Performance Analysis of Broadcast Protocols in Ad Hoc Networks Based on Self-Pruning

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

***Abstract*— Self-pruning is an effective method to reduce broadcast redundancy in ad hoc wireless networks. Unlike flooding, in a self-pruning broadcast protocol, a node may not forward a broadcast packet if a certain self-pruning condition is satisfied based on the neighborhood information. For each broadcasting, only a subset of nodes forward the broadcast packet and still guarantee the complete network delivery under the ideal network situation that no packet is lost due to packet collision and node mobility. We evaluate the performance of the family of self-pruning protocols under various network situations with $ns2$. The objective is to observe the efficiency and reliability of these protocols as a function of network density, congestion, and mobility, and provide a guideline of implementation in the "real world". Our performance analysis reveals that the protocol reliability is barely affected by packet collision. However, most self-pruning protocols suffer from low delivery ratio in highly mobile networks. We further explore various techniques that improve the delivery ratio and show that both high efficiency and reliability can be achieved in highly mobile networks.**

I. Introduction

Ad hoc wireless networks (or simply ad hoc networks) are dynamic in nature, where global information/infrastructure such as minimal spanning tree is no longer suitable to support broadcasting. *Flooding* is a simple approach to broadcasting without global information/infrastructure; in flooding, a broadcast packet is forwarded exactly once by every node. However, due to the broadcast nature of wireless communication media, redundant transmissions in blind flooding may cause the *broadcast storm problem* [1], in which redundant packets cause contention and collision.

Self-pruning is an effective method in reducing broadcast redundancy. In self-pruning-based broadcast protocols [2], [3], [4], [5], [6], [7], [8], each node collects neighborhood topology information via exchanging "Hello" messages and extracts broadcast history information from incoming broadcast packets. Each node decides its role in a specific broadcasting: it either becomes a *forward node* and forwards the broadcast packet, or becomes a non-forward node and does nothing. Collectively, forward nodes, including the source node, form a *connected dominating set* (CDS) and ensure the coverage. A set of nodes is a dominating set if every node in the network is either in the set or a neighbor of a node in the set. Recently, we proposed a general self-pruning scheme [9] that combines the strength of several existing broadcast protocols. According to this generic scheme, several existing broadcast algorithms can be viewed as special cases of a *coverage condition* with various configuration parameters. The term "coverage" comes from the fact that if a node's neighbor set is "covered" by several forward nodes, it becomes a non-forward node. A simulation study was conducted to compare the performance of several special cases of the coverage condition, and examine the effects of several implementation options. Although the coverage condition is a *localized algorithm* that is based on only local information and converges in constant rounds, it exhibits similar average efficiency (i.e., percentage of forward nodes) to several non-localized broadcast algorithms that provide constant approximation ratios. However, those protocols were simulated in networks without collision or mobility.

This work was supported in part by NSF grants CCR 9900646, CCR 0329741, ANI 0073736, and EIA 0130806. Email:{fdai,jie}@cse.fau.edu.

In this paper, we first evaluate the family of self-pruning protocols in "real" networks with collision, contention, and mobility. Simulation results show that collision is not a serious problem when a small forwarding jitter delay ($\leq 1ms$) is used, and contention is relieved with efficient self-pruning protocols. However, many efficient self-pruning protocols exhibit relatively low delivery ratio in mobile networks. One drawback of the self-pruning method is that it demands accurate neighborhood information and cannot ensure the coverage with outdated topology information. Then we discuss five optimization techniques, and show that, using appropriate techniques, high reliability (i.e., high delivery ratio) can be achieved with high efficiency in highly mobile networks.

Williams and Camp [10] have also simulated a self-pruning protocol the Scalable Broadcast Algorithm (SBA) [7]. SBA is a special case of our general self-pruning scheme. Compared with other self-pruning protocols, SBA has lower pruning efficiency (i.e., higher broadcast redundancy), causes higher end-to-end delay, and is more resilient to mobility. A variation of SBA called the neighbor coverage scheme, which is proposed and simulated by Tseng et al [11], shows similar properties in the simulation. A simulation study of the entire protocol family is still necessary in order to achieve both efficiency and reliability via the general self-pruning scheme.

