns the broadcast state, it can still be changed to the nonforward status as long as it has not sent out its status (i.e., no other node has used the status of node 2 in its decision).

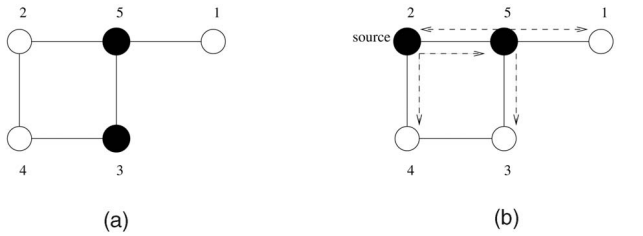

Fig. 4. (a) Forward node set without broadcast state (static). (b) Forward node set with upstream broadcast state (dynamic) with node 2 being the source (visited node).

4.2 Selection

The coverage condition only states the condition under which a node v can be labeled nonforward. The selection issue deals with who should check this condition (and, hence, determine the status) for v. There are three choices: 1) *Self-pruning*. The status of v is determined by node v itself. 2) *Neighbor-designating*. The status of v is determined by some other nodes, say neighbors of v. 3) *Hybrid*. The status of v is determined by both v and neighbors of v.

In the self-pruning approach, each node v determines its status using the coverage condition. In the neighbor-designating approach, each node v determines the status of all its neighbors. In case a node has multiple status as selected by its neighbors, it will forward once (and only once) if at least one status is forward. This requirement, however, can be further relaxed. We can redefine the status function $S(v,t)$ in the priority tuple $Pr(v,t) = (S(v,t), id(v))$ as follows: $S(v,t) = 2$ for visited node v, $S(v,t) = 1.5$ for an unvisited but designated node, and $S(v,t) = 1$ for an unvisited and undesignated node. A designated node does not need to forward the packet if it meets the coverage condition. In the hybrid approach, each node determines the status of some of its neighbors and leaves other neighbors to determine their own status.

4.3 Space

A view consists of network topology and broadcast state information (visited status of some nodes). Network topology information is relatively long lived and can be collected through periodic "hello" messages exchanged among neighbors. In an ad hoc network, it is too expensive to collect global network topology. A local view of network topology, in terms of k-hop *neighborhood information* (or simply k-hop *information*), is used as an approximation. The notion of k-hop information is often used liberally in literature and its meaning varies in different circumstances. To simplify the discussion, we give a definition as follows.

Definition 2. *Given a node v, its local view of network topology $G_k(v)$ is said to contain k-hop information if it takes at least k rounds of neighborhood information exchanges to build up.*

If the neighborhood information is collected via periodically exchanging "hello" messages, it takes k rounds for each node to collect its k-hop information. It is clearly

impossible to collect up-to-date network topology information for a large k; therefore, k is usually a small integer such as 2 or 3 in ad hoc networks.

The maximum subgraph $G_k(v)$ that can be derived from k-hop information is $(N_k(v), E_k(v))$, where $N_k(v)$ is the k-hop close neighbor set of node v. That is, $N_0(v) = \{v\}$ and $N_{k+1}(v) = N_k(v) \cup (\bigcup_{u \in N_k(v)} N(u))$, for $k \geq 0$. On the other hand, $E_k(v)$ does not contain all the links between its k-hop neighbors. Specifically, $E_k(v) = E \cap (N_{k-1}(v) \times N_k(v))$. Links between two nodes that are exactly k hops away from v do not belong to $E_k(v)$. For example, the 1-hop information of v is $G_1(v) = (N_1(v), \{(v,w)|w \in N(v)\})$. Note that $N(v) = N_1(v) - \{v\}$ is v's open 1-hop neighbor set. Link (u,w) between two nodes u and w in $N_1(v)$ is in $G_2(v)$, but not in $G_1(v)$.

Broadcast state information is relatively short lived and cannot be collected through relatively infrequent "hello" messages. Instead, such information can be collected through the following two means: 1) *Snooped*. Each node can snoop the activities of its neighbors. When a neighbor forwards the broadcast packet, it becomes a visited node. 2) *Piggybacked*. When a node forwards the broadcast packet, it also attaches information of some visited nodes (including designated forward neighbors). There are two ways that a forward node v piggybacks visited node information: a) v piggybacks information of visited nodes passed from its predecessor and b) v piggybacks information about designated forward neighbors that have been selected by v, that is, neighbors that will forward the packet. We normally assume that network topology information is not piggybacked since the broadcast packet needs to be kept relatively small.

4.4 Priority

The priority function used in the coverage condition can also affect the resulting forward node set. Based on the difficulty in collecting the priority values, the node properties that are used in the priority function can be divided into three categories:

0-hop priority. Node id, $id(v)$, represents a distinct identification of node v as used now in $Pr(v)$. Node id can be obtained without neighborhood topology information. Ids of nodes in $N_k(v)$ can be collected together with $N_k(v)$ and no extra round of "hello" message exchange is needed. Therefore, it is the least expensive, but it is also the least efficient one in reducing the forward node set.

1-hop priority. Node degree, $deg(v)$, is defined as the number of v's neighbors, i.e., $|N(v)|$. The higher the node degree of a node, the higher the priority. Node degree is based on 1-hop information. If k-hop information is collected together with node id, an extra round of information exchange is required before neighborhood information converges. Therefore, k-hop information plus node degree for each node in $N_k(v)$ requires $(k+1)$-hop information. Node degree is more efficient in reducing the forward node set, but it is also more expensive than 0-hop priority. When $deg(v) = deg(u)$, the ids of u and v are usually used to break a tie.

