A Generic Broadcast Protocol in Ad Hoc Networks Based on Self-Pruning [*]

Jie Wu and Fei Dai
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431, USA

Abstract

In ad hoc wireless networks, a successful broadcasting requires that the nodes forwarding the broadcast packet form a connected dominating set to ensure the coverage. An efficient broadcast method should generate a small forward node set without excessive overhead. Neighbor-knowledge-based methods, including neighbor-designating methods and self-pruning methods, are localized algorithms that reduce the number of forward nodes based on affordable local information. A generic broadcast protocol based on a simple self-pruning rule is proposed in this paper. The underlying local information, including the network information collected via exchanging "hello" messages and the broadcast history information carried by incoming broadcast packets, is discussed and formally defined. Most existing self-pruning protocols are shown to be special cases of the self-pruning rule. Simulation results show that more efficient protocols can be derived from the generic protocol, and high delivery ratio can be achieved with near-to-accurate local information.

1 Introduction

The problem of broadcasting in an ad hoc wireless network (or simply ad hoc network) can be viewed as a process of finding a set of *forward nodes*. The resultant forward node set should satisfy: (1) every *non-forward node* has at least one forward node as its neighbor, and (2) every forward node is connected to the source node via a path consisting of forward nodes only. In other words, the set of forward nodes, including the source node, form a *connected dominating set* (CDS) of an ad hoc network. A simple but inefficient broadcast method is blind flooding, where every node is a forward node. Blind flooding is independent of any topology information and has no maintenance overhead. But the high redundancy in packet forwarding may cause serious performance degeneration (*broadcast storm* [15]), where massive contentions and collisions occur. Another approach is to use an efficient greedy algorithm to find a *minimal connected dominating set* (MCDS) [5] based on the global information. This approach can generate a near-to-optimal forward node set, but the maintenance overhead is high in a mobile environment. Several schemes were proposed based on probability models [6, 15], where a small forward node set is determined independent of topology information or based on inexpensive local information only. However, these probability models demand fine tuning of certain parameters and cannot ensure a CDS. Most broadcast methods fall in deterministic models. Among them, *neighbor-knowledge-based methods* select forward nodes through a distributed process based on local information only. These methods ensure a CDS, produce a small forward node set, and have affordable maintenance overhead.

Neighbor-knowledge-based methods can be further divided into *neighbor-designating* methods and *self-pruning* methods. In neighbor-designating methods [7, 8, 11, 10], the forwarding status (i.e., forward/non-forward node) of each node is determined by its neighbors. In self-pruning methods [2, 4, 9, 12, 13, 17], each node determines its status based on local information. The decision processes of involved nodes are uncoordinated and purely localized; that is, the status of a node depends only on the network information in a small neighborhood and the routing history carried by incoming broadcast packets. Collectively, these independent decisions exhibit a desirable global property: the resultant forward node set is a small CDS. Although these protocols are based on similar principles, this similarity was not well recognized as these principles were presented in different frameworks. Since there is no general framework that accommodates both existing approaches and future enhancement, it is very hard to avoid repeated work in this area. For example, since each existing protocol has its own assumption on the local information collection, it is hard to compare these protocols in terms of performance (in deriving a small CDS) and overhead (in collecting local information).

[*] This work was supported in part by NSF grant CCR 9900646 and grant ANI 0073736. Email:{jie,fdai}@cse.fau.edu.

A generic broadcast protocol is proposed in this paper based on a simple *self-pruning rule*. The self-pruning rule provides a general framework that accommodates most existing self-pruning protocols. Furthermore, new protocols that are more efficient than existing ones can be derived from this framework. The underlying local information, including the neighborhood topology collected via exchanging "hello" messages and the routing history extracted from the incoming broadcast packets, are discussed and given formal definitions. Different implementation options are examined, and their performances are compared through a simulation. The simulation study shows that the generic protocol achieves good balance between performance and overhead with 2- or 3-hop neighborhood information, which happens to be the configurations of most existing protocols.

One drawback of neighbor-knowledge-based methods is their dependence on accurate local information. Simulation results in [16] show that neighbor-knowledge-based methods suffer from high delivery loss in a high mobility environment, where obsolete local information is used to determine the forward node set. Several approaches have been proposed to reduce the overhead in collecting local information with the aid of location information [12, 14]. Location information could also be used to further enhance the cost-effectiveness of the generic protocol. In this paper we assume target networks with moderate mobility, where near-to-accurate normal local information can be maintained with affordable cost. Simulation results in this paper show that the generic protocol can achieve high delivery ratio with near-to-accurate normal 2- or 3-hop information.