II. Self-Pruning Protocols

In self-pruning protocols, each node collects the k-hop topology information via k rounds of "Hello" message exchanges with its neighbors. The "Hello" messages also propagate the *priority* of each node, which could be a permanent property (e.g., node id) or a dynamic one (e.g., node degree). During a broadcast process, each node can also extract from the incoming broadcast packets a list of *visited nodes* that have forwarded the same broadcast packet. If a self-pruning protocol does not use visited node information in its coverage condition, it is a *static protocol*; otherwise, it is a *dynamic protocol*. Static protocols maintain a relatively stable CDS for the use of both broadcasting and unicasting. Dynamic protocols are for broadcasting only, but generate a smaller CDS. In the generic self-pruning scheme [9], each node decides its own status (forwarding/non-forwarding) independently based on the following condition.

Coverage Condition: *Node v has a non-forward node status if for any two neighbors u and w, a* replacement path *exists that connects u and w via several intermediate nodes (if any) with either higher priority values than the priority value of v or with visited node status.*

It was proved in [9] that the coverage condition ensures the coverage. Seven self-pruning protocols are simulated and compared with blind flooding and a new protocol derived from the coverage condition.

Wu and Li's algorithm: Wu and Li [2] proposed a *marking process* to determine a set of forward nodes (called *gateways*) 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 v becomes a non-gateway if all of its neighbors are also neighbors of another node u that has a higher priority value; that is, v's neighbor set is *covered* by u. According to pruning Rule 2, a marked node can be unmarked if its neighbor set is covered by two other nodes 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.

Dai and Wu's algorithm: Dai and Wu [3] extended the previous algorithm by using a more general pruning rule called Rule k: a gateway becomes a non-gateway if its neighbor set is covered by 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.

Span: Chen et al [4] proposed the Span protocol to construct a set of forward nodes (called *coordinators*). A node u 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 a 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 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 priorities.

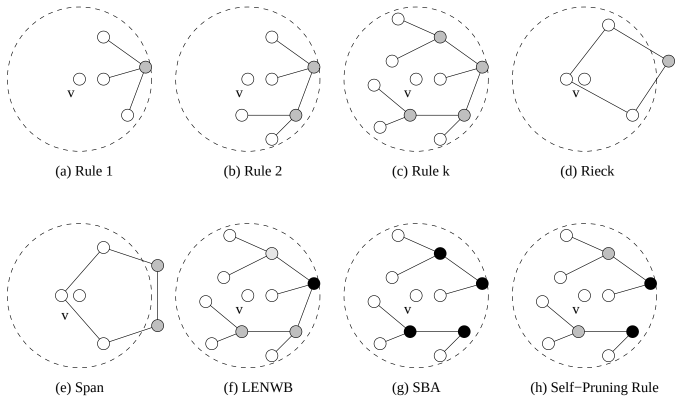


Fig. 1. Node v in the center of each subgraph can be self-pruned by the corresponding protocol. Gray nodes have higher priorities than v. Black nodes are visited nodes. Nodes in the the dashed circle are neighbors of v.

Rieck's algorithm: Rieck et al [5] recently proposed a CDS algorithm that can be viewed as a special case of the enhanced Span. In Rieck's algorithm, a node v is in the CDS if it has two neighbors that are not directly connected or indirectly connected via one intermediate node with higher priority than v. Unlike Span, the case that two neighbors are indirectly connected via two intermediate nodes with higher priorities is not considered. Therefore, the resultant CDS is larger than that in Span. On the other hand, Rieck's algorithm preserves all shortest paths in the original network.

LENWB: Sucec and Marsic [6] 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 w, it computes the set C of nodes that are connected to w 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.

SBA: Peng and Lu [7] proposed 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 w that has forwarded the broadcast packet, node v removes $N(w)$ 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.

Stojmenović's algorithm: Stojmenovic et al [8] extended 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 maintains only 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.

Figure 1 illustrates the pruning ability of different protocols. Node v can be pruned by Wu and Li's algorithm in subgraphs (a) and (b), Dai and Wu's algorithm in subgraphs (a) to (c), Span in subgraphs (a), (b), (d), and (e), Rieck's algorithm in

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Network Area	$900 \times 900\ m^2$
Transmission Range	$250\ m$
Bandwidth	$2\ Mb/s$
Data Packet Size	64 bytes
Maximal IFQ Length	50
Simulation Time	$100\ s$
Number of Trials	20
Confidence Level	95%

subgraphs (a), (b), and (d), LENWB in subgraph (f), SBA in subgraph (g), Stojmenović's algorithm in subgraphs (a), (b), and (g), and the coverage condition in all subgraphs.