2-hop priority. Neighborhood connectivity ratio, $ncr(v)$, is the ratio of pairs of neighbors that are not directly connected to pairs of any neighbors. That is,

$$ncr(v) = 1 - \frac{\sum_{u \in N(v)} |N(u) \cap N(v)|}{deg(v)(deg(v)-1)}.$$

Again, the higher the $ncr(v)$ value of node v, the higher the priority. Using $ncr(v)$ as the priority value is the most efficient in reducing the forward node set, but it is also the most expensive. Collecting $ncr(u)$ for each node u in $N_k(v)$ requires $(k+2)$-hop information. When $ncr(v) = ncr(u)$, node degrees followed by node ids of u and v are usually used to break a tie.

In neighbor-designating protocols, a designated node v is usually required to forward the broadcast packet. However, this is not necessary if neighbors of v are connected via other designated nodes with higher priorities. Any priority function discussed above can be used. MPR [13] uses a special priority, called *designating time*, for a designated node v, which is the time v is designated as a forward node. If v is designated several times in a broadcast process, then the first time is used as the designating time. Nodes with smaller designating time have higher priority. For example, suppose node v receives a broadcast packet twice, first from u and then from w. If v is designated to forward the packet by w, but not by u, v will not forward the packet because u and nodes designated by u have smaller designating times than v and $N(v)$ can be connected via those nodes with higher priorities than v.

5 A Generic Distributed Broadcast Scheme

Here, we propose a generic distributed broadcast scheme based on the coverage condition. This is a dynamic approach in which a connected forward node set is constructed for a particular broadcast request and it depends on the *location of the source and the progress of the broadcast process*. We assume that each node v determines its status and the status of some of its neighbors "on-the-fly" under a local view. The source node always forwards the packet. The approach can also be used in a static view where a connected forward node set is constructed independent of any particular broadcast process. We also assume that the broadcast packet that arrives at v carries information of h most recently visited nodes, $v_1, v_2, \ldots, v_h$, and the set of designated forward neighbors, $D(v_i)$, selected at each v_i (usually for small h such as 1 or 2). Fig. 5 shows a view at v with regard to the broadcast state information, assuming each node applied the first-receipt approach, and, hence, each node has only one upstream link (with respect to the source). A general case is that each node has more than one upstream link (i.e., the node forwards the broadcast packet after receiving several copies of the packet) and a *reverse forwarding tree* is formed with root v.

Algorithm 1 A Generic Distributed Broadcast Protocol
(for each node v)
1. Periodically v exchanges "hello" messages with neighbors to update local network topology $G_k(v)$.
2. v updates priority information Pr based on snooped/piggybacked messages.
3. v applies the coverage condition to determine its status.
4. **If** v is a nonforward node **then stop**.
5. v designates some neighbors as forward nodes if needed and updates its priority information Pr.
6. v forwards the packet together with Pr.

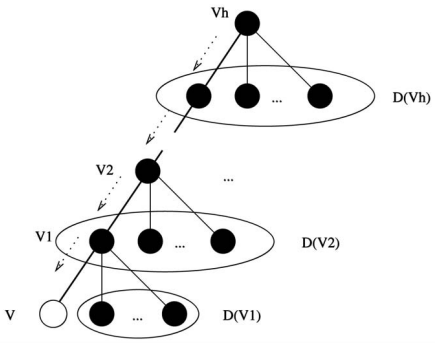

Fig. 5. Broadcast state that includes visited nodes and their designated forward neighbors.

In Algorithm 1, the first two steps collect information to establish a local view. Step 1 collects network topology information of the view while Step 2 collects broadcast state information of the view. In Step 3, each node v determines its status based on the coverage condition. The process stops at Step 4 if v is a nonforward node; otherwise, the view is enhanced by selecting some forward neighbors at Step 5. In addition, v is changed to a visited node. Finally, v forwards the packet at Step 6. The default status for each node is forward, but nondesignated forward node.

6 Special Cases

A large body of existing broadcast protocols can be considered as special cases of our generic distributed broadcast protocol. These special cases take one or more of the following approaches: 1) by skipping some of the steps in the scheme, 2) by using some special cases of the coverage condition at Step 3, 3) by applying a specific strategy in selecting designated forward nodes at Step 5.

One commonly used special case of the coverage condition uses a *coverage set*. A set $C(v)$ is called a coverage set of v if its neighbor set is "covered" by nodes in $C(v)$, that is, $C(v)$ dominates $N(v)$.

Strong Coverage Condition. Node v has a nonforward status if it has a coverage set. In addition, the coverage set belongs to a connected component of the subgraph induced from nodes with higher priorities than that of v.

Clearly, the strong coverage condition is stronger than the original coverage condition because the existence of a connected coverage set implies the existence of a replacement path for any two neighbors. The original coverage condition is more costly to check than the strong coverage condition. The computation complexity of the original coverage condition is $O(D^3)$, while the computation complexity of the strong coverage condition is $O(D^2)$ [18]. Here, D is the density of network (i.e., the maximum number of nodes per unit area). For both versions of the coverage condition, the overhead is high with large D. We assume that high density can be avoided by techniques such as adjustable transmitter range or clustering [8], [20] and, therefore, the computation cost can be kept reasonably small.