2 Related Work

Most neighbor-designating methods use similar heuristics to determine a set of 1-hop neighbors to cover its 2-hop neighbors. In *multipoint relaying* [11], the complete 2-hop neighbor set shall be covered. In *dominant pruning* [7], only partial 2-hop neighbor set shall be covered. Nodes that are also the 1-hop neighbors of the last visited node are not considered in the coverage requirement. This is also the case in AHBP [10]. A more efficient algorithm is proposed recently by Lou and Wu [8], where not only the 1-hop neighbors but also some of the 2-hop neighbors are excluded from the coverage area of each node. The rest of this section focuses on self-pruning methods, which can be viewed as special cases of our generic protocol.

Wu and Li's algorithm: Wu and Li [17] 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. According to pruning Rule 1, a gateway can become a non-gateway if all of its neighbors are also neighbors of another neighbor that has higher priority value. According to pruning Rule 2, a marked node can be unmarked if all of its neighbors are also neighbors of either of two other neighbors that are directly connected and have higher priority values. 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 is collected at each node. That is, each node knows which nodes are its neighbors and neighbors' neighbors.

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 non-gateway if all of its neighbors are also neighbors of any one of k other nodes that are connected and have higher priority values. Rules 1 and 2 are special cases of Rule k where k is restricted to 1 and 2, respectively. An efficient algorithm based on depth-first search was also proposed in [4] to implement a restricted version of Rule k, where the pruning of a gateway depends only on the connections among its neighbors. Simulation results show that the restricted Rule k is almost as efficient as the original one in reducing the forward node set.

Span: Chen *et al* [2] proposed the *Span* protocol 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 non-coordinator to coordinator, it waits for a backoff delay which is computed from its energy level, node degree, and the number of pairs of its neighbors that are not directly connected. The backoff delay can be viewed as a priority value, such that nodes with shorter backoff delay have higher chance of becoming coordinators. Span cannot ensure a CDS since two coordinators may simultaneously change back to non-coordinators and the remaining coordinators may not form a CDS. To conduct a fair comparison of Span and other broadcast algorithms that guarantee the coverage, we use in this paper 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 nodes with higher priority values.

LENWB: Sucec and Marsic [13] proposed the *Lightweight and Efficient Network-Wide Broadcast* (LENWB) protocol, which computes the forward node status on-the-fly. Whenever 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 priority values than v. If v's neighbor set, $N(v)$ is contained in C, node v is a non-forward node; otherwise, it is a forward node. LENWB uses 2-hop neighborhood information.

SBA: Peng and Lu [9] proposed the Scalable Broadcast Algorithm (SBA) to reduce the number of forward nodes. As in LENWB, 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 broadcast packet, node v removes $N(u)$ from $N(v)$. If $N(v)$ does not become empty after the backoff delay, node v becomes a forward node; otherwise, node v is a non-forward node. 2-hop information is used to implement SBA.

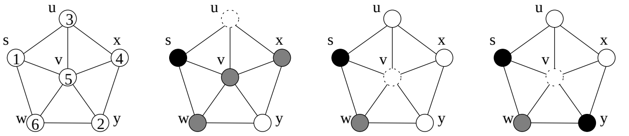
Stojmenovic's algorithm Stojmenovic et al [12] extended the Wu and Li's algorithm in two ways: (1) Suppose every node knows its accurate geographic position, only 1-hop information is needed to implement the marking process and Rules 1 and 2. That is, each node only maintains a list of its neighbors and their geographic positions (connections among neighbors can be derived). (2) The number of forward nodes are further reduced by a neighbor elimination algorithm similar to the one used in SBA.

3 Broadcasting through Self-Pruning

3.1 A generic broadcast protocol

We consider an ad hoc network as a unit disk graph $G = (V, E)$, where V is a set of nodes and E is a set of bidirectional links. For each node v, $N(v) = \{u \mid (u, v) \in E\}$ denotes its neighbor set. A broadcast process can be denoted by the set of the forward nodes $F \subseteq V$. A broadcasting is *successful* if every node receives the broadcast packets; that is, $V - F \subseteq N(F)$, where $N(F) = \bigcup_{v \in F} N(v)$. We say a broadcast protocol *ensures the coverage* if it guarantees successful broadcast, providing that G is connected, there is no topology change caused by movement during the broadcast process, and there is no packet loss in the MAC layer. It is relatively easy to ensure the coverage with a large F (e.g. blind flooding) or with a small F but based on expensive global information (e.g. MCDS). The problem is how to achieve a small forward node set based on affordable local information.

