Toward Broadcast Reliability in Mobile Ad Hoc Networks with Double Coverage

Wei Lou, *Member*, *IEEE*, and Jie Wu, *Senior Member*, *IEEE*

Abstract—The broadcast operation, as a fundamental service in mobile ad hoc networks (MANETs), is prone to the broadcast storm problem if forwarding nodes are not carefully designated. The objective of reducing broadcast redundancy while still providing high delivery ratio under high transmission error rate is a major challenge in MANETs. In this paper, we propose a simple broadcast algorithm, called double-covered broadcast (DCB), which takes advantage of broadcast redundancy to improve the delivery ratio in an environment that has rather high transmission error rate. Among the 1-hop neighbors of the sender, only selected forwarding nodes retransmit the broadcast message. Forwarding nodes are selected in such a way that 1) the sender's 2-hop neighbors are covered and 2) the sender's 1-hop neighbors are either forwarding nodes or nonforwarding nodes covered by at least two forwarding neighbors. The retransmissions of the forwarding nodes are received by the sender as the confirmation of their reception of the packet. The nonforwarding 1-hop neighbors of the sender do not acknowledge the reception of the broadcast. If the sender does not detect all its forwarding nodes' retransmissions, it will resend the packet until the maximum number of retries is reached. Simulation results show that the proposed broadcast algorithm provides good performance under a high transmission error rate environment.

Index Terms—Broadcast, double dominating set, forwarding node, mobile ad hoc networks (MANETs), performance evaluation, reliability.

1 Introduction

A mobile ad hoc network (MANET) enables wireless communications between participating mobile nodes without the assistance of any base station. Two nodes that are out of one another's transmission range need the support of intermediate nodes, which relay messages to set up a communication between each other. The broadcast operation is the most fundamental role in MANETs because of the broadcasting nature of radio transmission: When a sender transmits a packet, all nodes within the sender's transmission range will be affected by this transmission. The advantage is that, if one node transmits a packet, all its neighbors can receive this message. This scenario is also referred to as "all neighborhood nodes are *covered* or *dominated* by this transmitting node." On the negative side, one transmission may interfere with other transmissions, creating the *exposed terminal problem* where an outgoing transmission collides with an incoming transmission and the *hidden terminal problem* where two incoming transmissions collide with each other.

Blind flooding (BF), where each node forwards the packet once and only once, makes every node a forwarding node. If the forwarding nodes are not carefully designated, they will trigger many retransmissions at the same time, which might congest the network. This is referred to as the *broadcast storm problem* [1]. The fact that *only a subset of nodes forward the broadcast message and the remaining nodes are adjacent to the forwarding nodes* can be used to reduce the broadcast congestion but still fulfill the broadcast coverage. A MANET consists of randomly distributed nodes that result in some regions of the network being very dense and others being very sparse. A careful selection of forwarding nodes, i.e., selecting a similar number of forwarding nodes in both dense and sparse regions of the network, not only reduces the density of the network, but also balances the difference of the density among the different regions of the network. Basically, forwarding nodes form a *connected dominating set* (CDS). A *dominating set* (DS) is a subset of nodes such that every node in the graph is either in the set or is adjacent to a node in the set. If the subgraph induced from a DS of the network is connected, the DS is a CDS. Finding a *minimum connected dominating set* in a given graph is NP-complete; in a unit disk graph, it has also been proved to be NP-complete [2].

MANETs suffer from a high transmission error rate because of the high transmission contention and congestion. Therefore, it is a major challenge to provide high reliability for broadcasting operations under such dynamic MANETs. Unlike the probability-based broadcast algorithms, which only provide probabilistic coverage, we aim to provide full coverage in an ideal error-free environment and very high delivery ratio in a high transmission error rate environment when selecting the forwarding nodes. Usually, acknowledgment messages (ACKs) are used to ensure broadcast delivery. However, the requirement for all receivers to send ACKs in response to the reception of a packet may become another bottleneck of channel congestion and packet collision, which is called the *ACK implosion problem* [3].

Our goal is to reduce the number of forwarding nodes without sacrificing the broadcast delivery ratio. Specifically,

- *W. Lou is with the Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong. E-mail: csweilou@comp.polyu.edu.hk.*
- *J. Wu is with the Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431. E-mail: jie@cse.fau.edu.*

Manuscript received 26 July 2005; revised 10 Feb. 2006; accepted 1 May 2006; published online 14 Dec. 2006.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-0218-0705.

we propose a simple broadcast algorithm, called *double-covered broadcast* (DCB), which takes advantage of broadcast redundancy to improve the delivery ratio in the environment that has rather high transmission error rate. Only a set of selected nodes will forward the broadcast message. The selected nodes, called forwarding nodes, meet the following two requirements: 1) they cover the sender's 2-hop neighbor set, and 2) they cover the sender's 1-hop nonforwarding neighbors at least twice. Also, the retransmissions of the forwarding nodes are received by the sender as the acknowledgement of their reception of the packet.[1] Nonforwarding neighbors do not acknowledge the reception of the broadcast. If the sender fails to detect all its forwarding nodes' retransmissions, it repeatedly resends the packet until it detects that all the retransmissions or the maximum number of retries is reached. The proposed algorithm has many merits, such as balancing the average retransmission redundancy, avoiding both the broadcast storm problem and the ACK implosion problem, recovering the transmission error locally, and increasing the broadcast delivery ratio in a high transmission error rate environment. Simulation results show that the algorithm provides high delivery ratio, low forwarding ratio, low overhead, and low end-to-end delay for a broadcast operation under a high transmission error rate environment.

The remaining part of the paper is organized as follows: Preliminaries and related work are briefly introduced in Section 2. Section 3 describes in detail the double-covered broadcast protocol. In Section 4, we simulate the proposed broadcast protocol by using the ns-2 testbed and compare its performance with other reliable broadcast algorithms. Finally, some conclusions are drawn in Section 5.

2 Preliminaries

We describe a MANET as a unit disk graph $G = (V, E)$, where the node set V represents a set of wireless mobile nodes and the edge set E represents a set of bidirectional links between the neighboring nodes. Two nodes are considered neighbors if and only if their geographic distance is less than the transmission range r. In a localized broadcast protocol, a node v is equipped with a k-hop subgraph $G_k(v)$ for a small k, such as $k = 2$ or 3. $G_k(v)$, induced from the k-hop information of v, is ($N_k(v)$, $E_k(v)$). $N_k(v)$ denotes the k-hop neighbor set of node v which includes all nodes within k hops from v (and also includes v itself). $H_k(v)$ denotes the k-hop node set of v which includes all nodes that are exactly k hops away from v; that is, $N_0(v) = H_0(v) = \{v\}$, $N_k(v) = N_{k-1}(v) \cup H_k(v)$, and $H_k(v) = N_k(v) - N_{k-1}(v)$, for $k \geq 1$. For convenience, the 1-hop neighbor set $N_1(v)$ and the 1-hop node set $H_1(v)$ are represented as $N(v)$ and $H(v)$, respectively. $E_k(v)$ denotes the set of links between $N_k(v)$, excluding those links between $H_k(v)$. That is, $E_k(v) = N_{k-1}(v) \times N_k(v)$. For example, if v has 1-hop neighbor information, then it knows all its neighbors, but not the links between these neighbors. If V is a node set, $N(V)$ is the union of the neighbor sets of every node in V, that is, $N(V) = \cup_{\forall w \in V} N(w)$.

2.1 Neighbor-Designating-Based Broadcasting

Broadcasting algorithms can be classified into probabilistic and deterministic approaches [1]. The probabilistic approaches [1], [5], [6] provide good stochastic results, but do not guarantee full coverage of the network. On the contrary, the deterministic approaches provide full coverage of the network. In the deterministic approaches, Wu and Dai [7] proposed a generic localized broadcast scheme where each node v determines its own status (in self-pruning protocols) and the status of some of its neighbors (in neighbor-designating protocols) under a current local view. A *view*, with respect to a particular broadcast process, is a snapshot of network topology and broadcast state. In a typical neighbor-designating broadcast algorithm, each node v gets its 2-hop neighbor set $N_2(v)$ by including its neighbors in the HELLO message; thus, v can select a subset of nodes in its 1-hop node set $H(v)$ to cover its 2-hop node set $H_2(v)$. In the neighbor-designating broadcast algorithm, the upstream node that has sent a broadcast packet is viewed as a *forwarded node*. A *forwarding node* is a downstream node designated by the current node that will forward the broadcast packet; a *nonforwarding node* is a downstream node that is not designated to forward the packet. The node status under the current view will change in the next view after it forwards the packet; that is, a forwarding node in the current view will be a forwarded node in the next view. Neighbor-designating broadcast algorithms can be further divided into static and dynamic approaches. In a typical static approach, a node becomes an "active" forwarding node that will relay the broadcast packet if it is designated as a forwarding node by its lowest-ID neighbor. In a typical dynamic approach, if a node receives a new broadcast packet for the first time and is designated as a forwarding node, it will relay the packet (See Algorithm 1).

Algorithm 1 Dynamic Neighbor-Designating Broadcast Algorithm

1. When a sender u sends a broadcast packet, it designates some neighbors that form its forwarding node set $F(u)$ to cover its 2-hop node set $H_2(u)$. u sends the packet together with $F(u)$.
2. When a node v receives the packet from u for the first time, if v is not designated as a forwarding node by u, it does nothing; otherwise, it becomes a new sender and goes to step 1.

All of the following algorithms belong to the class of dynamic neighbor-designating broadcast algorithms and adopt the greedy strategy where a minimum number of designated forwarding nodes are selected so that other neighbors can take the nonforwarding status. These algorithms differ in how they select the forwarding node sets, although some of them were not initially designed for dynamic neighbor-designating broadcast algorithms.

In [8], [9], multipoint relays (MPRs) are selected as the forwarding nodes to propagate link state messages. The MPRs are selected from 1-hop neighbors to cover 2-hop

1. In normal approaches, the forwarding nodes send explicit ACKs back to the sender to confirm the reception of the broadcast packet [4]. The advantage of this approach is that the ACKs are very short messages, while broadcast packets are usually large messages. The possibility of collision when sending ACKs is less than when sending broadcast packages.