In the following, we examine several existing approaches as special cases of our generic distributed broadcast protocol. Special cases are grouped into static and dynamic. Within dynamic approaches, they are further classified as self-pruning, neighbor designating, and hybrid. Some special cases of the coverage condition that do not appear in any of the existing algorithms are also discussed.

6.1 Static Algorithms

The typical static algorithms are the generic distributed broadcast protocol with Steps 1 and 3.

Wu and Li's algorithm. Wu and Li [19] proposed a *marking process* to determine a set of *gateways* (i.e., forward nodes) that form a CDS: A node is marked as a gateway if it has two neighbors that are not directly connected. Two pruning rules are used to reduce the size of the resultant CDS. Based on pruning Rule 1, a gateway becomes a nongateway if all its neighbors are also neighbors of another node, called *coverage node*, that has a higher priority. According to pruning Rule 2, a gateway becomes a nongateway if all its neighbors are also neighbors of either of two coverage nodes that are directly connected and have higher priorities. Two types of priority are used: node id and the combination of node degree and node id. In order to implement the marking process and pruning rules, 2-hop information (if each coverage node is a neighbor) or 3-hop information (if one coverage node is a neighbor's neighbor) is collected at each node.

Dai and Wu's algorithm. Dai and Wu [4] extended the previous algorithm by using a more general pruning rule called Rule k: A gateway becomes a nongateway if all its neighbors are also neighbors of any one of k coverage nodes that are connected and have higher priorities. Rule k can be implemented in a restricted way, which is as efficient as Rule 1 and more efficient than Rule 2, using either 2 or 3-hop information.

Span. Chen et al. [3] proposed an approach, called Span, to construct a set of forward nodes called *coordinators*. A node v becomes a coordinator if it has two neighbors that are not directly connected, indirectly connected via one intermediate coordinator, or indirectly connected via two intermediate coordinators. Before a node changes its status from noncoordinator to coordinator, it waits for a backoff delay, which is computed from its energy level, node degree, and neighborhood connectivity ratio. The backoff delay can be viewed as a priority value such that nodes with a shorter backoff delay period have a higher chance of becoming coordinators. Span cannot ensure full coverage because two coordinators may simultaneously change back to noncoordinators and the remaining coordinators may not form a CDS. To conduct a fair comparison of Span and other broadcast algorithms, we use in the simulation an enhanced version of Span, where a node becomes a coordinator if it has two neighbors that are not directly connected or indirectly connected via one or two intermediate coordinators with higher priority values. 3-hop information is needed to implement Span.

Both Wu and Li's algorithm and Dai and Wu's algorithm use the strong coverage condition and each coverage set consists of nodes with higher priorities. In Span, the original coverage condition is used with the restriction that no replacement path is more than three hops. Fig. 6a shows an example of the difference between the original coverage condition and the strong coverage condition. Node 4 is a nonforward node under the original coverage condition but is a forward node under the strong coverage condition. Note that node 4 is a nonforward node only when the local

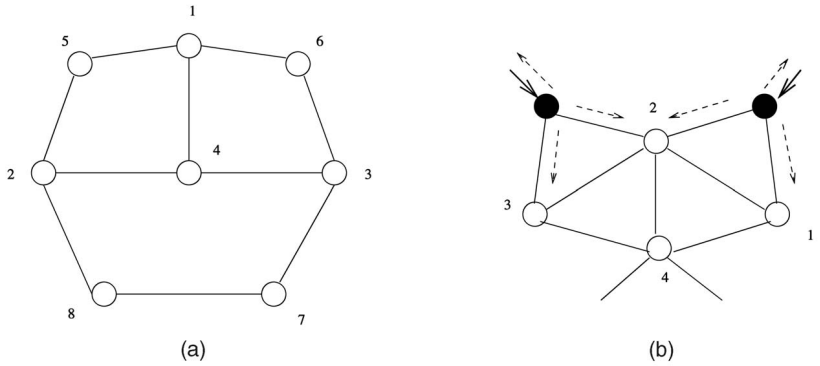

Fig. 6. Two sample ad hoc networks.

view includes 3-hop information. In 2-hop information, link $(7, 8)$ is invisible and the replacement path $(3, 7, 8, 2)$ is unknown to node 4.

6.2 Dynamic and Self-Pruning Algorithms

The dynamic and self-pruning algorithms are usually the generic distributed broadcast protocol with Steps 1, 2, 3, and 6.

SBA. Peng and Lu [11] proposed the Scalable Broadcast Algorithm (SBA) to reduce the number of forward nodes. Unlike the static algorithms, the status of a forward node is computed on-the-fly. When a node v receives a broadcast packet, instead of forwarding it immediately, v will wait for a backoff delay. For each neighbor u that has forwarded the packet, node v removes $N(u)$ from $N(v)$. If $N(v)$ does not become empty after the backoff delay, node v forwards the packet; otherwise, node v becomes a nonforward node. Since all neighbors of a nonforward node are directly connected to a visited node and all visited nodes are connected to the source, each pair of neighbors is connected via a path of visited nodes. Therefore, the coverage condition is satisfied. 2-hop information is used to implement SBA.

Stojmenovic's algorithm. Stojmenovic et al. [15] explicitly applied Wu and Li's CDS algorithm [19] to broadcasting. This algorithm also extends Wu and Li's algorithm in two ways: 1) Suppose every node knows its accurate geographic position, only 1-hop information is needed to implement the marking process and Rules 1 and 2. That is, each node only maintains a list of its neighbors and their geographic positions (connections among neighbors can be derived). 2) The number of forward nodes is further reduced by a neighbor elimination algorithm similar to the one used in SBA. In addition, Stojmenovic et al's algorithm removes an unnecessary round of information exchange in [19], uses node degree as priority, and suggests rebroadcasting after negative acknowledgements.