Local information collected at each node can be divided into two categories: *static information,* including neighborhood topology and a certain node attribute that serves as a priority value, and *dynamic information,* including a small set of nodes that have forwarded the broadcast packet. The static information is independent of any broadcasting and can be collected by periodically exchanging "hello" messages among neighbors. The priority values are used to establish a total order among nodes. High priority nodes usually bare more responsibilities in a broadcast process than low priority nodes. The priority value can be a topology-related attribute, such as node degree, or a topology-unrelated attribute, such as node id. The dynamic information depends on each broadcasting and is carried by the broadcast packet. More formally, for each broadcasting the local information L_v collected at a node v is a triple (G_v, p, F_v), where $G_v = (V_v, E_v)$ is a subgraph of G that usually represents the topology of a small vicinity, p is the priority function on V_v, and $F_v \subseteq F \cap V_v$ represents a list of forward nodes extracted from incoming broadcast packets. For the sake of clarity, we call nodes in F_v *black nodes*, non-black nodes with higher priorities than v *gray nodes*, and all other nodes *white nodes*. Note that the coloring of nodes is relative and depends on the view of each node. A gray node in the view of one node could be a white node in the view of another node. For example, in Figure 1, node x is a gray node in the view of node u but a white node for node v.



(a) priority values (b) u's local view (c) v's local view (d) v's local view after finding black node y

Figure 1. Local information for nodes u and v.

For any black node u in V_v, we say a neighbor w of v is *covered* by u if there exists a *replacement path* that connects u and w with gray nodes as intermediate nodes; that is, there exists a path $(u, v_1, v_2, \ldots, v_k, w)$ in G_v, where $p(v_i) > p(v)$, for $i = 1, 2, \ldots, k$. For example, in Figure 1 (b), the black node s covers nodes v and x; in Figure 1 (c), s covers nodes u, w and y, etc. In the proposed generic protocol, each node decides its own status (forward node/non-forward node) independently based on its local information. When a node receives a broadcast packet, it either forwards the packet or drops it (i.e., *pruned* from the forward node set) according to the following rule.

Self-Pruning Rule: *A node can be pruned from the forward node set if each of its neighbors is either a black node or covered by a black node.*

According to the self-pruning rule, node u in Figure 1 (b) and node v in Figure 1 (d) can be pruned. However, node v in Figure 1 (c) cannot be pruned because there is a uncovered node x. Algorithm 1 outlines the proposed generic broadcast protocol, and its correctness is shown by Theorem 1.

Theorem 1 *The self-pruning rule ensures coverage.*

Proof: We prove the theorem by contradiction. Suppose in a broadcasting the set of nodes not receiving the broadcast packet, $U = V - F - N(F)$, is not empty, let $W = N(U) - U$ be the "outer rim" of U that have received and dropped

Algorithm 1 Broadcast via Self-Pruning (for each node v)

```
Periodically exchange “hello” message with neighbors, updating the static information $G_v$ and $p(V_v)$.
On receiving a broadcast packet, build up the black node set $F_v$.
Test the self-pruning rule with local information $L_v = (G_v, p, F_v)$. If the rule applies, do nothing; otherwise, forward the received packet.
```

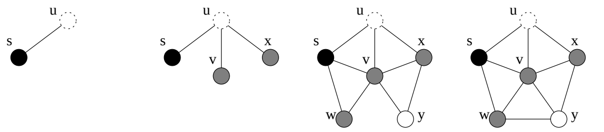
the broadcast packet. Apparently, $W \cap F = \emptyset$. Note that $W \neq \emptyset$; otherwise, a network partition exists that separates U from the source node. Let w be the node in W with the highest priority. From the assumption there is at least one neighbor $u \in U$ of w, which must be covered by a black node according to the self-pruning rule. Note that any replacement path to u must contain at least one node $w' \in W$ (i.e., must pass the outer rim), and w' must be a gray node in w's view. That contradicts the assumption that $p(w) > p(w')$. $\square$

3.2 Static Information

Algorithm 1 has many existing and potential special cases. The most significant difference among those special cases is the local information collected at each node. Usually more nodes can be safely pruned if each node has better knowledge of the global information. However, this information comes with a price. A study of options in information collection can help the understanding of existing protocols as well as the tradeoff between overhead and pruning efficiency in designing new protocols. The collection of static information, including neighborhood topology and priority values, is discussed in this subsection. Dynamic information collection is discussed in the next subsection.