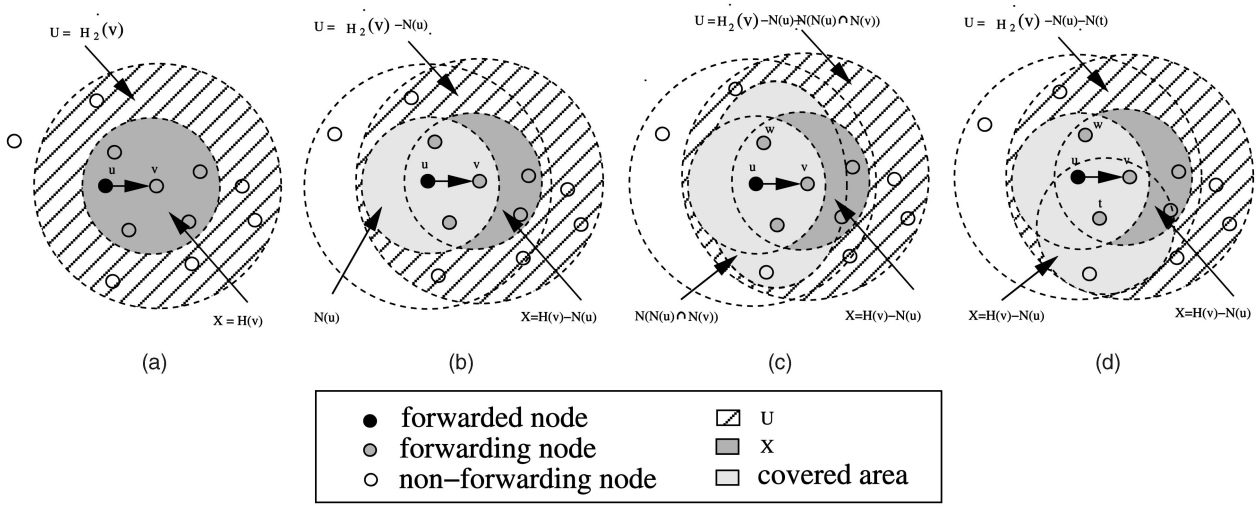


Fig. 1. Illustration of the forwarding node set selection process of four algorithms: (a) multiple relays (MPR), (b) dominant pruning (DP), (c) partial dominant pruning (PDP), and (d) CDS-based broadcasting (CDSB).

neighbors. Forwarded nodes are not considered for a node to select its MPRs and, therefore, the entire set of 2-hop neighbors must be covered (Fig. 1a). Specifically, v selects its forwarding node set F from all candidate neighbors $X = H(v) = N(v) - \{v\}$ to cover its uncovered 2-hop neighbors $U = H_2(v) = N_2(v) - N(v)$ with a simple greedy algorithm used in the set coverage problem [10]. This forwarding node set selection process (FNSSP) is described in Algorithm 2:

Algorithm 2 Forwarding Node Set Selection Process (FNSSP) (for node v)

1. Initially, $X = H(v)$, $U = H_2(v)$, and $F = \phi$.
2. Find w (in X) with the maximum effective neighbor degree $deg_e(w) = |N(w) \cap U|$.
3. $F = F \cup \{w\}$, $U = U - N(w)$, and $X = X - \{w\}$.
4. Repeat steps 2 and 3 until U becomes empty.

Lim and Kim [11] provided a dominant pruning algorithm (DP). Compared to the MPR, the DP excludes the coverage of the forwarded node from the current node's 2-hop neighbor set. Supposing u is the last forwarded node and v is a designated forwarding node of u, v selects its forwarding node set from $X = H(v) - N(u)$ to cover 2-hop neighbor set $U = H_2(v) - N(u)$ (Fig. 1b). Note that the DP does not properly terminate when $N(X) \not\supseteq U$, e.g., some nodes in U can be covered only by nodes in $N(u)$ but not by nodes in $H(v) - N(u)$.

A similar algorithm, called the ad hoc broadcast algorithm (AHBP), was proposed by Peng and Lu [12]. In AHBP, the selected forwarding nodes are called Broadcast Relay Gateway (BRG). BRGs are proactively chosen by each upstream sender, which is a BRG itself. The BRG selection process is identical to the forwarding node selection process that is used in DP. An extended AHBP (AHBP-EX) also takes node mobility into consideration: Supposing node v receives a broadcast packet from node u and v finds $u \notin H(v)$, v will assume itself a BRG and rebroadcast the packet. It generally has a better performance than DP [13].

Lou and Wu [14] proposed a partial dominant pruning algorithm (PDP) to extend the DP by further reducing the number of 2-hop neighbors to be covered by 1-hop neighbors. In the PDP, node v extracts the neighbors of the common neighbors of u and v (i.e., neighbors of nodes in $N(u) \cap N(v)$) from $H_2(v)$, that is, the uncovered 2-hop neighbor set $U = H_2(v) - N(u) - N(N(u) \cap N(v))$ (Fig. 1c).

Peng and Lu proposed a CDS-based broadcast algorithm (CDSB) in [15]. When a node receives a broadcast packet and determines its forwarding node, it excludes the neighbors of the sender of the packet and all the neighbors of the sender's forwarding nodes with lower node IDs to determine its own forwarding node set. Suppose a sender u selects nodes t, v, and w ($id(t) < id(v) < id(w)$) as its forwarding nodes. When nodes t, v, and w receive the packet, t updates its uncovered 2-hop neighbor set $U(t) = H_2(t) - N(u)$; v updates its uncovered 2-hop neighbor set $U(v) = H_2(v) - N(u) - N(t)$ because $N(t)$ is covered by t. Notice that, if $t \in N(v)$, t is a common neighbor of both u and v, v can exclude $N(t)$ from $U(v)$. This becomes a special case of the PDP. If t is two hops away from v, v only excludes t from $U(v)$. Likewise, w's uncovered 2-hop neighbor set is $U(w) = H_2(w) - N(u) - N(t) - N(v)$ (Fig. 1d).

2.2 Reliable Broadcasting

In general, a reliable communication needs some feedback from receivers. Many approaches are provided for reliable communications in wired networks [16], [17], [18], [19]. The basic categories of reliable communication schemes are *sender initiated* and *receiver initiated* approaches [20]. In the sender-initiated approach [16], [19], the receiver returns a positive ACK to the sender for each message it receives. The sender needs to maintain all records for each receiver to confirm the success of the delivery. Only missing packets are retransmitted by the sender, either to individual requested receivers or to all receivers. The drawback of this scheme is that the sender may become the bottleneck of transmission when simultaneous ACKs return. Moreover, the amount of records that the sender must maintain may also grow large. In the receiver-initiated approach [17], [18], the receiver is responsible for reliable delivery. Each receiver maintains receiving records and requests repairs

via a negative acknowledgement (NACK) when errors occur. Several strategies can be applied to the receiver-initiated approach, such as sender-oriented, flat-receiver-oriented, and hierarchical-receiver-oriented approaches. The problem of the receiver-initiated approach is the long end-to-end delay since the sender must wait for the next broadcast packet to determine whether the previous one is successfully delivered or not. Therefore, it can be applied only when the sender has many packets to send.

There are several reliable broadcast schemes [3], [21] which aim to suppress MAC layer collisions and provide reliable MAC layer transmission. In the network layer, most reliable broadcast protocols come from the routing protocol proposed by Merlin and Segall [22]: The source starts a broadcast operation by sending a message to all its neighbors and waiting for the ACKs from its neighbors. When it receives all these ACKs, it sends a message asking the neighbors to propagate the message one more hop to their own neighbors. The neighbors of the source forward the message to their neighbors and send the ACKs back to the source when they receive all ACKs from all their own neighbors, and so forth. The scheme incurs too much communication overhead and needs stable linkages for MANETs.

Alagar et al. [23] propose a reliable broadcast (RB) protocol based on flooding. The protocol works as follows: The source broadcasts the message to its 1-hop neighbors. When a node receives the message, it sends an ACK back to the sender. If the message is a new one, the node retransmits the message; otherwise, it drops the message. If the sender does not receive an ACK from any of its neighbors for a predefined period, it resends the message. In case some links happen to be broken up, a handshake process is provided to make two neighbor nodes exchange all of the messages they have so far to keep all records identical.

Reliable broadcast protocols are also proposed by other researchers. Garcia-Luna-Aceves and Zhang [24] use a flooding-based approach that allows the nodes that received the broadcast packet to forward the packet without further notice from the sender. When the source sends the message, it waits for the ACKs from all its neighbors. A node v (other than the source) that receives the message from its neighbor u forwards the message to the rest of its neighbors and waits for their replies. When node v receives the replies from all those neighbors, it sends its own reply to node u. Topological changes, including link establishments or failures, can be handled with additional treatments. Pagani and Rossi [25] propose to set up a forwarding tree, which is rooted from the clusterhead of the source to each clusterhead, based on a virtual cluster architecture for a reliable broadcast in MANETs. The broadcast packet is forwarded down the tree from the root source to the leaf nodes and the ACKs are collected by each clusterhead and sent up the tree from the leaves to the root. The source retransmits the packet if an error occurs. The algorithm changes to flooding when the rate of topology change of the network becomes high. Apparently, maintaining the underlying cluster and the forwarding tree is almost impractical in dynamic MANETs. Pleisch et al. [26] propose an approach that relies on proactive compensation packets to overcome low-level residual packet losses. The time complexity of the broadcasting operations is investigated in [27], [28], [29], [30].

All these reliable broadcast algorithms require each receiver to send ACKs in response to the reception of a packet. These ACKs may become another bottleneck of channel congestion and lead to severe transmission contention and collision, which is referred to as the ACK implosion problem.

2.3 Double Dominating Set

The concepts of *double dominating set* and *total double dominating set* are provided by some researchers in an attempt to provide a degree of robustness to the network [31]. The double dominating set is a set $S \subseteq V$ if every node not in S is dominated by at least two nodes in S. The total double dominating set is a set $S \subseteq V$ if every node of V is dominated by at least two nodes in S. A centralized tree search type algorithm is applied to construct the double dominating set [32] and some theoretical bounds are given in [31], [33]. In [34], three heuristic algorithms are provided to construct a double dominating set and many related parameters are measured by simulations. A general k-tuple domination is discussed in [31], [35]. The proposed DCB algorithm also provides a double dominating set. Unlike algorithms proposed in [31], [32], this double dominating set is built incrementally when the broadcast packet is disseminated throughout the network. Moreover, the generated double dominating set is also connected.

3 A Double-Covered Broadcast Algorithm

3.1 Basic Idea

A network-wide broadcast requires a packet to be received by all nodes in the network. But, transmission interference and the movement of the nodes may cause some nodes to lose the broadcast packet. The redundancy of the broadcast packet can bring more opportunities for a node to receive the packet successfully. Moreover, if the sender can retransmit the packet, the number of nodes that receive the broadcast packet is also increased.

The proposed double-covered broadcast (DCB) algorithm works as follows: When a sender broadcasts a packet, it selects a subset of 1-hop neighbors as its forwarding nodes to forward the packet based on a greedy approach. The selected forwarding nodes satisfy two requirements: 1) they cover all the sender's 2-hop neighbors, and 2) the sender's 1-hop neighbors are either forwarding nodes or nonforwarding nodes covered by at least two forwarding nodes (e.g., once by the sender itself and once by one of the selected forwarding nodes). After receiving a new broadcast packet, each forwarding node records the packet, computes its forwarding nodes, and rebroadcasts the packet as a new sender. The retransmissions of the forwarding nodes are overheard by the sender as the acknowledgement of the reception of the packet. The nonforwarding 1-hop neighbors of the sender do not acknowledge the receipt of the broadcast. The sender waits for a predefined duration to overhear the rebroadcast from its forwarding nodes. If the sender fails to detect all its forwarding nodes retransmitting

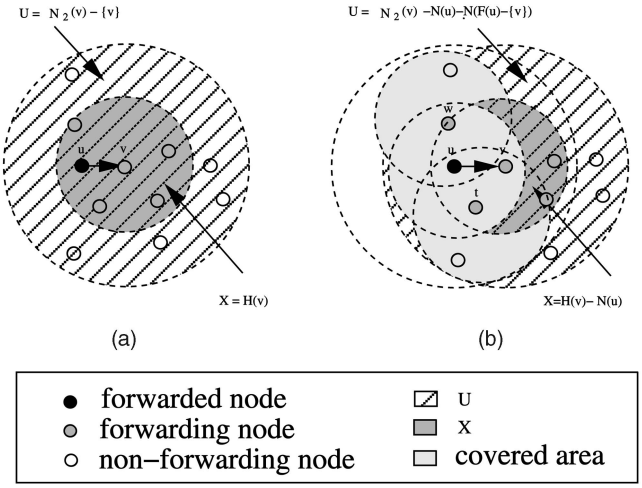


Fig. 2. Illustrations of the forwarding node set selection process of the DCB algorithm at: (a) a source node and (b) a selected forwarding node.

during this duration, it assumes that a transmission failure has occurred for this broadcast. The sender then resends the packet until all the forwarding nodes' retransmissions are detected or the maximum number of retries is reached. The sender may miss a retransmission from a forwarding node, and therefore resends the packet. When the forwarding node receives a duplicated broadcast packet, it sends an ACK to acknowledge the sender.