LENWB. Sucec and Marsic [16] proposed the Lightweight and Efficient Network-Wide Broadcast (LENWB) protocol, which computes the forward node status without using a backoff delay. When a node v receives a broadcast packet from a neighbor u, it computes the set C of nodes that are connected to u via nodes that have higher priorities than v. Node v forwards the packet only when $N(v) \not\subseteq C$. LENWB uses 2-hop information.

Both SBA and Stojmenovic et al.'s algorithm use a special case of the strong coverage condition, where all neighbors are covered by a set of (connected) visited nodes. The first-receipt-with-backoff approach is used. Sucec and Marsic's approach also uses the strong coverage condition with a coverage set consisting of one visited node (black node) and several unvisited but higher priority nodes. In this approach, the first-receipt approach is adopted. Fig. 6b shows a case of neighbor coverage that cannot be covered by SBA and Stojmenovic et al.'s algorithm. After node 2 has two visited neighbors, neighbor 4 is still not covered based on both SBA and Stojmenovic et al.'s algorithm. However, using the strong coverage condition, node 2 is a nonforward node because its neighbor set is covered by white nodes 3, 4, and two black nodes. Note that the two black nodes are viewed as connected in node 2's local view.

6.3 Dynamic and Neighbor Designating Algorithms

The typical dynamic and neighbor designating algorithms are the generic distributed broadcast protocol with Steps 1, 2, 4, 5, and 6. All of the following approaches adopt the greedy strategy where a minimum set of designated forward nodes is selected so that the other neighbors can take the nonforward status.

Dominant pruning. Lim and Kim [7] provided two broadcast algorithms. One of them is based on simple self-pruning, which can be viewed as the first-receipt version of SBA. The other one is based on dominant pruning (DP). The DP algorithm uses 2-hop information to compute the forward node set of each node. Specifically, if u is the last forward node and v is designated as the next forward node, v selects its local forward node set from $X = N(v) - N(u)$ to cover 2-hop neighbors in $Y = N_2(v) - N(u) - N(v)$. The local forward node set is selected using a greedy heuristic algorithm similar to the one solving the set coverage problem.

Multipoint relays. Qayyum et al. [13] proposed selecting multipoint relays (MPRs) as forward nodes to propagate link state messages in their optimized link state routing (OLSR) protocol. The MPRs are selected from 1-hop neighbors to cover 2-hop neighbors, using the same greedy algorithm used by DP. Visited nodes are not considered in the selection of MPRs and, therefore, the entire set of 2-hop neighbors must be covered. MPR can be viewed as a static version of DP and is maintained in a proactive manner. The difference is that a relaxed neighbor-designating requirement is applied to MPR. If an MPR receives a broadcast packet first from a neighbor that is not its designator, it does not need to forward this packet because its neighbor set is covered by MPRs designated by that neighbor. Here, the designating time serves as a priority function to avoid cyclic dependence.

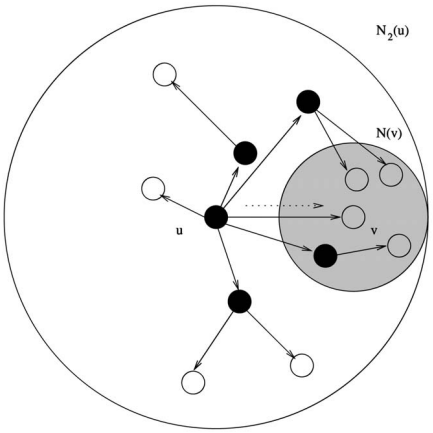

Fig. 7. Neighbor-designating algorithms.

Lou and Wu's algorithm. Lou and Wu [9] extended the DP algorithm by further reducing the number of 2-hop neighbors to be covered by 1-hop neighbors. Two algorithms, total dominant pruning (TDP) and partial dominant pruning (PDP), are proposed. TDP requires the last forward node u piggyback $N_2(u)$ along with the broadcast packet. With this information, the next forward node v can remove $N_2(u)$, instead of $N(u)$ in DP, from $N_2(v)$. PDP, without using the piggybacking technique, directly extracts the neighbors of the common neighbors of u and v (i.e., neighbors of nodes in $N(u) \cap N(v)$) from $N_2(v)$. Simulation results show that PDP avoids the extra cost in TDP introduced by piggybacking 2-hop information with the broadcast packet, but achieves almost the same performance improvement.

All the above approaches designate some 1-hop neighbors to cover 2-hop neighbors so that the remaining 1-hop neighbors become nonforward nodes. As shown in Fig. 7, suppose u is the current node and node v is any neighbor that is not selected as a forward neighbor. Since u and the selected designated forward neighbors cover all the 2-hop neighbors of u which include 1-hop neighbors of v, node v is covered by a set of visited or designated forward nodes.