Neighborhood topology G_v in each node v is collected via exchanging “hello” messages periodically among neighbors. We use the term “k-hop information” to denote the topology information that can be collected after k rounds of “hello” message exchanges. The maintenance overhead of k-hop information is related to the mobility pattern of an ad hoc network. A highly mobile network demands a smaller “hello” interval than a slightly mobile network to keep the local information up-to-date. A larger k also means a smaller interval to disseminate topology changes faster to a larger neighborhood. Therefore, it is important to keep k small in a moderately mobile network.

Figure 2 shows neighborhood topologies with different values of k. $k = 0$ means that there is no “hello” message, and each node knows only the black neighbors. The self-pruning rule cannot apply in this case, as it requires a complete list of neighbors. When $k = 1$, each node advertises its id via “hello” messages, but it will not relay its neighbors' “hello” messages. Because the links between neighbors are not known, the self-pruning rule is still difficult to apply except for the rare case that a node has all-black neighbors. Another exception is when each node also advertises its geographic location. In this case, a link between two neighbors can be determined based on their distances. Most self-pruning protocols require 2-hop information; that is, each node advertises its id and its neighbor set through “hello” messages. In this case, the replacement paths can be constructed as the links between neighbors are known. Note that the links between two nodes 2 hops away are still unknown, as shown in Figure 2 (c). Generally, k-hop information can be collected by advertising $(k-1)$-hop information in each node's “hello” messages, and the resultant topology includes nodes within k hops, links between any two nodes within $k-1$ hops, and links between a node k hops away and a node $k-1$ hops away. Using k-hop information with $k > 2$ may increase the pruning efficiency. However, this benefit needs to be justified through simulation.

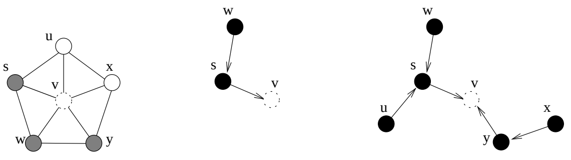


(a) 0−hop information (b) 1−hop information (c) 2−hop information (d) 3−hop information

Figure 2. Neighborhood topology of node u.

Some existing protocols, like SBA, use no gray nodes and do not need to collect priority values. To increase pruning efficiency, the priority value $p(v)$ of each node v can be advertised in the “hello” messages. We say the priority value of a node has a k-hop property if it depends on the k-hop information of this node. For example, node id has a 0-hop priority value because it is independent of any topology information. Node degree, defined as the number of neighbors, has a 1-hop priority value. Neighborhood connectivity, defined as the ratio of pairs of directly connected neighbors to pairs of any neighbors, has a 2-hop priority value.

Using the node degree as the priority value is usually more efficient than the node id in self-pruning, but it is also more expensive; that is, it requires an extra round of “hello” message exchange. Therefore, a self-pruning protocol based on 2-hop information and using node degree as priority value has similar overhead to the one using 3-hop information and node id. This drawback is avoided in the restricted Rule k, where only neighbors can be gray nodes and their priority values can still be collected in two rounds. Neighborhood connectivity is more efficient than node degree, but two extra rounds of “hello” message exchanges is expensive. In the original Span protocol, neighborhood connectivity is used to compute the backoff delay, which is



(a) no history (b) 2−hop history in 1 packet (c) 2−hop history from 2 packets

Figure 3. Broadcasting history information.

not propagated to its neighborhood as the priority value is.

3.3 Dynamic Information

The set of black nodes F_v contains the dynamic (i.e., broadcasting-specific) history information carried by incoming broadcast packets. We say a self-pruning protocol is a *static protocol* if it does not collect or use any dynamic information; otherwise, it is a *dynamic protocol*. In a static protocol, step 2 of Algorithm 1 is eliminated, and a modified version of the self-pruning rule is used in step 3:

Static Self-Pruning Rule: *A node can be pruned from the forward node set if each neighbor is covered simultaneously by all other neighbors.*

If a node satisfies the static self-pruning rule, then there is a replacement path between any pair of its neighbors. Therefore, regardless of which neighbor a node receives a broadcast packet from, its neighbors are always covered by this black node; that is, the original self-pruning rule is guaranteed. For example, node v in Figure 3 (a) can be pruned by the static self-pruning rule. The benefit of static protocols is that step 3 of Algorithm 1 can be applied prior to any broadcasting, which reduces the computation delay. A more important consideration is to form a relatively stable CDS that facilitates unicasting and multicasting as well. The drawback is that static protocols usually produce a larger CDS than the dynamic ones.