The DCB algorithm selects a set of forwarding nodes that form a virtual backbone of the network. The forwarding nodes are selected in such a way that they balance the average retransmission redundancy for the delivery of a broadcast packet throughout the entire network. The scheme avoids the broadcast storm problem: Since only the forwarding nodes transmit the packet, the broadcast collision and congestion are both reduced. This scheme also avoids the ACK implosion problem: The retransmissions of forwarding nodes are also used as the ACKs to the sender so that no extra ACKs are needed. The failure of overhearing forwarding nodes' relays will trigger the sender to retransmit the packet, so that the packet loss can be recovered in a local region. Each nonforwarding node is covered by at least two forwarding neighbors so that it can tolerate a single transmission error and its chance to receive the broadcast packet successfully is greatly increased even in a high transmission error rate environment. Moreover, the algorithm does not suffer the disadvantage of the receiver-initiated approach that needs a much longer delay to detect a missed packet.

3.2 Forwarding Node Set Selection Process

We assume that each node v knows its 2-hop subgraph $G_2(v) = (N_2(v), E_2(v))$. A forwarding node v uses the FNSSP-DC (Algorithm 3) to determine its forwarding node set $F(v)$: v uses the FNSSP algorithm (Algorithm 2) to find $F(v)$ in $H(v)$ to cover $N_2(v) - \{v\}$ (Fig. 2a). Unlike the MPR algorithm [8], where only nodes in $H_2(v)$ need to be covered by forwarding node set $F(v)$, the FNSSP-DC algorithm guarantees that v's 2-hop neighbor set $N_2(v)$ (excluding v itself) is completely covered by v's forwarding node set $F(v)$. Since v also transmits the packet to cover $H(v)$, any nonforwarding node in $H(v)$ is covered twice.

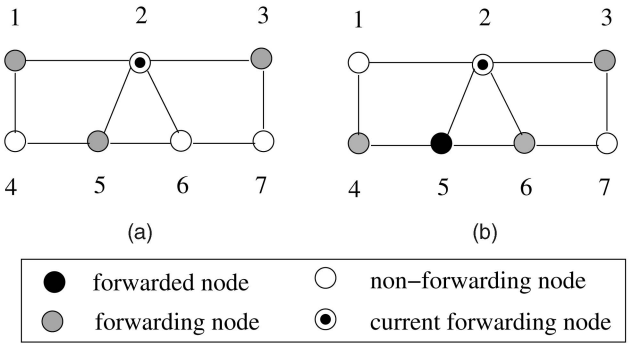


Fig. 3. A sample network where node 2 uses the FNSSP-EDC to select its forwarding nodes. In (a), node 2 is the source and, in (b), node 2 is a selected forwarding node.

Algorithm 3 Forwarding Node Set Selection Process—Double Coverage (FNSSP-DC)

1. Each node v computes $X = H(v)$ and $U = N_2(v) - \{v\}$.
2. Node v uses the FNSSP to find $F(v)$ in X to cover U.

The source of a broadcast operation uses the FNSSP-DC algorithm to determine its forwarding node set. Other forwarding nodes consider the impact of the sender of the broadcast packet. If v is a designated forwarding node of u, that is, v receives a new packet from u and v finds itself in $F(u)$, v uses the FNSSP-EDC algorithm (Algorithm 4) to determine its forwarding node set (Fig. 2b): v finds $F(v)$ in $H(v) - N(u)$ to cover $N_2(v) - N(u) - N(F(u) - \{v\})$. The goal of FNSSP-EDC is to cover all those nodes in the 2-hop neighborhood of v, excluding those that have been already covered by u and those that will be covered by some other forwarding nodes of u.

Algorithm 4 Forwarding Node Set Selection Process—Enhanced Double Coverage (FNSSP-EDC)

1. Each node v sets $X = H(v) - N(u)$ and $U = N_2(v) - N(u) - N(F(u) - \{v\})$.
2. Node v uses the FNSSP to find $F(v)$ in X to cover U.

Theorem 1. *Both FNSSP-DC and FNSSP-EDC algorithms provide a set of forwarding nodes that cover all the nodes within a 2-hop neighbor set and doubly cover the nonforwarding nodes within a 1-hop neighbor set.*

Proof. The FNSSP-DC is a special case of the FNSSP-EDC. The correctness of the FNSSP-EDC is proven as follows: Based on the description of the algorithm, $N_2(v) - \{v\}$ is partitioned into three sets: 1) the set $N(u)$ that is covered by u, 2) the set $N(F(u) - \{v\})$, which is covered by those forwarding nodes of u (excluding v), and 3) the set $U = N_2(v) - N(u) - N(F(u) - \{v\})$ that is covered by v's forwarding node set $F(v)$. Thus, $N_2(v)$ is fully covered. Moreover, $H(v)$ is also covered by v itself and, therefore, $H(v)$ is covered twice. □

For the sample network shown in Fig. 3a, $N(2) = \{1, 2, 3, 5, 6\}$ and $N_2(2) = \{1, 2, 3, 4, 5, 6, 7\}$. When using the

FNSSP-EDC, sender node 2 selects nodes 1, 3, and 5 as its forwarding nodes. Node 1 is selected because there is no node in $N(1)$ to cover it. In Fig. 3b, suppose the source of a broadcast is node 5 and node 2 has received the broadcast packet from node 5. Node 5's forwarding node set is $F(5) = \{2, 4, 6\}$. Therefore, node 2's uncovered 2-hop neighbor set is $N_2(2) - N(5) - N(\{4, 6\}) = \{3\}$. Using the FNSSP-EDC, node 2 selects node 3 as its forwarding node.

3.3 The Double-Covered Broadcast Algorithm

The double-covered broadcast (DCB) algorithm uses the following symbols:

- $F(v)$: the forwarding node set of node v.
- $U(v)$: the uncovered 2-hop neighbor set of node v.
- $X(v)$: the selectable 1-hop neighbor set of node v.
- $P(v, F(v))$: a unique broadcast packet P forwarded by node v that attaches v's forwarding node set $F(v)$.
- T_{wait}: the predefined duration of a timer for a node to overhear the retransmission of its forwarding nodes.
- R: the maximum number of retries for a node.

The DCB algorithm (Algorithm 5) works as follows:

1. When a node s starts a broadcast process, s uses the FNSSP-DC algorithm to select its forwarding node set $F(s)$ and broadcasts the packet P together with $F(s)$.
2. When a node v receives P from an upstream sender u, it records P. v also updates its $X(v) = X(v) - N(u)$ and $U(v) = U(v) - N(u) - N(F(u) - \{v\})$. Note that $X(v)$ and $U(v)$ are initialized to $H(v)$ and $H_2(v)$. Then, v checks whether it is a designated forwarding node of u. If not, v drops the packet and stops the process; otherwise, v further checks whether P is ever received. If P is a new packet for v, v uses the FNSSP-EDC algorithm to compute its forwarding nodes $F(v)$ and sends P with $F(v)$. If v has already received P from another node, v will not forward P, but send an ACK to u to confirm the reception so that u will not retransmit the same packet at a later time.
3. When the sender u broadcasts P, it waits for a predefined duration T_{wait} to overhear the retransmission of its forwarding nodes. If u overhears a retransmission packet from its forwarding node v, u regards this as an ACK from v. u may receive explicit ACKs from some of its forwarding nodes to confirm the reception. If u does not overhear all of its forwarding nodes when the timer expires, it assumes that the transmission failure has occurred for this packet. u then determines a new $F(u)$ to cover the rest of the uncovered $U(u)$ and resends the packet until the maximal number of retries R is reached. The algorithms that determine the $F(u)$, such as Resend or Reselect, will be discussed in the next subsection.

Algorithm 5 The Double-Covered Broadcast Algorithm (DCB)

1. When source s wants to broadcast P, it uses the FNSSP-DC to find $F(s)$ and broadcasts $P(s, F(s))$.

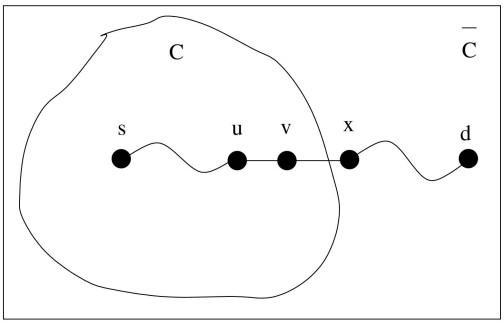


Fig. 4. An illustration of the proof for Theorem 2.

2. When node v receives $P(u, F(u))$ from u,
   2.1. v records $P(u, F(u))$.
   2.2. v updates $X(v) = X(v) - N(u)$ and $U(v) = U(v) - N(u) - N(F(u) - \{v\})$.
   2.3. **if** $v \in F(u)$ **then**
      if the packet has not been received before **then**
        v uses the FNSSP-EDC to find $F(v)$ that covers $U(v)$ and broadcasts $P(v, F(v))$.
      else
        v sends an ACK to u to confirm the reception of P and drops the packet.
      end if
    else
      v drops the packet.
    end if
3. When node u has sent the packet, it starts a timer T_{wait} and overhears the channel. After T_{wait} is expired, if u does not overhear all nodes in $F(u)$ to resend P or to send ACKs, u retransmits P until the maximal number of retries R is reached.

In Algorithm 5, a forwarding node will forward the packet if it receives the packet for the first time. The correctness of Algorithm 5 is guaranteed by Theorem 2.

Theorem 2. *Given a connected network, the DCB algorithm works correctly based on the assumption that broadcasting through this network is an atomic operation.*

Proof. We prove Theorem 2 by contradiction. Assume that the network is not fully covered when broadcasting a packet with the DCB algorithm; that is, we can find at least one node d such that d does not receive the broadcast packet from the source s. In Fig. 4, the set C inside the circle represents the covered node set and $\overline{C}$ represents the uncovered node set. Therefore, $s \in C$ and $d \in \overline{C}$. Since the network is connected, there exists a path from s to d. Suppose node x is the uncovered node that is closest to s on the path and v is the predecessor of x on the path. Based on the assumption, v has received the broadcast packet, say v has received the packet from node u for the first time. Because $x \in N_2(u)$, Theorem 1 guarantees that u covers x. This contradicts the assumption. Therefore, the DCB algorithm guarantees that the network is fully covered. □

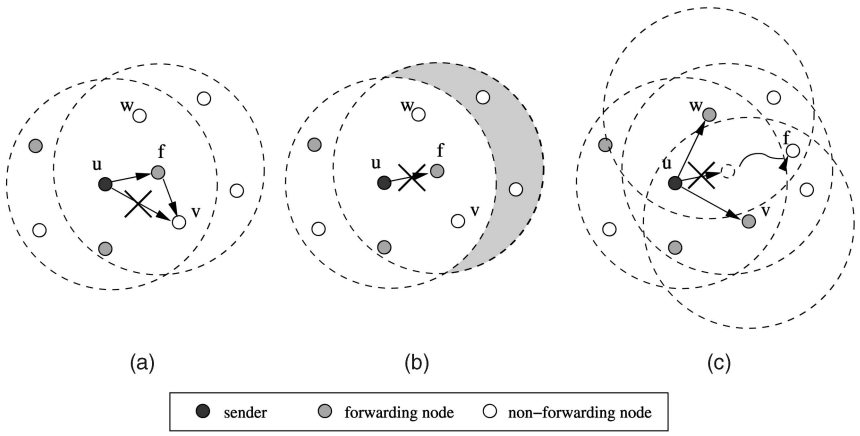


Fig. 5. An illustration of the transmission errors and solutions: (a) a transmission error occurs at a nonforwarding node v, (b) transmission error occurs at a forwarding node f that causes nodes in the shaded area to miss the packet, and (c) alternative forwarding nodes v and w are selected to cover the area that is supposed to be covered by the missing forwarding node f.