6.4 Dynamic and Hybrid Algorithms

We consider here a hybrid of self-pruning and neighbor-designating algorithms. The first-receipt approach is still used. Upon receiving a broadcast packet from u with designated forward node set $D(u)$ selected by u, v uses the following steps: If v is not a designated forward node and v has not sent the packet before, then v applies the coverage condition to determine its status. If v is a forward node (self-selected or designated), v selects a neighbor $w \notin u \cup D(u)$ as its designated forward node (if any) based on a certain priority scheme. A neighbor that covers some nodes in $N_2(v)$ is selected with either the lowest id or the maximum effective node degree (with respect to uncovered nodes in $N_2(v)$). Node id is used to break a tie in node degree. Then, v forwards the packet together with $D(v) = \{w\}$. Note that the selected forward neighbor should cover at least one 2-hop neighbor. In this hybrid approach, each node v only uses 2-hop information.

Consider the example in Fig. 8 and suppose nodes 2 and 9 are forwarding the packet to its neighbors. Using self-pruning, nodes 4 and 6 will be forward nodes and nodes 1 and 3 will be nonforward nodes based on the coverage condition with 2-hop information. It is assumed nodes 1 and 6 receive their first copy of the packet from node 2 and node 4 from node 9. Node 3 receives its first copy of the packet from either node 2 or node 9. Using the proposed hybrid approach with node degree as the priority, node 2 is selected as a designated forward node by node 9 and node 6 by node 2. Note that nodes 2 and 9 do not know each other's forward status and, hence, there is no coordination in selecting their designated forward nodes. Node 4 is no longer a forward node since nodes 2 and 9 are visited nodes under the local view of node 4 (passed from node 9). If node id is used as the priority in the hybrid approach, node 2 is selected by node 9. Node 3 is selected by node 2 since node 1 does not cover any 2-hop neighbor of node 2. Once node 3 receives the packet, it will pick node 4 to cover node 7. Using the neighbor-designating approach, node 9 selects node 2 first followed by node 4 to complete the 2-hop coverage. Similarly, node 2 selects node 6 and then node 9.

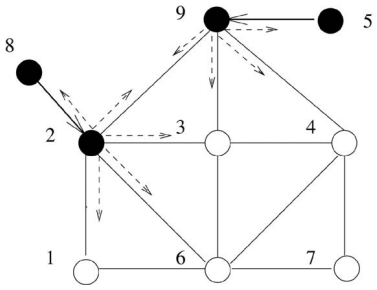

Fig. 8. A sample ad hoc network for different selection policies.

7 Simulation

Our simulation study focuses on two issues: implementation options that build the local views and special cases of the generic framework. Our interest here is on efficiency (i.e., the number of forward nodes) rather than reliability (i.e., the percentage of nodes receiving the broadcast packet). Therefore, all simulations are conducted on static networks with a collision-free MAC layer. Each ad hoc network is generated by randomly placing n nodes in a restricted 100×100 area. The transmitter range is adjusted to achieve a given average node degree d. Two average node degrees are used, one for relatively sparse networks ($d = 6$) and another for relatively dense networks ($d = 18$). The 90 percent confidence interval of each result is within ± 1 percent. Many existing broadcast algorithms use 2-hop information and node id as priority because they have the lowest cost in view update. Unless otherwise specified, they are used in the following simulations. Fig. 9 shows several forward node sets derived from different schemes.

7.1 Implementation Options

Timing. Fig. 10 compares the efficiency of static, first-receipt (FR), and two first-receipt-with-backoff algorithms with a random backoff delay (FRB) and a backoff delay that is proportional to the inverse of node degree (FRBD), respectively. The static algorithm requires less computation and no extra end-to-end delay, but also produces more forward nodes. The FR algorithm causes no extra end-to-end delay, but recomputes the forward/nonforward status for each broadcasting. FR produces fewer forward nodes

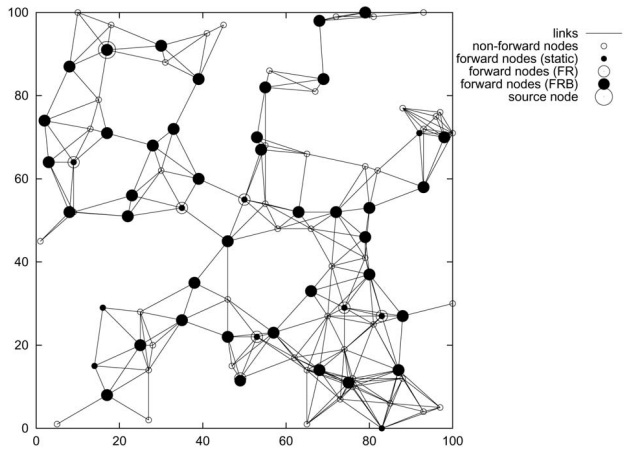

Fig. 9. Broadcasting on a sample ad hoc network of 100 nodes. The numbers of forward nodes are 49, 45, 41 for the static, first-receipt, and first-receipt-with-backoff algorithms, respectively.

than the static algorithm. FRB and FRBD recompute node status for each broadcasting and cause extra end-to-end delay. They produce the smallest forward node set and, between them, FRBD is slightly better than FRB. Since the computation time is negligible in a broadcast process, the dynamic algorithm is more desirable than the static one. Among the dynamic algorithms, FR is appropriate for delay-sensitive applications, and FRBD is appropriate for delay-insensitive applications.