In dynamic protocols, a node is more likely to be pruned if it has a larger black node set F_v in its local information. For example, node v in Figure 1 can be pruned only after it identifies two black nodes in its neighborhood. There are two methods to increase the number of black nodes: *backoff delay* and *piggybacked history*. The backoff delay scheme postpones the testing of the self-pruning rule for a backoff delay, hoping that new black nodes can be observed forwarding the same broadcast packet. Figure 1 (c) shows the situation when the first packet comes from node s, and Figure 1 (d) shows that another copy of the same packet is received from node y. Backoff delay can also relieve the contention and collision problem in a broadcast process. The drawback is the longer overall delay.

Piggybacking broadcast history information (i.e., a list of forward nodes) in the broadcast packet is an inexpensive method to increase the number of black node. The corresponding overhead is a few extra bytes in each packet. There are three available options: 0-hop history (i.e., static protocol), 1-hop history where the id of the last visited node can be extracted from the sender field of the incoming packet and no piggybacking is necessary (Figures 1 (b) and 1 (c)), and k-hop history where id's of the last $k-1$ visited nodes are piggybacked into the broadcast packet (Figure 3 (b)). In the last option, k is the same as the one used in collecting k-hop information. Using more than k-hop history is a waste because only black nodes within k hops are useful to the self-pruning rule. Note that the piggybacked history can be combined with the backoff delay. For example, Figure 1 (d) combines backoff delay and 1-hop history information, and Figure 3 (c) combines backoff delay and 2-hop history information.

Table 1. Existing self-pruning protocols.

	Static Info			Dynamic Info	
Protocol	k	priority	round	delay	hist.
Wu-Li [a]	2	id/degree	2[b]	No	0
Dai-Wu	2	id/degree	2[b]	No	0
Span	3	connectivity	5[c]	No	0
LENWB	2	degree	3	No	1
SBA	2	-	2	Yes	1
Stojmenovic[a]	1[d]	degree	2	Yes	1

[a] Allowing at most two gray nodes in each replacement path.
[b] When restricted pruning rules are used. That is, $V_v \subseteq N(v)$
[c] When the enhanced version is used.
[d] Requiring a GPS device installed on each node.

3.4 Special cases

All self-pruning protocols described in Section 2 can be viewed as special cases of our generic self-pruning protocol, as shown in Table 1. In the first three protocols (i.e., Wu and Li's algorithm, Dai and Wu's algorithm, and Span), the static self-pruning rule applies. In the remaining three protocols (i.e., LENWB, SBA and Stojmenovic's algorithm), the dynamic (i.e., original) self-pruning rule applies. Note that none of these protocols exhausts the potential efficiency of the generic protocol. For example, the combination of 2-hop neighborhood information, node degree as the priority value, backoff delay and 2-hop history information shall be more efficient than any of these protocols.

4 Simulation

The simulation is conducted with a custom simulator *ds* [3] and focuses on three aspects: (1) the performance of the generic protocol compared with several other broadcast protocols, (2) the effects of configuration parameters on the

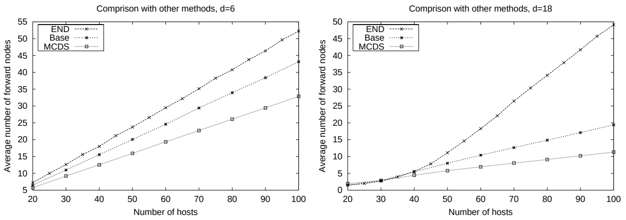


Figure 4. The effect of different methods on the number of forward nodes.

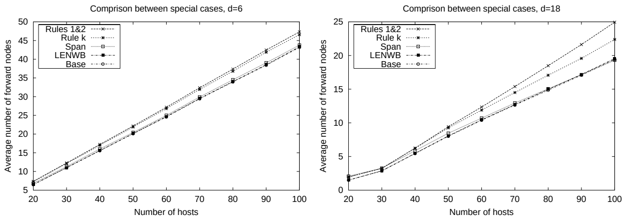


Figure 5. The effect of different special cases on the number of forward nodes.