3.4 Reliability Issues

When a sender transmits a packet to all its neighbors, a neighbor may miss this packet. We treat the nonforwarding node and forwarding node differently: When a nonforwarding node v misses the packet (Fig. 5a) based on the FNSSP-EDC, v is covered by at least two forwarding nodes u and f; even when v misses the packet from u, it still has a second chance to receive the packet from f. Note that a nonforwarding node that misses the packet does not cause other transmission error propagations in the network. When the forwarding node f misses the packet, it may cause the transmission error to propagate since forwarding nodes are the key nodes in the network that need to relay the broadcast packet. There are two main causes for packet loss: the high transmission error and the out-of-range movement of the node.

High transmission error. In Fig. 5b, if f missed the transmission from u because of the transmission error, the nodes in the shaded area may also miss the packet. The simple *Resend* algorithm is adaptive to this case: u waits a period of time T_u when it sends a broadcast packet. If u fails to detect f's retransmission signal during T_u, u resends the packet until the maximum retry is reached.

Out-of-range movement of the node. A selected forwarding node may move out of the range of the sender node, and this results in a transmission failure. In Fig. 5c, f moved out of the transmission range of u and missed the packet. The *Reselect* algorithm is used for this case: When u fails to detect f's retransmission signal during T_u, u supposes f has moved out of its range and reselects alternative forwarding nodes to cover the area which is supposed to be covered by f.

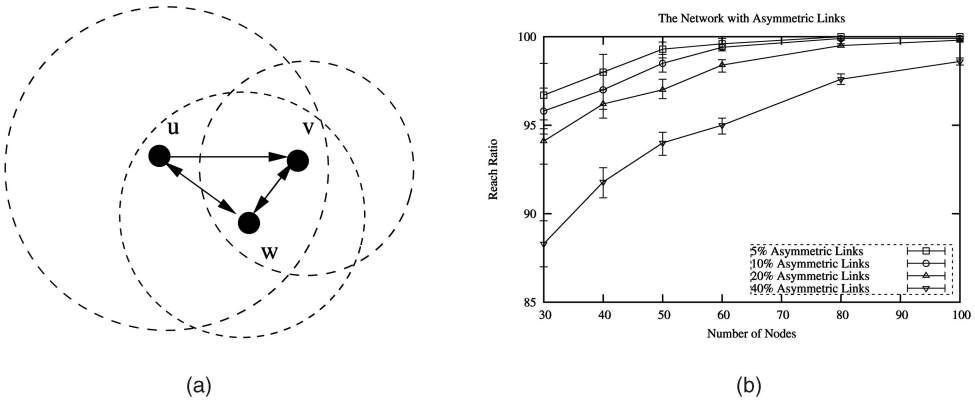
More specifically, suppose u selects a sequence of nodes that form the forwarding node set $F(u) = \{f_1, f_2, \ldots, f_m\}$ and sends the broadcast packet. u waits for T_u and does not detect the retransmission from the forwarding nodes $f'_1, f'_2, \ldots, f'_k$. The uncovered $U(u)$ is $N_2(u) - N(F(u) - \bigcup_{i=k}^{i=1}\{f'_i\})$. The selection criteria are as following: 1) Add $f_n \in N(u)$ into $F(u)$ such that f_n covers the largest number of nodes in $U(u)$. If there is a tie, the node that sent a HELLO message most recently has the highest priority. 2) Set the nodes $f'_1, f'_2, \ldots, f'_k$ to the least priority to be selected even though they may cover more nodes in $U(u)$ than other nodes. In Fig. 5c, when u does not overhear f's retransmission, u may select v and w to substitute f to cover all neighbors of f.

In the above two cases, u does not know if f is out of its range or not. If u can refresh its neighbor set on time, u can recalculate its forwarding node set on demand when it needs to resend the duplicated packet based on the FNSSP-EDC algorithm. This method, called the *Recalculate* algorithm, is suitable for the case when some new nodes move into the transmission range of u and u resends its stored packets locally. The downside of this algorithm is its long delay since each node has to wait for enough time to gather all neighbor's HELLO messages to refresh their neighbor set information.

3.5 Networks with Asymmetric Links

In the above discussion, we assume all nodes have the same transmission range r. Therefore, the generated network is always symmetric. In real ad hoc networks, asymmetric links may occur for several reasons, including different transmission ranges of nodes, local congestion, use of directional antennas, and external interference. For networks with asymmetric links, a conservative approach is that two nodes consider each other as neighbors only when they are both within the transmission range of each other. The asymmetric links are ignored in order to apply the ACK mechanism used only in the presence of symmetric links.

Asymmetric links can also be used to send ACKs. That is, the receiver uses a directed path with multiple nodes to send the ACK back to the sender to confirm the reception of the packet. Fig. 6a shows such a case: Due to the different transmission ranges of nodes u, v, and w, there is an asymmetric link (u, v) (from node u to v) and symmetric links between v and w and between w and u. v realizes an asymmetric link (u, v) if v receives the HELLO message from u with u's 1-hop neighbor set $N(u) = \{w\}$ and finds itself not in $N(u)$. v starts a local broadcast REQ with $\mathrm{TTL} = 2$ to find u. Intermediate node w attaches its ID and forwards the REQ. When u receives the REQ, it recognizes the asymmetric link (u, v), builds the feedback path (v, w, u),



(a) (b)

Fig. 6. (a) A sample network with asymmetric links and (b) the reach ratio of such networks.

and informs v of the feedback path. Then, v can use this path to send the ACK to u via w. Thus, a node can build a feedback path with one intermediate node. In this way, the node can "reach" its 1-hop neighbors with at most one intermediate node if such paths exist. The reach ratio can be used to indicate the degree of exchanging messages between the pair of nodes with an asymmetric link. It is defined as the ratio of number of asymmetric links that can reach each other to the total number of asymmetric links in the network.

Fig. 6b shows the reach ratio of a random network with asymmetric links with 90 percent confidence interval. The network topology is controlled such that the links of the network are composed of different percentages of asymmetric links from 5 percent to 40 percent. In Fig. 6b, we can see that the reach ratio increases when the network becomes dense. In the case that the size of the network is 100, even when the percentage of asymmetric links is as high as 40 percent of the total links, over 98 percent of the total asymmetric links have feedback paths with one intermediate node. The connectivity of the network can therefore be greatly improved.

For the DCB algorithm, the ACK message is omitted because the sender can overhear the broadcast message when the selected forwarding nodes retransmit it. In an asymmetric network, if a forwarding node v has an asymmetric link from its upstream sender u, v's retransmission can not be overheard by u, but by w. As w recognizes that the packet is coming from u to v, w explicitly sends an ACK to u.[2] Therefore, one extra ACK overhead from w to u is introduced.

3.6 Probabilistic Analysis

We study the probability of the scenario where K coverage is provided. We assume that nodes can only send messages successfully with probability P to their neighbors because the wireless channel is not an error-free channel. For a single transmission from u to v (Fig. 7a), the probability of a successful transmission is P and the probability of a failed transmission Q is $1 - P$. With the retransmission mechanism, a sender can resend the packet up to R retries if the forwarding node fails to receive the packet, where R is the maximum number of retries. Therefore, the probability that a forwarding node successfully receives the message increases. For a forwarding node f, a failed reception of the packet means that none of the sender's retransmissions succeeded. Then, the probability of a failed reception for a forwarding node f after R retries, $Q_f(R)$, is $Q_f(R) = Q^{R+1}$. The probability of a successful reception for a forwarding node f, $P_f(R)$, is given by

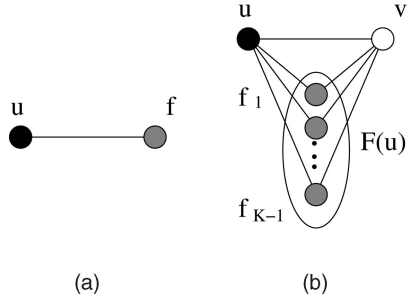
$$P_f(R) = 1 - Q_f(R) = 1 - Q^{R+1}. \quad (1)$$

A nonforwarding node v is at least adjacent to the node u and its $K - 1$ forwarding nodes $f_i \in F(u)$ (Fig. 7b). Therefore, v can receive a packet from u directly or indirectly via $f_i \in F(u)$. The probability that v successfully receives the packet after R retries, $P_v(R)$, is calculated as following:

We define random variables X and Y as follows: Random variable X is j, where $0 \leq j \leq R$, which represents that u successfully retransmits the packet to all its forwarding nodes at the jth retry; otherwise, X is -1, which represents the scenario that u fails to send the packet to all its forwarding nodes and gives up retrying because the maximum number of retries R has been reached. Y is 1 when v successfully receives the packet from u directly or indirectly; otherwise, Y is 0.

For a broadcast operation without retransmission mechanism, the probability that v successfully receives the packet, $P_v(1)$, is given by

$$P_v(1) = 1 - (1 - P^2)^{K-1}(1 - P). \quad (2)$$



(a) (b)

Fig. 7. Illustration for (a) forwarding f and (b) nonforwarding v.

2. For example, w can determine whether u is v's predecessor by checking the packet's previous route information.

Here, $1-P^2$ is the probability that v successfully receives the packet from u via a forwarding node f_i and $(1-P^2)^{K-1}(1-P)$ is the probability that v fails to receive the packet from u.

If u can retransmit the packet until all u's forwarding nodes $f_i \in F(u)$ receive the packet or u reaches the maximum number of retries R, the probability that v receives the packet increases. The probability that v successfully receives the packet after R retries, $P_v(R)$, is given by

$$P_v(R) = \sum_{j=-1}^{R} P\{X=j\}P\{Y=1|X=j\}. \tag{3}$$

$j \neq -1$ suggests that all the forwarding nodes in $F(u)$ receive the packet within j retransmissions. For each forwarding node $f_i \in F(u)$, the probability that f_i receives the packet is $1-(1-P)^{j+1}$. Therefore, the probability that all forwarding nodes in $F(u)$ receive the packet is $[1-(1-P)^{j+1}]^{K-1}$. Thus, we can get

$$\begin{cases} P\{X=0\} = P^{K-1} \\ P\{X=1\} + P\{X=0\} = [1-(1-P)^2]^{K-1} \\ P\{X=2\} + P\{X=1\} + P\{X=0\} = [1-(1-P)^3]^{K-1} \\ P\{X=j\} + P\{X=j-1\} + \ldots + P\{X=0\} \\ \quad = [1-(1-P)^{j+1}]^{K-1} \\ \cdots\cdots \\ P\{X=R-1\} + P\{X=R-2\} + \ldots + P\{X=0\} \\ \quad = [1-(1-P)^{R}]^{K-1} \\ P\{X=R\} + P\{X=R-1\} \\ \quad + P\{X=R-2\} + \ldots + P\{X=0\} \\ \quad = [1-(1-P)^{R+1}]^{K-1}. \end{cases} \tag{4}$$

Combining (4) and $\sum_{j=-1}^{R} P\{X=j\} = 1$, we get

$$P\{X=j\} = \begin{cases} (1-Q^{j+1})^{K-1} - (1-Q^{j})^{K-1}, & 0 \le j \le R \\ 1-(1-Q^{R+1})^{K-1}, & j=-1, \end{cases} \tag{5}$$

where $Q = 1-P$.