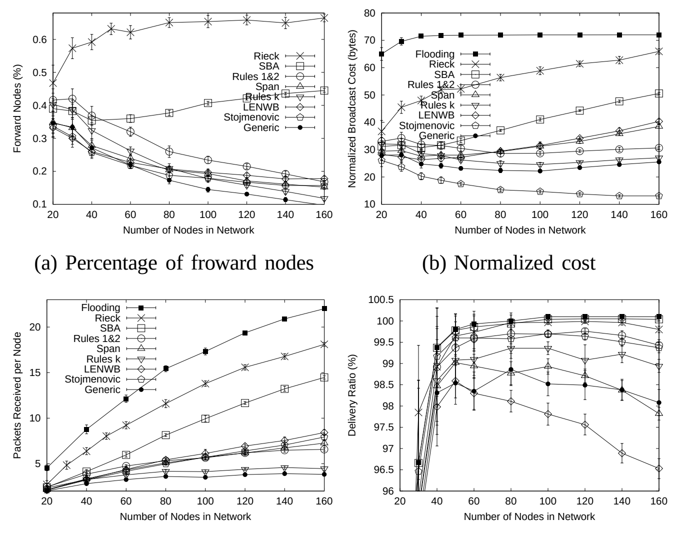
III. Performance Analysis

This section presents results from the first part of our simulation, i.e., the major threat to self-pruning protocols under congestion and mobility. Techniques that handle the threat are discussed in the next section. Unlike simulations in [9], simulations in this paper focus on reliability rather than efficiency, and are conducted on *ns-2*$(1b7a)$ [12] and its CMU wireless and mobility extension [13], using the IEEE 802.11 MAC layer, limited queue space in the link layer, random waypoint mobility model [14], and two-ray ground reflection radio propagation model with constant antenna heights. An ad hoc network is a dynamic unit disk graph under this model. A more realistic propagation model [15] may yield a higher topology changes frequency and lower broadcast reliability.

Nine protocols are simulated, including blind flooding (Flooding), Wu and Li's algorithm (Rules 1&2), Dai and Wu's algorithm (Rule k), a variation of Span that ensures the coverage (Span), Rieck's algorithm (Rieck), LENWB, SBA, Stojmenović's algorithm (Stojmenović), and a new protocol using the generic coverage condition (Generic). In order to avoid excessive packet collisions, all protocols use a random jitter between 0 and 1 millisecond before forwarding a received packet. By default, all self-pruning protocols use 1 second "Hello" interval and collect 2-hop information. The only exception is Stojmenović's algorithm, which uses 1-hop location information. Rules 1&2, Rule k and Generic use node id as priority, LENWB and Stojmenović's algorithm use node degree, and Span uses neighborhood connectivity ratio (NCR), defined as the ratio of pairs of directly connected neighbors to pairs of any neighbors. Only SBA and Stojmenović's algorithm use a random backoff delay (around $10ms$, depending on local topology). Other simulation parameters are listed in Table I.

Efficiency: Figure 2 exhibits the performance of nine protocols under low traffic load (10 broadcast packets per second) and low mobility (with average moving speed 1 m/s). Figure 2 (a) shows the efficiency of self-pruning protocols. Rieck's algorithm is the most inefficient, because its conservative pruning rule keeps all shortest paths. SBA is inefficient in dense networks, partially due to the relatively small backoff delay. Other protocols have similar efficiency; Generic is the most efficient.

Note that the cost of a broadcasting includes not only data packets sent by forward nodes, but also the overhead



(a) Percentage of froward nodes (b) Normalized cost

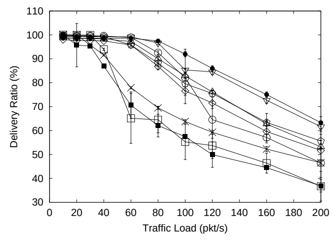
(c) Broadcast redundancy (d) Delivery ratio

Fig. 2. Performance versus network size ($1m/s$, $10pps$).

of "Hello" messages. We measure the overall broadcast cost of a protocol in terms of the normalized broadcast cost, i.e., average number of bytes sent per node per broadcasting. As shown in Figure 2 (b), Flooding has the highest cost, followed by Rieck's algorithm and SBA, which have relatively large forward node sets. LENWB and Span, which use node degree and NCR as priorities, have higher cost than the three id-based protocols, where no extra bytes are used to store neighbor priority values. Stojmenović's algorithm has the least overall cost, because it uses 1-hop location information, which has a small fixed packet size.