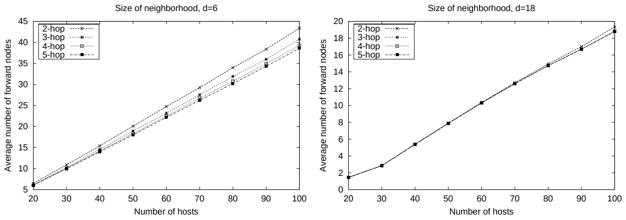


Figure 6. The effect of neighborhood radius on the number of forward nodes.

pruning efficiency and overhead of the generic protocol, and (3) the effect of node mobility on the delivery ratio of the generic protocol. Due to the limit of space, the option of backoff delay is not included, and the two corresponding protocols, SBA and Stojmenovic's algorithm, are not compared with other protocols.

Unlike ns-2, where the entire network protocol stack is considered, ds considers only functions in the network layer, assuming an ideal MAC layer without contention or collision. Simulations that cover the entire network protocol stack can be found in [16]. To generate a random ad hoc network, n hosts are randomly placed in a restricted 100×100 area. To study the behaviors of different algorithms under a given average node degree d, the transmitter range r is adjusted to produce exactly $\frac{nd}{2}$ links in the corresponding unit disk graph. Networks that cannot form a strongly connected graph are discarded. Each simulation is repeated until the confidence interval of the average results are small enough ($\pm 5\%$ at 90% confidence level).

Pruning efficiency: The efficiencies of various broadcast protocols are compared in terms of the numbers of forward nodes. We say an algorithm is more efficient than another algorithm if it generates a smaller forward node set. Figure 4 compares efficiencies of three broadcast protocols. The base protocol (Base) is a new protocol derived from the generic protocol using 2-hop neighborhood topology, node degree as the priority value, and 2-hop broadcast history. The enhanced neighbor-designating protocol (END) as described in [8] is the most efficient neighbor designating algorithm. The third is MCDS [5] based, which is based on global information and produce a near-optimal forward node set. The simple flooding method does not appear in this figure, because it always has n forward nodes. The probability-based methods are not considered, since we compare only the algorithms that ensure the coverage. In relatively sparse networks (the left graph, $d = 6$), Base is about 20% worse than MCDS and 20% better than END. These ratios remain stable as the number of nodes increases from 20 to 100. In relatively dense networks (the right graph, $d = 18$), Base is about 40% worse than MCDS and about 150% better than END. That is, the generic protocol with 2-hop approximation is closer to optimal than neighbor designating protocols, and performs much better in dense networks.

Figure 5 compares several special cases of the generic protocol, including Base, Wu and Li's algorithm (Rules 1&2), Dai and Wu's algorithm (Rule k), the enhanced Span, and LENWB. For a fair comparison, all the special cases use 2-hop neighborhood information and node degree as the priority value, except for the enhanced Span, which uses neighborhood connectivity as the priority value. Under all circumstances, Base is better than all the existing algorithms. Span and LENWB are very close to Base. Rules 1&2 are worse than Rule k, which in turn, is worse than Span, LENWB, and Base. The difference is more significant in dense networks (about 20%). This is understandable because Rules 1&2 and Rule k are based on the static self-pruning rule.

Overall, all special cases exhibit similar efficiencies, and a new protocol (i.e., Base) derived from the generic protocol is more efficient than any existing protocols.

Configuration parameters: Since different configurations have different communication and computation overheads, fine tuning of configuration parameters is necessary to achieve better balance between efficiency and overhead. Three parameters are considered: (1) k, the "radius" of the neighborhood topology G_v, (2) type of priority values, and (3) h, the size of piggybacked broadcast history. Figure 6 compares four configurations with different neighborhood radii: $k = h = 2$ (2-hop), $k = h = 3$ (3-hop), $k = h = 4$ (4-hop), and $k = h = 5$ (5-hop). All these configurations use node degree as the priority value. This is also the default setting in subsequent comparisons. In sparse networks, 2-

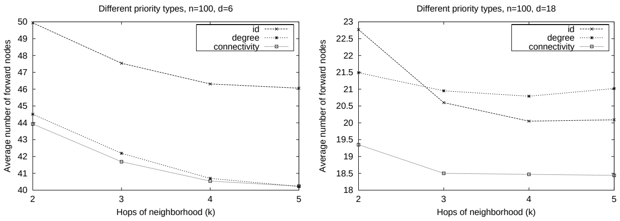


Figure 7. The effect of priority value on the number of forward nodes.