Selection. Fig. 11 compares the efficiency of self-pruning (SP), neighbor-designating (ND), and two hybrid algorithms: The first one designates a neighbor with the highest degree (MaxDeg) and the second one designates a neighbor with the lowest id (MinPri). In the neighbor-designating and hybrid schemes, we use the strict rule that every designated node becomes forward node. In sparse networks, the sequence from the worst performance to the best performance is MinPri, ND, SP, and MaxDeg. In dense networks, when the number of nodes is small ($n \leq 50$), ND, MinPri, and MaxPri stay close and perform better than SP. In larger networks ($n = 100$), ND is worse than MinPri and even worse than MaxDeg and SP. ND algorithm has the lowest computation cost, but performs poorly in relatively dense networks. SP, MaxDeg, and MinPri have almost the same computation cost. Among them, MinPri is the worst in all circumstances, MaxDeg is the best in relatively sparse

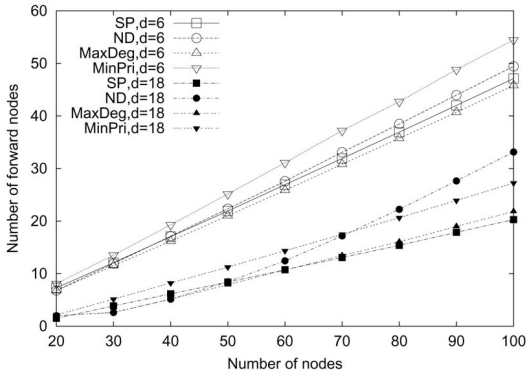

Fig. 11. Selection options.

networks, and SP is the best in relatively dense networks. Note that MaxDeg is a new algorithm derived from the generic framework.

Space. Fig. 12 shows results of using 2-hop, 3-hop, 4-hop, 5-hop, and global information. Although the performance progressively improves as the hop count increases, the difference becomes marginal as the hop count increases. In fact, algorithms based on 2 and 3-hop information do not perform significantly worse than the one based on the global information. Considering the cost in gathering neighborhood information, algorithms based on 4, 5-hop, or global information are not cost-effective compared with the ones based on 2 or 3-hop information.

Priority. Fig. 13 shows results of using node id (ID), node degree (Degree), and neighborhood connectivity ratio (NCR) as priority values. ID requires no extra maintenance cost but produces more forward nodes. Degree has higher maintenance cost and produces fewer forward nodes than ID. NCR has the highest maintenance cost and produces the smallest forward node set. In sparse networks, Degree is much better than ID and is very close to NCR. In dense networks, all three metrics stay very close. Considering the cost of collecting and maintaining degree and NCR information, Degree in relatively dense networks and NCR in general have the worst cost-effectiveness. Trade offs must be made between performance and maintenance cost in selecting ID and Degree in relatively sparse networks.

Overall, there is no single combination of implementation options that is the best for all circumstances. Fine

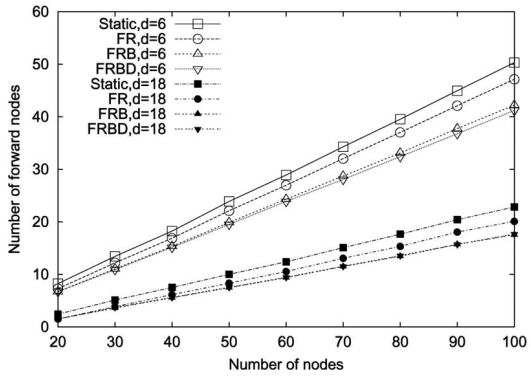

Fig. 10. Timing options.

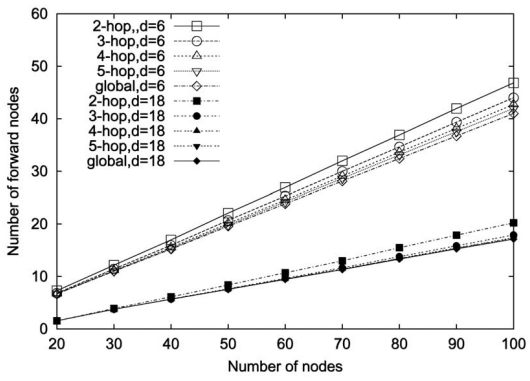

Fig. 12. Space options.

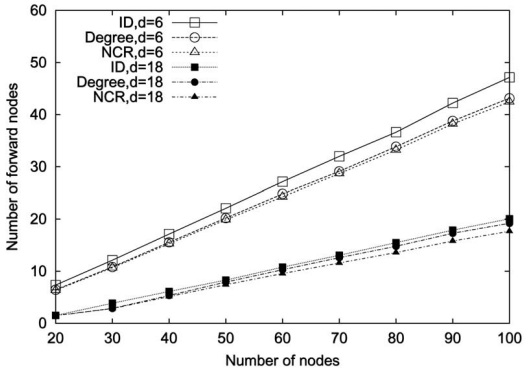

Fig. 13. Priority options.

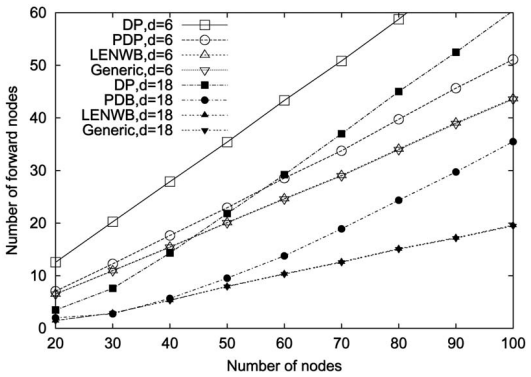

Fig. 15. First-receipt broadcast algorithms.

tuning is needed to achieve better trade off between performance and overhead based on the types of ad hoc networks and applications.