Tolerance to collision: We measure reliability of broadcast protocols in terms of the delivery ratio, i.e., the percentage of nodes that received the broadcast packet. Low reliability may be caused by collision, contention, or mobility. Note that packet collision is different from contention. In a collision, several nodes send packets simultaneously and packets are lost due to interference. In a contention, a node backs off when the channel is occupied, which may cause extra delay and IFQ queue overflow. Redundancy can compensate for the effects of collision and mobility, but not for those of contention. Our focus is to identify the main threat to the reliability of a protocol under various environments. Figure 2 (c) shows the broadcast redundancy, which is defined as the average number of duplicated packets received at each node. Four groups are observed: (1) Flooding, which has the highest redundancy, (2) Rieck's algorithm and SBA, which have lower redundancy than Flooding, but much higher than the others, (3) Rules 1&2, Stojmenović's algorithm, Span, and LENWB, which have moderate redundancy, and (4) Rule k and Generic, which have the lowest redundancy. Although high redundancy means low efficiency, moderate redundancy is critical for reliability.

In a low traffic and low mobility environment, the dominating effect is collision. Figure 2 (d) shows that collision is not a real threat to reliability in this case. High delivery ratio ($\geq 95\%$) is observed for all protocols. Both Rule k and Generic have very low redundancy, but both have high delivery


(a) Delivery ratio

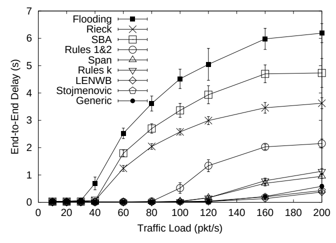
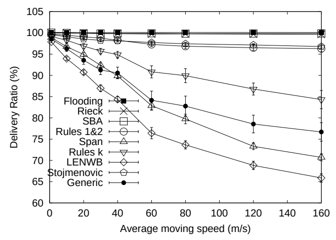

(b) End-to-end delay

Fig. 3. Performance versus traffic load (100 nodes, $1m/s$).


(a) Varying average speed

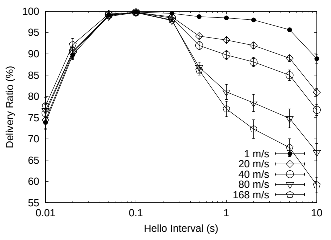
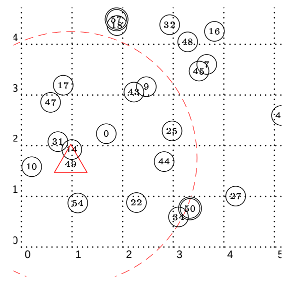

(b) Varying "Hello" interval

Fig. 4. Delivery ratio in mobile networks (100 nodes, $10pps$).

ratio ($\geq 98\%$). The only exception is in very sparse (30 nodes) networks, where network partition occurs. Five protocols with high or moderate broadcast redundancy are more reliable than other protocols; these include Flooding, Rieck's algorithm, SBA, Rules 1&2, and Stojmenović's algorithm.

Tolerance to congestion: In a congested network, low reliability is caused by either collision or contention. In the latter case, broadcast packets are dropped from the sender's queue (a queue length of 50 at each node is used in the simulation) when the network bandwidth is exhausted by the heavy traffic. Efficient protocols like Generic and Rule k are more vulnerable to collision, while protocols with high redundancy, such as Flooding, Rieck's algorithm, and SBA, suffer mainly from contention. When there is high contention, an increased average end-to-end delay will be observed. As shown in Figure 3, Flooding collapses (i.e., has lower than 90% delivery ratio) when the number of broadcast packets issued per second (pps) reaches 40, and Rieck's algorithm and SBA collapse at 60. In the mean time, obvious increases of end-to-end delay are observed. For the remaining protocols, Generic and Rule k are more tolerant to congestion than other special cases. Since there is no significant increase in their end-to-end delays, packet collision is the dominating factor.