Similarly, when $j \neq -1$, $P\{Y=1|X=j\} = 1-Q^{K+j}$, which represents the scenario that v has $K+j$ chances to receive a packet from u, including $j+1$ chances directly from u and $K-1$ chances indirectly from u via f_i. $j=-1$ suggests that not all forwarding nodes in $F(u)$ receive the packet. Suppose that there are r forwarding nodes receiving the packet from u, where $0 \le r \le K-2$. The probability that v receives the packet is $1-Q^{R+1+r}$ because v has $R+1+r$ chances to receive a packet from u, including $R+1$ chances directly from u and r chances indirectly from u via f_i. Moreover, given $j=-1$, the probability that r forwarding nodes receive the packet from u is

$$\frac{\binom{\mathrm{K-1}}{\mathrm{r}}(1-Q^{R+1})^r Q^{(R+1)(K-1-r)}}{P(X=-1)}. \tag{6}$$

Therefore, we can get

$$P\{Y=1|X=-1\} = \frac{\sum_{r=0}^{K-2}\binom{\mathrm{K-1}}{\mathrm{r}}(1-Q^{R+1})^r Q^{(R+1)(K-1-r)}(1-Q^{R+1+r})}{1-(1-Q^{R+1})^{K-1}}.$$

Therefore,

$$P\{Y=1|X=j\} = \begin{cases} 1-Q^{K+j}, & 0 \le j \le R \\ \frac{\sum_{r=0}^{K-2}\binom{\mathrm{K-1}}{\mathrm{r}}(1-Q^{R+1})^r Q^{(R+1)(K-1-r)}(1-Q^{R+1+r})}{1-(1-Q^{R+1})^{K-1}}, & j=-1. \end{cases} \tag{7}$$

Replacing expressions (5) and (7) into (8), we get

$$P_v(R) = \sum_{j=0}^{R}[(1-Q^{j+1})^{K-1} - (1-Q^{j})^{K-1}]\cdot(1-Q^{K+j}) + \sum_{r=0}^{K-2}\binom{\mathrm{K-1}}{\mathrm{r}}(1-Q^{R+1})^r \cdot Q^{(R+1)(K-1-r)} \cdot (1-Q^{R+1+r}). \tag{8}$$

For the DCB algorithm where $K=2$, we can get the probability that v successfully receives the packet after R retries from (8) as

$$\begin{aligned} P_v(R) &= \sum_{j=0}^{R}[(1-Q^{j+1}) - (1-Q^{j})] \\ &\quad \times (1-Q^{2+j}) + Q^{R+1}(1-Q^{R+1}) \\ &= \frac{(1+Q-Q^2)(1-Q^{2R+2})}{1+Q}. \end{aligned} \tag{9}$$

We now calculate the probability that a forwarding node correctly forwards a broadcast packet. A propagation error may happen when a node that is selected as a forwarding node does not receive the broadcast packet because of transmission error. Among the 1-hop neighbor set of f, suppose there are m forwarded nodes (black nodes) that select f as a forwarding node and n forwarded nodes (gray nodes) that select f as a nonforwarding node. The probability that f correctly forwards a packet is equal to the probability that the first attempted transmission to f is from a black node and the first successful transmission to f is also from a black node.

Random variable U is 1 when a black node first sends a packet to f; otherwise, U is 0. Similarly, V is 1 when f first receives the packet from a black node; otherwise, V is 0.

Without the retransmission mechanism, the probability of an attempted transmission that comes from a black neighbor under the condition that it has m black neighbors and n gray neighbors, $P_{\langle m,n\rangle}\{U=1\}$, is $\frac{m}{m+n}$. Similarly, the probability that an attempted transmission comes from a gray neighbor under the condition that it has m black neighbors and n gray neighbors, $P_{\langle m,n\rangle}\{U=0\}$, is $\frac{n}{m+n}$.

The probability that a successfully received transmission first comes from a black neighbor under the condition that it has m black neighbors and n gray neighbors, $P_{\langle m,n\rangle}\{V=1\}$, is given by

$$\begin{aligned} P_{\langle m,n\rangle} &= P_{\langle m,n\rangle}\{U=1\}P_{\langle m,n\rangle}\{V=1|U=1\} \\ &= P_{\langle m,n\rangle}\{U=1\}[P+(1-P)P_{\langle m-1,n\rangle}] \\ &= \frac{m}{m+n}[P+(1-P)P_{\langle m-1,n\rangle}], \end{aligned} \tag{10}$$

where $P_{\langle 0,n\rangle}\{V=1\} = 0$, $\forall n > 0$.

TABLE 1
Simulation Scenario: (a) Parameters and (b) Algorithms

Parameter	Value
Simulator	ns-2 (version 2.26)
Network Area	$900 \times 900\ m^2$
Transmission Range	$250\ m$
MAC Layer	IEEE 802.11
Data Packet Size	64 bytes
Bandwidth	2 M b/s
Simulation Time	$100\ s$
Number of Trials	10
Confidence Interval	90%

(a)

Algorithm	Description		
	Transmit	Acknowledge	Retransmit
DCB-SD	forwarding nodes	forwarding nodes	Resend
DCB-ST	forwarding nodes	forwarding nodes	Reselect
DCB-RE	forwarding nodes	forwarding nodes	Recalculate
AHBP-EX	forwarding nodes	none	none
BF	all nodes	none	none
RB	all nodes	all nodes	flooding

(b)

If each forwarding node has the retransmission mechanism and suppose each forwarding node can retry up to R retries, which is equal to the case that there are $(R+1)m$ black nodes and $(R+1)n$ gray nodes in f's neighborhood, then the probability that f may first receive a packet from a black node with R retries, $P^{(R)}_{\langle m,n\rangle}$, is given by

$$P^{(R)}_{\langle m,n\rangle} = P_{\langle (R+1)m,(R+1)n\rangle}. \tag{11}$$

4 Simulations

4.1 Simulation Description

4.1.1 Simulation Environment

In order to analyze the performance of the proposed algorithm, we ran the simulation under the $ns-2$ testbed with a CMU wireless extension. The simulator parameters are listed in Table 1a: The network area is confined within $900 \times 900\ m^2$. Each node in the network has a constant transmission range of 250 m. We use a *two-ray ground reflection model* as the radio propagation model. The MAC layer scheme follows the IEEE 802.11 MAC specification. We use the broadcast mode with no RTS/CTS/ACK mechanisms for all message transmissions, including HELLO, DATA, and ACK messages. Since transmission errors may occur when nodes send messages in real wireless channels, we assume a probability P for each wireless channel to successfully transmit a message. The movement pattern of each node follows the *random way-point model*: Each node moves to a randomly selected destination with a constant speed between 0 and the maximum speed V_{max}. When it reaches the destination, it stays there for a random period T_s and starts moving to a new destination. The pause time T_s is always 0 in our simulation. The network traffic load also affects the performance of the protocol; we change the value of constant-packet-rate CPR (packet per second) while each packet has a constant length of 64 bytes. A node may fail to receive a message because of a transmission error, a transmission collision, or the node's out-of-range movement. After sending a message, a node will wait for a period of time, T_{wait}, and resend the message until it reaches the maximum value R. Each simulation was run for 100 seconds. In order to avoid the initialization bias of the system state on the broadcast operation, we first make all nodes move around within the area for 1,000 seconds so that they can thoroughly exchange HELLO messages to build up 1-hop and 2-hop neighbor sets. Then, some randomly selected nodes start to send broadcast packets. This procedure lasts for 100 seconds. To make sure all the broadcast packets propagate throughout the network, the simulation will last for another 10 seconds after the last broadcast process has been sent. We run the simulation 10 times to achieve a 90 percent confidence interval for the results.

4.1.2 Simulation Algorithm and Metrics

We compare the performances of the broadcast algorithms through simulations to see the benefits and losses of the double-covered broadcast algorithm. Table 1b lists the double-covered broadcast (DCB) algorithm with three different retransmission schemes described in Section 3.4, Resend, Reselect, and Recalculate, which are respectively referred to as DCB-SD, DCB-ST, and DCB-RE. Three other broadcast algorithms described in Section 2, blind flooding (BF), extended ad hoc broadcast protocol (AHBP-EX) [12], and reliable broadcast (RB) [23], are also included for comparison.

We measure the following metrics:

1. *Broadcast delivery ratio*. Broadcast delivery ratio is the ratio of the number of the nodes that received packets to the number of the nodes in the network for one broadcast operation.
2. *Broadcast forwarding ratio*. Broadcast forwarding ratio is the fraction of the total number of the nodes in the network that at least retransmit broadcast packets once for one broadcast operation.
3. *Broadcast overhead*. Broadcast overhead indicates the normalized transmissions of the broadcast operation per node. It defines the ratio of the total transmissions, including the broadcast packets and extra control packets such as HELLO and ACK messages, to the broadcast packets per node. It is measured by bytes per broadcast byte per node.

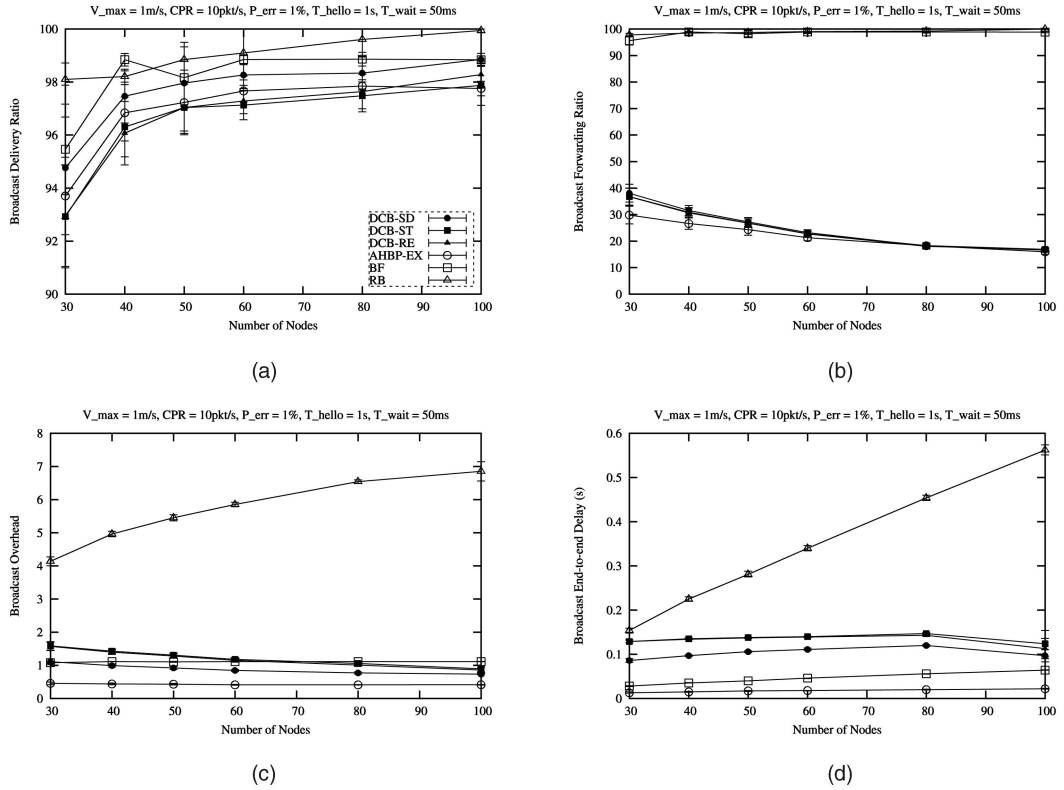


Fig. 8. Sensitivity to size of the network: (a) delivery ratio, (b) forwarding ratio, (c) overhead, and (d) end-to-end delay.