7.2 Special Cases

Here, we compare performances of several existing special cases of the generic framework, including Dai and Wu's algorithm (Rule k), the enhanced Span (Span), MPR, LENWB, dominant pruning (DP), partial dominant pruning (PDP), and SBA. These algorithms can be divided into static and dynamic algorithms, and dynamic algorithms can be further divided into first-receipt and first-receipt-with-back-off algorithms. Each category, except the last one, contains both self-pruning and neighbor-designating algorithms. For the sake of fairness, only algorithms under the same category are compared. Three new algorithms derived from the generic framework using the coverage condition, one for each category, are simulated and compared with existing algorithms. The corresponding performance data of these new algorithms are labeled "Generic" in the result diagrams.

Static algorithms. Fig. 14 compares four static broadcast algorithms. All algorithms except MPR use NCR as the priority value as it is the original configuration of Span. In MPR, the first designator's transmission time is used to as the priority function in reducing the number of forward nodes. The sequence from the worst performance to the best performance is MPR, Span, Rule k, and Generic. MPR is less efficient in relative dense networks because of uncoordinated forward node sets designated by different nodes. Span is slightly worse than Rule k because of its restriction on the length of each replacement path. Generic performs slightly better than Rule k because it uses the original coverage condition and has no constraint on the lengths of replacement paths, while Rule k only uses the strong coverage condition.

First-receipt algorithms. Fig. 15 compares three first-receipt broadcast algorithms. All algorithms use node degree as priority values as it is the original configuration of LENWB. The sequence from the worst performance to the best performance is DP, PDP, LENWB, and Generic. Both DP and PDP are much worse than the other two algorithms because neighbor-designating in general is worse than self-pruning and cannot take advantage of the 1-hop priority. PDP is better than DP since it has fewer 2-hop neighbors to cover than DP does. LENWB is slightly worse than Generic and can be viewed as a good approximation of Generic. Note that LENWB uses less broadcast state information than Generic. In LENWB, only the last visited node is used in checking the strong coverage condition. In Generic, each node also knows the second last visited node that is piggybacked in the broadcast packet. However, this extra broadcast state information has little impact on performance.

First-receipt-with-backoff algorithms. Fig. 16 compares two first-receipt-with-backoff algorithms. Generic significantly outperforms SBA because SBA requires direct neighbor set coverage, while Generic allows indirect coverage. More specifically, a node does not need to forward a broadcast packet, even if some of its neighbors are not directly covered by any visited node but are indirectly

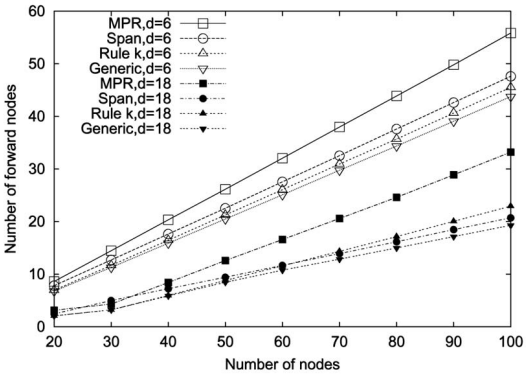

Fig. 14. Static broadcast algorithms.

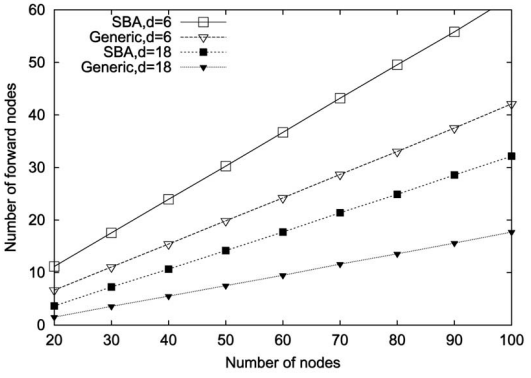

Fig. 16. First-receipt-with-backoff broadcast algorithms.

connected to a visited node via several intermediate nodes with higher priorities.

Overall, within each category, the generic algorithm performs better than existing self-pruning algorithms which, in turn, perform better than existing neighbor designating algorithms.

8 Conclusion

A generic framework of distributed broadcasting in ad hoc networks has been proposed and its correctness has been shown. Four important implementation issues, namely, timing, selection, space, and priority, have been discussed and their impacts on broadcast efficiency have been examined. Nine existing broadcast algorithms, which represent a broad spectrum of state-of-art distributed broadcast techniques in ad hoc networks, have been shown to be special cases of the generic framework. Simulation results show that, by adjusting the four implementation options, the generic distributed broadcast protocol can be well-adapted to different configurations of ad hoc networks and upper layer applications. We have also shown that several new algorithms can be derived from the generic framework and that these algorithms produce smaller forward node sets than existing broadcast algorithms under the same requirement of neighborhood information.

Our future work includes a performance evaluation of the generic broadcast protocol in realistic simulation environments with packet collision and node mobility. In addition, we plan to investigate methods to maintain high delivery ratio in those networks. Another goal is using directional antennas in distributed broadcasting to further reduce the broadcast cost.

Acknowledgments

This work was supported in part by US National Science Foundation grants CCR 0329741, ANI 0073736, and EIA 0130806. The preliminary version of this paper appeared in the Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS), May 2003.