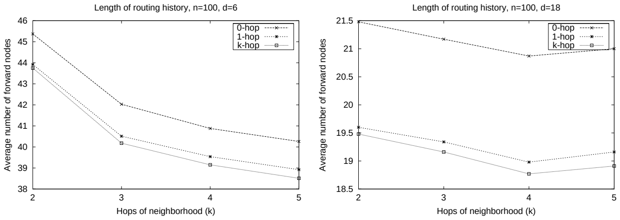


Figure 8. The effect of broadcast history on the number of forward nodes.

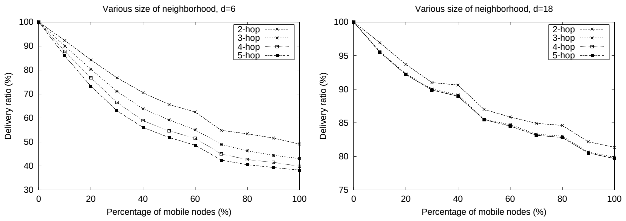


Figure 9. The effect of neighborhood radius on delivery ratio.

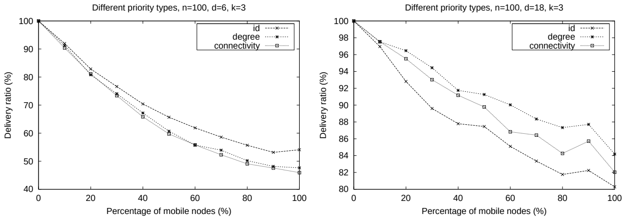


Figure 10. The effect of priority value on delivery ratio.

hop is about 10% less efficient than 3-hop, which in turn is slightly worse than 4-hop and 5-hop. In dense networks, all configurations have almost the same efficiency. We can conclude that 2-hop information is relatively cost-effective for dense networks, and 3-hop information is relatively cost-effective for sparse networks.

Figure 7 compares three configurations that use node id, node degree, and neighborhood connectivity as priority values. In sparse networks, id is the least efficient, and degree and connectivity are very close. This simulation is conducted on networks with 100 nodes ($n = 100$), with k varying from 2 to 5. In dense networks, id and degree have similar efficiencies. Degree is better with small k, and id is better with large k. Connectivity is the most efficient priority under all circumstances. There is no optimal choice of the priority type. Node id is the best for minimizing the converging time. Neighborhood connectivity is the best for relatively stationary networks. Node degree is more desirable when the computation power of each node is limited and longer converging time is tolerable.

Figure 8 compares three configurations that piggyback 0-hop, 1-hop and k-hop broadcast history information in broadcast packets. In sparse networks, 0-hop is about 5% less efficient than 1-hop and k-hop. In dense networks, 0-hop is about 10% less efficient than 1-hop and k-hop. Under both circumstances, 1-hop and k-hop have similar efficiency. Therefore, using 1-hop routing history is more cost-effective.

Overall, a cost-effective configuration shall be based on 2- or 3-hop information and 1-hop routing history.

Network mobility: Although self-pruning protocols are not designed for highly mobile networks, it is still important to know how much mobility is "too high". Furthermore, a close study can reveal the impacts of different configuration parameters on the overall mobility resilience. We measure the mobility resilience through delivery ratio, which is defined as the ratio of the number of nodes that receive the broadcast packet to the total number of nodes in the network. The mobility model of the simulation is as follows: Suppose each node updates its k-hop neighborhood information for every fixed time period Δt. During each Δt, a given percentage ($p\%$) of nodes may leave their original positions and cause inaccuracy in neighborhood information. The movement pattern is similar to the random waypoint model [1], with a maximum speed r/Δ, where r is the wireless transmitter range, and no pause time.

Figure 9 compares the delivery ratio of the generic protocols with different neighborhood radii k. The network mobility varies from no mobility ($p = 0\%$) to full mobility ($p = 100\%$). In sparse networks, even slow mobility ($p = 10\%$) may cause serious loss of delivery ratio ($< 90\%$). In dense networks, 20% mobility is still tolerable because of the higher topological redundancy. Using 2-hop information achieves higher delivery ratio than other approximations, but the difference is quite small. Figure 10 compares different priority types. In sparse networks, id as the priority value performs best, and degree and connectivity are very close. In dense networks, degree is the best, then connectivity, and id fares the worst. Figure 11 compares different lengths of piggybacked broadcast history. Here we use 3-hop neighborhood information and node degree as the priority value. In sparse networks, the length of routing his-