Resilience to mobility: Figure 4 (a) shows the scenario under low traffic load ($pps = 10$) and varying mobility level. The general rule is that protocols with high broadcast redundancy (i.e., low pruning efficiency) have high delivery ratio. For example, Flooding and SBA have almost 100% delivery ratio. Rieck's algorithm and Rules 1&2 also have a very high ($\geq 95\%$) delivery ratio. The delivery ratio of Generic, the protocol with the least redundancy, drops under 90% when the average node speed is above $40m/s$. This rule has two exceptions. First, Stojmenović's algorithm maintains a high delivery ratio under a high moving speed ($160m/s$), i.e., location-based protocols are more resilient to network mobility. Second, Span


(a) A broadcast failure

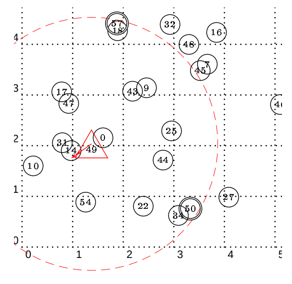

(b) 2 seconds ago

Fig. 5. A broadcast failure due to outdated neighborhood information.

and LENWB have higher redundancy but lower delivery ratio than Generic, suggesting that node degree and NCR are less resilient to mobility than node id.

Simulation results in this section can be summarized as follows:

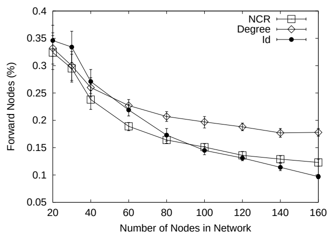
1) Generic has higher efficiency than all existing self-pruning protocols.
2) With a small forwarding jitter, all broadcast protocols achieve high delivery ratio ($\geq 96\%$) under light traffic and low mobility. That is, collision alone cannot cause serious damage on reliability.
3) Heavy contention is observed under heavy traffic, and is the major cause of the unreliability in this case, which can be relieved only with high efficiency.
4) High mobility can seriously damage the reliability of several efficient protocols, including Rule k, Span, LENWB, and Generic.
5) Using location information, Stojmenović's algorithm is both efficient and reliable under high mobility.

IV. Optimization Techniques

We examine five implementation techniques that achieve resilience to mobility in self-pruning protocols while maintaining high efficiency. The first four techniques are derived from existing protocols and have their limitations. We further propose the fifth technique to overcome those limitations.

"Hello" interval: In a mobile environment, nodes may be mistakenly pruned based on inaccurate neighborhood information. As shown in Figure 5 (a), when node 49 broadcasts a packet, none of its neighbors forward this packet. The reason is that all its neighbors have the outdated information that two high priority nodes, 50 and 57, are neighbors of node 49 and will forward the broadcast packet as in Figure 5 (b). Suppose the interval between each "Hello" message is $1s$, and a link failure is detected after missing two "Hello" messages, it takes at most $3s$ to detect a topology change in a node's 1-hop neighborhood.

One way to reduce the number of pruning mistakes is to collect more accurate neighborhood information via more frequent "Hello" messages. Figure 4 (b) shows that the delivery ratio is very low when the "Hello" interval is very large ($10s$), and becomes higher with smaller "Hello" intervals. When the "Hello" interval is around $0.1s$, very high delivery ratio (90-100%) can be achieved under very high moving speed


(a) Efficiency ($1m/s$)

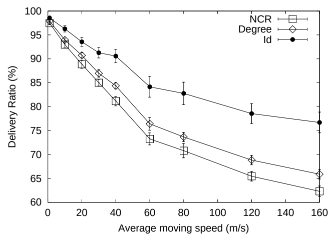
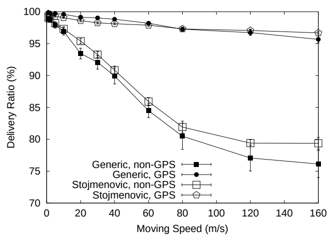

(b) Reliablity (100 nodes)

Fig. 6. Comparison of different priority types ($10pps$).


(a) Delivery ratio

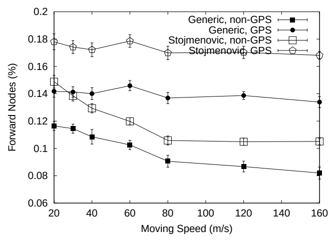
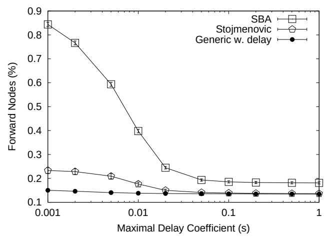
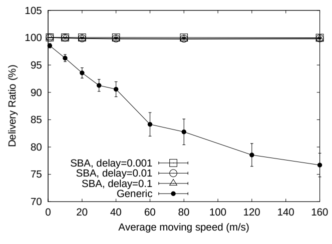

(b) Percentage of forward nodes

Fig. 7. Effects of using 1-hop location information (100 nodes, $10pps$).