References

[1] S. Alagar and S. Venkatesan, "Reliable Broadcast in Mobile Wireless Networks," *Proc. Military Comm. Conf.*, pp. 236-240, 1995.
[2] K.M. Alzoubi, P.-J. Wan, and O. Frieder, "Distributed Heuristics for Connected Dominating Sets in Wireless Ad Hoc Networks," *J. Comm. and Networks*, vol. 4, no. 1, pp. 22-29, Mar. 2002.
[3] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *ACM Wireless Networks J.*, vol. 8, no. 5, pp. 481-494, Sept. 2002.
[4] F. Dai and J. Wu, "Distributed Dominant Pruning in Ad Hoc Wireless Networks," *Proc. Int'l Conf. Comm.*, May 2003.
[5] S. Guha and S. Khuller, "Approximation Algorithms for Connected Dominating Sets," *Algorithmica*, vol. 20, no. 4, pp. 374-387, Apr. 1998.
[6] Z.J. Haas, J.Y. Halpern, and L. Li, "Gossip-Based Ad Hoc Routing," *Proc. INFOCOM Conf.*, pp. 1707-1716, June 2002.
[7] H. Lim and C. Kim, "Flooding in Wireless Ad Hoc Networks," *Computer Comm. J.*, vol. 24, nos. 3-4, pp. 353-363, 2001.
[8] C.R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *IEEE J. Selected Areas in Comm.*, vol. 15, no. 7, pp. 1265-1275, 1996.
[9] W. Lou and J. Wu, "On Reducing Broadcast Redundancy in Ad Hoc Wireless Networks," *IEEE Trans. Mobile Computing*, vol. 1, no. 2, pp. 111-123, Apr.-June 2002.
[10] E. Pagani and G.P. Rossi, "Providing Reliable and Fault-Tolerant Broadcast Delivery in Mobile Ad Hoc Networks," *Mobile Networks and Applications*, vol. 4, pp. 175-192, 1999.
[11] W. Peng and X. Lu, "On the Reduction of Broadcast Redundancy in Mobile Ad Hoc Networks," *Proc. ACM Symp. Mobile Ad Hoc Networking and Computing*, pp. 129-130, Aug. 2000.
[12] W. Peng and X. Lu, "AHBP: An Efficient Broadcast Protocol for Mobile Ad Hoc Networks," *J. Science and Technology*, 2002.
[13] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint Relaying for Flooding Broadcast Message in Mobile Wireless Networks," *Proc. Hawaii Int'l Conf. System Sciences*, p. 298, Jan. 2002.
[14] V. Ramasubramanian, R. Chandra, and D. Mosse, "Providing a Bidirectional Abstraction for Unidirectional Ad Hoc Networks," *Proc. INFOCOM Conf.*, pp. 1258-1267, June 2002.
[15] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating Sets and Neighbor Elimination Based Broadcasting Algorithms in Wireless Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 13, no. 1, pp. 14-25, Jan. 2002.
[16] J. Sucec and I. Marsic, "An Efficient Distributed Network-Wide Broadcast Algorithm for Mobile Ad Hoc Networks," CAIP Technical Report 248, Rutgers Univ., Sept. 2000.
[17] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network," *Wireless Networks*, vol. 8, nos. 2/3, pp. 153-167, Mar.-May 2002.
[18] J. Wu and F. Dai, "Broadcasting in Ad Hoc Networks Based on Self-Pruning," *Int'l J. Foundations of Computer Science*, vol. 14, no. 2, pp. 201-221, Apr. 2003.
[19] J. Wu and H. Li, "On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks," *Proc. Int'l Workshop Discrete Algorithms and Methods for Mobile Computing and Comm.*, pp. 7-14, 1999.
[20] J. Wu and W. Lou, "Forward-Node-Set-Based Broadcast in Clustered Mobile Ad Hoc Networks," *Wireless Comm. and Mobile Computing*, special issue on algorithmic, geometric, graph, combinatorial, and vector, vol. 3, no. 2, pp. 155-173, 2003.

Jie Wu is a professor in the Department of Computer Science and Engineering at Florida Atlantic University. He has published more than 200 papers in various journal and conference proceedings. His research interests are in the area of mobile computing, routing protocols, fault-tolerant computing, and interconnection networks. He was a co-guest-editor of a special issue in *IEEE Computer* on "Ad Hoc Networks." He also edited several special issues in *Journal Parallel and Distributing Computing* and the *IEEE Transactions on Parallel and Distributed Systems*. He is the author of the text *Distributed System Design* (CRC Press). Currently, he serves as an associated editor for the *IEEE Transactions on Parallel and Distributed Systems* and three other international journals. Dr. Wu is a recipient of the 1996-1997 and 2001-2002 Researcher of the Year Award at Florida Atlantic University. He served as an IEEE Computer Society Distinguished Visitor. He is a member of the ACM and a senior member of the IEEE and the IEEE Computer Society.

Fei Dai received the BS and MS degrees in computer science in 1990 and 1993, respectively, from Nanjing University, Nanjing, China. He is a PhD candidate in the Department of Computer Science and Engineering at Florida Atlantic University. His current research focuses on the design and applicaton of localized algorithms in wireless add hoc and sensor networks. Other research areas include interconnection networks, fault-tolerant routing, and software architecture. He worked as a senior programmer at Greatwall Computer, China, from 1993 to 1996 and as a software architect and team leader at J&A Securities, China from 1996 to 2000. He is a student member of the IEEE and IEEE Computer Society.