4. *Broadcast end-to-end delay*. Broadcast end-to-end delay measures the period from the time the source broadcasts the packet to the time the last node receives the packet or no more nodes resend the packet for one broadcast operation.

4.1.3 Affected Parameters

We consider the following parameters that affect the performance of the broadcast:

1. *Network size* (n). The number of nodes in a network determines the density of the network. A dense network will cause more collision and contention.
2. *Transmission error rate* (P_{err}). The physical radio channel is affected by many environmental parameters. Therefore, the Signal-to-Noise Ratio (SNR) at the receiver may be below the threshold even though the receiver is in the transmission range of the sender. This affect can be estimated as transmission error rate P_{err}, which specifies a simple transmission error model in which messages may have been lost in the physical wireless channel.
3. *Mobility of the node* (V_{max}). The mobility of the node affects the performance of the broadcast operation. The faster the node moves, the higher is the possibility of the node to lose the broadcast packet.
4. *Interval of HELLO messages* (T_{HELLO}). Since the nodes get neighbor information through HELLO messages, the hello interval determines the accuracy of one node's neighbor set. A large value of the interval will cause the information of the neighbor set to be out-of-date, which misleads the forwarding node's broadcast decision. But, increasing the frequency of the interval also increases the overhead and causes network congestion because sending HELLO messages too frequently is similar to a flooding operation.
5. *The number of retries* (R). It is intuitive that increasing the number of retries can improve the broadcast delivery ratio but also increases the end-to-end delay. On the other hand, if R is set to 0, the algorithm can benefit only from the double coverage mechanism but not from the message retransmission mechanism. By default, we set R to 1.

4.2 Results and Analysis

4.2.1 Sensitivity to Network Size

Fig. 8 shows the scenario that the network has low mobility, where V_{max} is 1 meter per second (m/s), and low transmission error rate $(P_{err} = 1\%)$. The data traffic load CPR is 10 packets per second (pkt/s), the hello interval T_{HELLO} is 1 second (s), and the waiting time T_{wait} is 50 milliseconds (ms). We identify the effect of network size n to each metric. The network under this environment can be considered a static error-free network. Most of the packet losses come from transmission collisions.

Fig. 8a shows the broadcast delivery ratio. We can see that, under such an environment, all algorithms have good delivery ratios ($> 90\%$). The delivery ratio of DCB-SD is higher than the other two algorithms (DCB-ST, DCB-RE) for all the ranges. The delivery ratios of all DCB algorithms are slightly higher than AHBP-EX when the network is dense $(n = 100)$, which suggests that the broadcast delivery ratio benefits from the retransmission mechanism. The DCBs

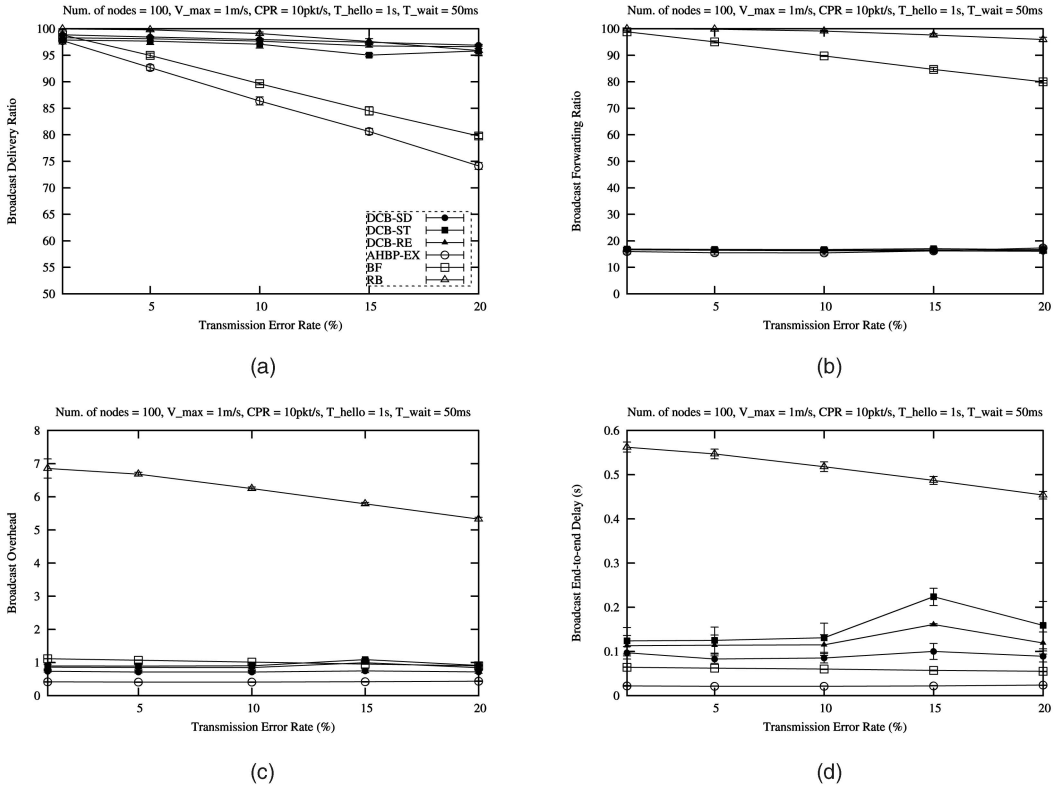


Fig. 9. Sensitivity to transmission error rate of the network: (a) delivery ratio, (b) forwarding ratio, (c) overhead, and (d) end-to-end delay.

have a little bit lower delivery ratio than BF and they are comparable when n becomes 100. The RB has the best delivery ratio. Notice that, even under such a very low mobility and very low transmission error rate environment, BF cannot guarantee 100 percent coverage. When the size of the network is small $(n = 30)$, the network may sometimes disconnect, which leads to a delivery ratio lower than that in a large network. Fig. 8b shows the broadcast forwarding ratio. Both BF and RB have almost every node forwarding while all DCBs and AHBP-EX have less than 40 percent of total nodes forwarding a broadcast packet. The AHBP-EX has the fewest forwarding nodes but the gap between AHBP-EX and DCBs becomes slight as n increases. Fig. 8c shows the broadcast overhead. The overhead of the AHBP-EX has the smallest value which is only half of the BF. The overhead of the DCBs is less than 1 when the network becomes denser $(n > 80)$; that is, the transmission overhead can be saved for a broadcast operation under this scenario when compared with the BF algorithm. RB shows the highest overhead of all the algorithms since each node that receives a packet needs to send back an ACK message. Fig. 8d shows the broadcast end-to-end delay. The AHBP-EX, BF, and DCBs have similar short end-to-end delays that are not sensitive to the size of the network, while RB has a much longer end-to-end delay that is proportional to the size of the network.

From this simulation, we can see that all algorithms have high delivery ratios under the scenario that the network is almost static and transmission error-free. DCBs and AHBP-EX have comparable performance under this scenario, and they have a much smaller forwarding ratio and overhead when compared with BF. Although RB has the highest delivery ratio, its other metrics are much worse than other algorithms. Also, we notice that DCB-SD performs best among three DCB algorithms.

4.2.2 Sensitivity to Transmission Error Rate

Fig. 9 shows the performance of the algorithms under different transmission error rates. In this case, $n = 100$, $V_{max} = 1\,m/s$, $CPR = 10\,pkt/s$, $T_{HELLO} = 1s$, and $T_{wait} = 50\,ms$. We change the transmission error rate P_{err} from 1 percent to 20 percent to see its effect on each metric.

In Fig. 9a, we see that the delivery ratio is affected by P_{err}. When P_{err} increases, the delivery ratio drops for all algorithms. But, the DCBs are much better than AHBP-EX and BF when P_{err} increases. The RB has a similar delivery ratio to DCBs even when P_{err} is high, but the forwarding ratio of RB is much higher than DCBs and AHBP-EX, as is the overhead (Figs. 9b and 9c). The end-to-end delay of DCB is longer than AHBP-EX and BF (Fig. 9d) due to the retransmission mechanism. As we can see, RB has the largest value for forwarding ratio, overhead, and end-to-end delay.

From this simulation, we conclude that DCBs outperform AHBP-EX and BF when P_{err} becomes high. This is due to the retransmission mechanism of DCB. Compared with RB, DCB uses much less broadcast overhead to provide a comparable delivery ratio while RB needs the high forwarding ratio, large overhead, and long end-to-end delay to reach a high delivery ratio.

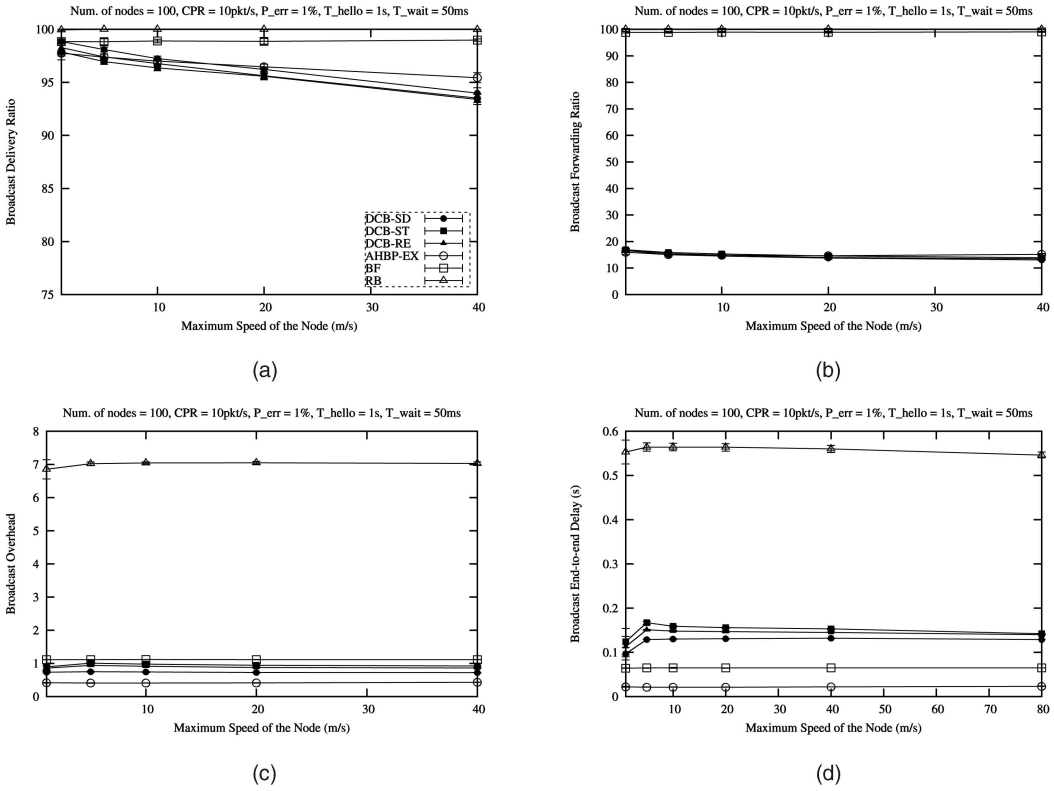


Fig. 10. Sensitivity to mobility of the node: (a) delivery ratio, (b) forwarding ratio, (c) overhead, and (d) end-to-end delay.