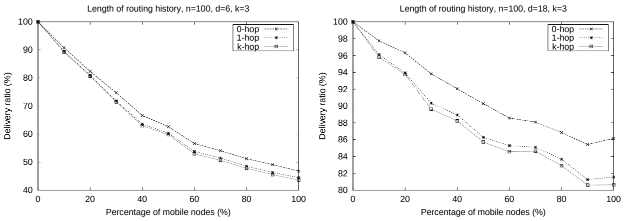


Figure 11. The effect of broadcast history on delivery ratio.

tory makes no difference. In dense networks, using 0-hop routing history is much better than the other two options. It achieves a high delivery ratios (90%) under a relatively high mobility (50%).

Overall, the coverage conditions are very sensitive to network mobility, which may cause heavy packet loss even when 10% of neighborhood information is inaccurate. However, the situation is much better in relatively dense networks. Using 2-hop neighborhood information, 0-hop routing history, and node degree as the priority value can also improve the delivery ratio.

5 Conclusion

We have proposed a general framework for broadcasting in ad hoc networks that uses self-pruning techniques to reduce the number of forward nodes. The proposed generic protocol is the superset of several existing self-pruning protocols. The general framework provides a uniform platform, upon which existing protocols can be compared and more efficient protocols be developed. A comprehensive simulation study reveals that:

1. There is a special case of the generic protocol that is more efficient than most existing neighbor-knowledge-based protocols.

2. A good balance between efficiency and overhead can be achieved by using 2- or 3-hop neighborhood information and 1-hop broadcast history information. Selection among node id, node degree and neighborhood connectivity as priority value shall be a tradeoff between efficiency and stability.

3. High delivery ratio can be achieved under moderate node mobility, especially in relatively dense networks with 2-hop neighborhood information and 0-hop broadcast history information.

Our future work includes enhancement of the general framework to interpret other existing neighbor-knowledge-based broadcast schemes, including neighbor-designating methods.

References

[1] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva. Multi-hop wireless ad hoc network routing protocols. In *Proceedings of MobiCom*, pages 85–97, 1998.

[2] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *Proceedings of MobiCom*, pages 85–96, July 2001.

[3] F. Dai. Wireless routing simulation suit. http://sourceforge.net/projects/wrss/, 2001.

[4] F. Dai and J. Wu. Distributed dominant pruning in ad hoc wireless networks. In *Proceedings of ICC*, 2003. To appear.

[5] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, Apr. 1998.

[6] Z. J. Haas, J. Y. Halpern, and L. Li. Gossip-based ad hoc routing. *Proc. of IEEE Infocom*, June 2002.

[7] H. Lim and C. Kim. Multicast tree construction and flooding in wireless ad hoc netowrks. In *Proceedings of MSWiM*, Aug. 2000.

[8] W. Lou and J. Wu. On reducing broadcast redundancy in ad hoc wireless networks. *IEEE Transactions on Mobile Computing*, 1(2):111–123, Apr.-June 2002.

[9] W. Peng and X. Lu. On the reduction of broadcast redundancy in mobile ad hoc networks. In *Proceedings of MobiHoc*, pages 129–130, 2000.

[10] W. Peng and X. Lu. AHBP: An efficient broadcast protocol for mobile ad hoc netowrks. *Journal of Science and Technology, Beijing, China*, 2002.

[11] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying for flooding broadcast message in mobile wireless networks. In *Proceedings of HICSS-35*, Jan. 2002.

[12] I. Stojmenovic, M. Seddigh, and J. Zunic. Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 13(1):14–25, Jan. 2002.

[13] J. Sucec and I. Marsic. An efficient distributed network-wide broadcast algorithm for mobile ad hoc networks. CAIP Technical Report 248, Rutgers University, Sep. 2000.

[14] M.-T. Sun and T.-H. Lai. Computing optimal local cover set for broadcast in ad hoc networks. In *Proceedings of ICC*, pages 3291–3295, Apr. 2002.

[15] 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*, 8(2/3):153–167, Mar.-May 2002.

[16] B. Williams and T. Camp. Comparison of broadcasting techniques for mobile ad hoc networks. In *Proceedings of MobiHoc*, pages 194–205, 2002.

[17] J. Wu and H. Li. On calculating connected dominating set for efficient routing in ad hoc wireless networks. In *Proceedings of DiaLM*, pages 7–14, 1999.