4.2.3 Sensitivity to Mobility of the Node

Fig. 10 shows the effect of the node's mobility on the performance of broadcast operation. In this case, $n = 100$, $CPR = 10\ pkt/s$, $P_{err} = 1\%$, $T_{HELLO} = 1\ s$, and $T_{wait} = 50\ ms$. We change the maximum speed of each node V_{max} from 1 to $40\ m/s$ to show the effect of the node's mobility to each metric.

Fig. 10a shows the broadcast delivery ratio of each algorithm. The delivery ratios of BF and RB are almost 100 percent while those of DCBs and AHBP-EX drop as the node's mobility increases. DCBs are even a little worse than AHBP-EX when the node's mobility increases. Fig. 10b shows the broadcast forwarding ratio. DCBs and AHBP-EX have almost the same forwarding ratio and their value decreases as the node mobility increases. The value of forwarding ratio for the BF and RB is always close to 100 percent. Figs. 10c and 10d show the broadcast overhead and end-to-end delay. Mobility affects these metrics only slightly.

4.2.4 Sensitivity to Hello Interval

In order to investigate the effect of the hello interval on the performance of the DCB algorithm, we set the hello interval T_{HELLO} at 0.2, 1, and 5 s. Here, we use the DCB-SD algorithm; other DCB algorithms have similar results. In this case, $n = 100$, $P_{err} = 1\%$, $CPR = 10\ pkt/s$, and $T_{wait} = 50\ ms$. V_{max} ranges from 1 to 40 m/s.

In Fig. 11a, the delivery ratio decreases as the maximum speed of the node increases, especially when T_{HELLO} is 5 s. This is because, as the node's mobility increases, the neighbor set information each node maintains becomes stale more quickly. The short hello interval causes the neighbor information to be kept more accurately in the dynamic network environment. Simulation results show that updating too infrequently causes the neighbor information to be inaccurate (Figs. 11a and 11b) while updating the HELLO messages too frequently generates large overhead (Figs. 11c and 11d). We notice that updating the HELLO message too frequently ($T_{HELLO} = 0.2\ s$) does not provide a higher delivery ratio than $T_{HELLO} = 1\ s$. This is because the very short hello interval causes the timer for each node's 2-hop neighbor set information to expire quickly and then the corresponding neighboring nodes will be easily removed from the neighbor set. Therefore, fewer nodes will be selected as forwarding nodes and the delivery ratio will drop. From these figures, a proper value for the hello interval should be close to 1 s.

4.2.5 Sensitivity to Number of Retries

We test the performance of the DCB under different values of R. In this case, $n = 100$, $V_{max} = 1\ m/s$, $CPR = 10\ pkt/s$, $T_{HELLO} = 1\ s$, and $T_{wait} = 50\ ms$. RT_{max} is set from 0 to 3. The transmission error rate P_{err} is changed from 1 to 20. Fig. 12 shows the effect of the number of retries on the performance of the DCB-SD algorithm. Fig. 12a shows that the delivery ratio can be improved when a retransmission mechanism is applied (comparing between the curves of "DCB-SD, retry=0" and "DCB-SD, retry=1"). On the contrary, increasing the number of retries (comparing between the curves of "DCB-SD, retry=1" and "DCB-SD, retry=2") only slightly improves the delivery ratio, or can even decrease the delivery ratio (comparing between the curves of "DCB-SD, retry=1" and "DCB-SD, retry=3"), but

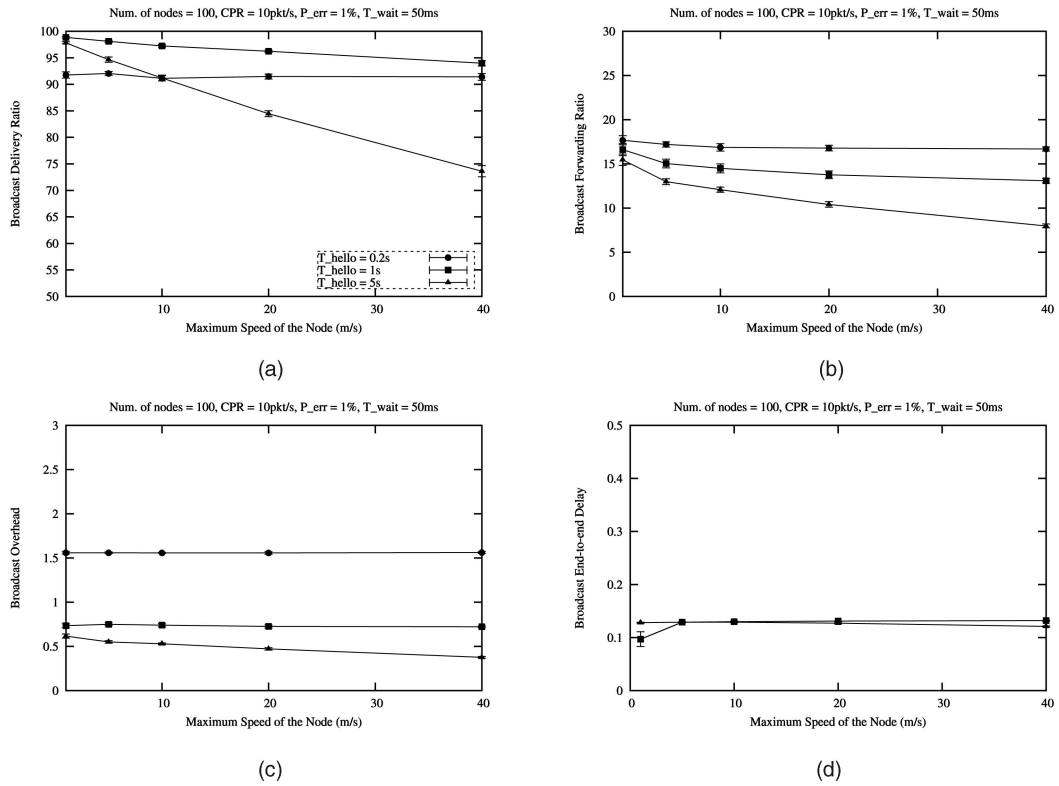


Fig. 11. Sensitivity to hello interval: (a) delivery ratio, (b) forwarding ratio, (c) overhead, and (d) end-to-end delay.

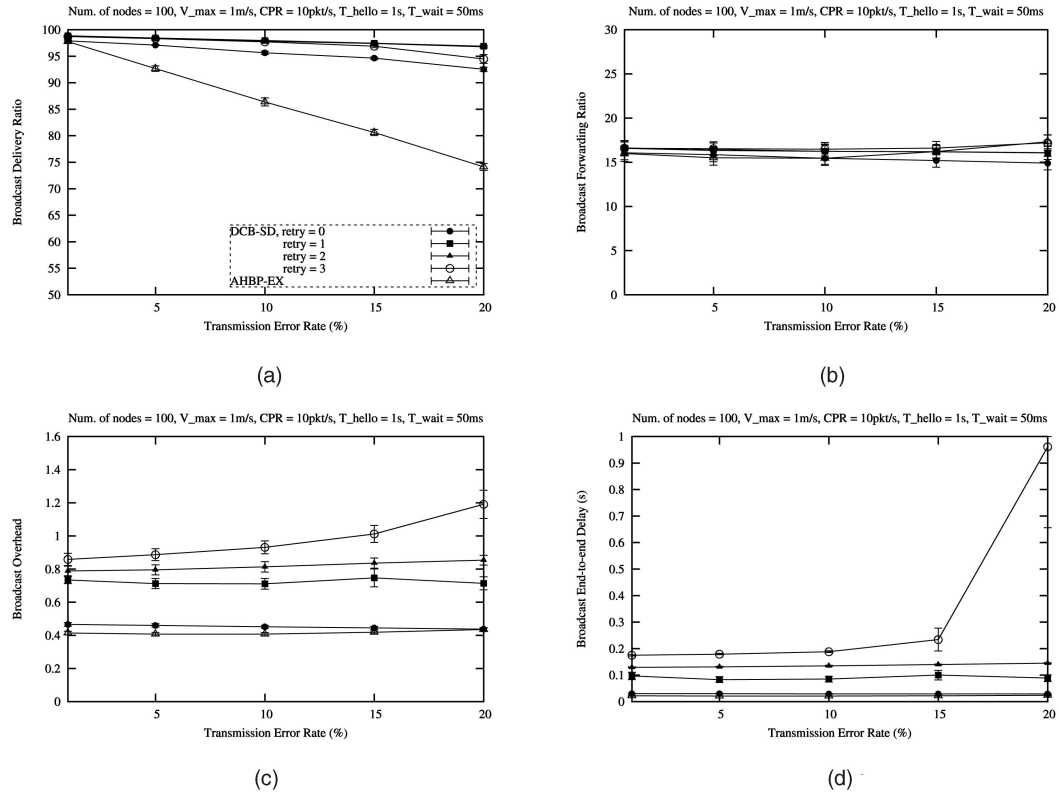


Fig. 12. Sensitivity to number of retries: (a) delivery ratio, (b) forwarding ratio, (c) overhead, and (d) end-to-end delay.

results in increasing forwarding ratio, broadcast overhead, and end-to-end delay (Figs. 12b, 12c, and 12d). Therefore, the value for the number of retries can be set to 1.

Compared to AHBP-EX, DCB has two mechanisms that improve the delivery ratio. That is, the sender retransmits when not receiving the implicit ACK from its chosen

forwarding nodes and selects additional forwarding nodes for the double coverage. Because of these, DCB outperforms AHBP-EX in the delivery ratio, while incurring more overhead. In Fig. 12a, DCB without the retransmission mechanism (the curve "DCB-SD, retry=0") has a much higher deliver ratio than AHBP-EX. Moreover, Fig. 12c shows that the overhead is almost the same for these two curves. This indicates that the double coverage is very effective for improving the delivery ratio when the transmission error rate is high. On the other hand, the contribution of the retransmission mechanism to the delivery ratio is less than that of the double coverage mechanism. This is indicated as the extra delivery ratio improvement and the extra overhead between the curve "DCB-SD, retry=0" and the curve "DCB-SD, retry=1" in Figs. 12a and 12c, respectively.

4.2.6 Summary

From the simulation, we obtain the following observations:

1. In the scenario that the network is almost static and transmission error-free, all algorithms have a high delivery ratio regardless of how the size of the network changes. The overhead and the end-to-end delay of the RB are much worse than other algorithms.
2. When the transmission error rate is high, the DCB improves the delivery ratio while keeping the overhead and end-to-end delay low.
3. The DCB is sensitive to the node's mobility, while flooding-based algorithms (RB, BF) are insensitive to the node's mobility (at the cost of high broadcast overhead).
4. A short hello interval can remarkably improve the performance of the DCB algorithm with respect to the higher overhead cost, but frequently sending HELLO messages adversely affects the delivery ratio.
5. Both double coverage and retransmission mechanisms can increase the delivery ratio when the transmission error rate is high, but increasing the number of retries only slightly improves or even decreases the delivery ratio.

5 Conclusions and Future Work

In this paper, we propose a simple broadcast algorithm that provides a high delivery ratio while suppressing broadcast redundancy. This is achieved by only requiring some selected forwarding nodes among the sender's 1-hop neighbor set to forward the packet. The double-covered forwarding node set selection process provides some redundancy to increase the delivery ratio for nonforwarding nodes so that retransmissions can be remarkably suppressed when transmission errors are considered. The simulation results show that the double-covered broadcast algorithm has high delivery ratio, low forwarding ratio, low overhead, and low end-to-end delay for a broadcast operation under a high transmission error ratio environment. From the simulation, we observe that the DCB is sensitive to the node's mobility. When the node's mobility increases, the delivery ratio of the DCB drops significantly. The reason for this is that the high mobility of nodes makes node neighbor sets outdated quickly. This incorrect neighbor set information may lead to more nodes missing the broadcast packet. An effective method to handle this issue is to allow each node to use two transmission ranges, a small one for sending HELLO messages to find neighbors and a large one for sending broadcast data messages [36].

The DCB provides full reliability for all forwarding nodes but not for nonforwarding nodes. In order to provide full reliability for all nonforwarding nodes, we can use the NACK mechanism such that a nonforwarding node will send a NACK message when the node notices a packet loss during the continuous broadcasting transmissions. Our future work is to investigate the strategies of applying the NACK mechanism and the effects when the NACK mechanism is applied.

Acknowledgments

Wei Lou's work was supported in part by HKPU ICRG grants A-PG53 and A-PH12. Jie Wu's work was supported in part by US National Science Foundation grants ANI 0083836, EIA 0130806, CCR 0329741, CNS 0422762, CNS 0434533, and CNS 0531410.

References

[1] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network," *Proc. ACM/IEEE MobiCom '99,* pp. 151-162, Aug. 1999.

[2] M.V. Marathe, H. Breu, H.B. Hunt III, S.S. Ravi, and D.J. Rosenkrantz, "Simple Heuristics for Unit Disk Graphs," *Networks,* vol. 25, pp. 59-68, 1995.

[3] M. Impett, M.S. Corson, and V. Park, "A Receiver-Oriented Approach to Reliable Broadcast Ad Hoc Networks," *Proc. Wireless Comm. and Networking Conf. (WCNC '00),* vol. 1, pp. 117-122, Sept. 2000.

[4] W. Lou and J. Wu, "A Reliable Broadcast Algorithm with Selected Acknowledgements in Mobile Ad Hoc Networks," *Proc. GLOBECOM '03,* Dec. 2003.

[5] Z.J. Haas, J.Y. Halpern, and L. Li, "Gossip-Based Ad Hoc Routing," *Proc. INFOCOM '02,* vol. 3, pp. 1707-1716, June 2002.

[6] R. Chandra, V. Ramasubramanian, and K.P. Birman, "Anonymous Gossip: Improving Multicast Reliability in Mobile Ad-Hoc Networks," *Proc. IEEE Int'l Conf. Distributed Computer Systems (ICDCS '01),* pp. 275-283, Apr. 2001.

[7] J. Wu and F. Dai, "A Generic Distributed Broadcast Scheme in Ad Hoc Wireless Networks," *Proc. IEEE Int'l Conf. Distributed Computer Systems (ICDCS '03),* pp. 460-468, May 2003.

[8] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint Relaying for Flooding Broadcast Message in Mobile Wireless Networks," *Proc. 35th Hawaii Int'l Conf. System Sciences (HICSS-35),* pp. 3898-3907, Jan. 2002.

[9] P. Jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized Link State Routing Protocol," draft-ietf-manet-olsr-07.txt, Nov. 2002.

[10] L. Lovasz, "On the Ratio of Optimal Integral and Fractional Covers," *Discrete Math.,* vol. 13, pp. 383-390, 1975.

[11] H. Lim and C. Kim, "Flooding in Wireless Ad Hoc Networks," *Computer Comm. J.,* vol. 24, nos. 3-4, pp. 353-363, 2001.

[12] W. Peng and X. Lu, "AHBP: An Efficient Broadcast Protocol for Mobile Ad Hoc Networks," *J. Computer Science and Technology,* vol. 16, no. 2, pp. 114-125, Mar. 2001.

[13] B. Williams and T. Camp, "Comparison of Broadcasting Techniques for Mobile Ad Hoc Networks," *Proc. MobiHoc '02,* pp. 194-205, Jun. 2002.

[14] 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.

[15] W. Peng and X. Lu, "Efficient Broadcast in Mobile Ad Hoc Networks Using Connected Dominating Sets," *J. Software,* vol. 12, no. 4, pp. 529-536, 2001.
[16] P. Bhagwat, P. Misra, and S. Tripathi, "Effect of Topology on Performance of Reliable Multicast Communication," *Proc. IEEE INFOCOM '94,* pp. 602-609, June 1996.
[17] A. Erramilli and R.P. Singh, "A Reliable and Efficient Multicast Protocol for Broadband Broadcast Networks," *Proc. ACM SIGCOMM '88,* pp. 343-352, Aug. 1988.
[18] S. Floyd, V. Jacobson, S. McCanne, C.G. Liu, and L. Zhang, "A Reliable Multicast Framework for Light-Weight Sessions and Application-Level Framing," *Proc. ACM SIGCOMM '95,* pp. 342-356, Aug. 1995.
[19] J.C. Lin and S. Paul, "RMTP: A Reliable Multicast Transport Protocol," *Proc. IEEE INFOCOM '96,* pp. 1414-1424, Apr. 1996.
[20] D. Towsley, J. Kurose, and S. Pingali, "A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols," *IEEE J. Selected Areas in Comm.,* vol. 15, no. 3, pp. 398-406, Apr. 1997.
[21] S.-T. Shue, Y. Tsai, and J. Chen, "A Highly Reliable Broadcast Scheme for IEEE 802.11 Multi-Hop Ad Hoc Networks," *Proc. IEEE Int'l Conf. Comm. (ICC '02),* vol. 1, pp. 610-615, Apr.-May 2002.
[22] P.M. Merlin and A. Segall, "A Failsafe Distributed Routing Protocol," *IEEE Trans. Comm.,* vol. 27, no. 9, pp. 1280-1288, 1979.
[23] S. Alagar, S. Venkatesan, and J. Cleveland, "Reliable Broadcast in Mobile Wireless Networks," *Proc. Military Comm. Conf. (MILCOM '95),* pp. 236-240, Nov. 1995.
[24] J.J. Garcia-Luna-Aceves and Y.X. Zhang, "Reliable Broadcasting in Dynamic Network," *Proc. 1996 IEEE Int'l Conf. Comm. (ICC '96),* vol. 3, pp. 1630-1634, June 1996.
[25] 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.
[26] S. Pleisch, M. Balakrishnan, K. Birman, and R. Renesse, "MISTRAL: Efficient Flooding in Mobile Ad-Hoc Networks," *Proc. ACM MobiHoc '06,* May 2006.
[27] N. Alon, A. Bar-Noy, N. Linial, and D. Peleg, "A Lower Bound for Radio Broadcast," *J. Computer System Science,* vol. 43, pp. 290-298, Oct. 1991.
[28] I. Chlamtac and O. Weistein, "The Wave Expansion Approach to Broadcasting in Multihop Radio Networks," *IEEE Trans. Comm.,* vol. 39, pp. 426-433, Mar. 1991.
[29] I. Gaber and Y. Mansour, "Broadcast in Radio Networks," *Proc. Sixth Ann. ACM-SIAM Symp. Discrete Algorithms,* pp. 577-585, Jan. 1995.
[30] E. Kushilevitz and Y. Mansour, "An $\Omega(D\log(N/D))$ Lower Bound for Broadcast in Radio Networks," *SIAM J. Computing,* vol. 27, pp. 702-712, June 1998.
[31] F. Harary and T. Haynes, "Double Domination in Graphs," *ARS Combinatoria,* vol. 55, pp. 201-213, 2000.
[32] M. Satratzemi and K. Tsouros, "Double Domination Algorithms in Graphs," *Proc. Hellenic European Conf. Math. and Informatics (HERMIS),* vol. 55, pp. 201-213, 2000.
[33] J. Harant and M.A. Henning, "On Double Domination in Graphs," *Discussiones Math., Graph Theory,* vol. 25, pp. 29-34, 2005.
[34] H. Koubaa and E. Fleury, "On the Performance of Double Domination in Ad Hoc Networks," *Proc. IFIP Ann. Mediterranean Ad Hoc Networking Workshop (Medhoc),* June 2003.
[35] C.S. Liao and G.J. Chang, "k-Tuple Domination in Graphs," *Information Processing Letters,* vol. 87, no. 1, pp. 45-50, 2003.
[36] J. Wu and F. Dai, "Mobility Management and Its Applications in Efficient Broadcasting in Mobile Ad Hoc Networks," *Proc. IEEE INFOCOM '04,* Mar. 2004.

Wei Lou received the BE degree in electrical engineering from Tsinghua University, Beijing, China, in 1995, the ME degree in telecommunications from the Beijing University of Posts and Telecommunications, Beijing, China, in 1998, and the PhD degree in computer engineering from Florida Atlantic University, Boca Raton, in 2004. From 1998 to 1999, he worked as a software engineer at the Beijing Yinhe Computer Company. He is now an assistant professor in the Department of Computing at The Hong Kong Polytechnic University, HKSAR, China. His current research interests are in the areas of wireless ad hoc and sensor networks, mobile computing, and computer networks. He has worked on designing, analyzing, and evaluating practical algorithms with a theoretical basis, as well as building prototype systems. He is a member of the IEEE.

Jie Wu is a professor at Department of Computer Science and Engineering, Florida Atlantic University. He has published more than 300 papers in various journal and conference proceedings. His research interests are in the areas of mobile computing, routing protocols, fault-tolerant computing, and interconnection networks. Dr. Wu served as a program vice chair for the 2000 International Conference on Parallel Processing (ICPP) and as a program vice chair for the 2001 IEEE International Conference on Distributed Computing Systems (ICDCS). He is a program cochair for the IEEE First International Conference on Mobile Ad Hoc and Sensor Systems (MASS '04). He was a co-guest-editor of a special issue in *IEEE Computer* on ad hoc networks. He also editored several special issues in the *Journal of Parallel and Distributing Computing* (JPDC) and the *IEEE Transactions on Parallel and Distributed Systems* (TPDS). He is the author of the text *Distributed System Design* and is the editor of the text *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks*. Currently, Dr. Wu serves as an associate editor for the *IEEE Transactions on Parallel and Distributed Systems* and several other international journals. He is a recipient of the 1996-97 and 2001-2002 Researcher of the Year Awards at Florida Atlantic University and has 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.