(a) Efficiency versus delay ($1m/s$)


(b) Reliability versus mobility

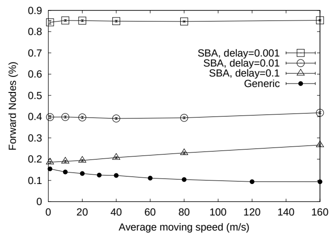

(c) Efficiency versus mobility

Fig. 8. Effects of backoff delay (100 nodes, $10pps$).

($160m/s$). On the other hand, the "Hello" interval cannot be reduced without limit. When $pps = 10$, using a "Hello" interval of $0.1s$ has higher normalized broadcast cost than blind flooding. Actually, the delivery ratio is lower with a smaller "Hello" interval ($0.01s$), as many broadcast packets are lost in collisions with "Hello" messages.

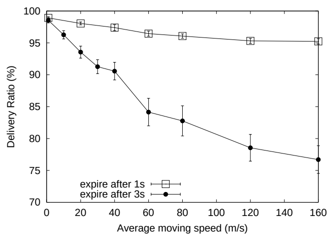
Priority type: Several previous literatures [8], [6], [9] suggested using node degree instead of node id as priority to improve the pruning efficiency, and NCR was proposed in [4] to be more efficient than node degree. This was confirmed by simulation results in [9]. On the other hand, using node id has faster convergence than node degree and NCR. In mobile networks, faster convergence means higher delivery ratio. It seems that efficiency contradicts reliability in the selection of priority types.

An interesting finding is that, although node id as priority produces more forward nodes than node degree and NCR in sparse networks (40 nodes), it is more efficient than node degree in denser networks (60-80 nodes), and more efficient than both node degree and NCR in very dense networks ($\geq$ 100 nodes), as shown in Figure 6 (a). When node id is used as priority, the forward nodes are evenly distributed to the entire area; when node degree is used as priority, most nodes are crowded in the center area and, therefore, cause higher redundancy. This is partially caused by the random waypoint mobility model we used in the simulation, where the center area tends to have higher node density than the border area. In very dense networks, only a few nodes with the highest priorities become forward nodes. If high priority nodes are evenly distributed, the entire network can be covered by fewer nodes with less redundancy; otherwise, more nodes will be forced to forward the broadcast packet. Node id also achieves higher delivery ratio than node degree, which in turn, has higher delivery ratio than NCR, as shown in Figure 6 (b).

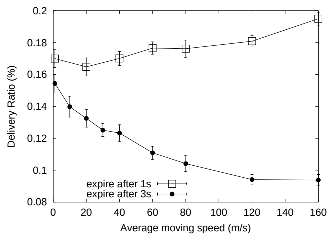
Location information: Using 1-hop location information has two benefits: lower overhead and fresher neighborhood information. If a node knows the last locations of its neighbors, it can detect link failure earlier. For example, in Figure 5, node 43 may determine that the link between nodes 49 and 57 is broken, based on location information in their last "Hello" messages, and forwards the broadcast packet. We believe the high performance of Stojmenović's algorithm is due to the use of location information, and try to apply this technique to other protocols. We simulate two variations of Stojmenović's algorithm, one with location information and the other without, and two variations of Generic as well. Figure 7 (a) shows that, when location information is available, both Generic and Stojmenović's algorithm achieve high delivery ratio ($\geq 95\%$) under the highest mobility ($160m/s$). On the other hand, when location information is unavailable, the delivery ratio of both protocols drops dramatically as the moving speed increases. It is clear that this technique also applies to the generic protocol. On the other hand, Generic is more efficient than Stojmenović's algorithm, as shown in Figure 7 (b).

Note that using location information has its drawbacks. Location information providers, such as GPS devices, usually cause higher cost and energy consumption, and the obtained location information may be inaccurate. Furthermore, the actual transmission range of mobile hosts varies in different environments, and predicting the link existence among two nodes may be unreliable, even with the accurate location information.

Backoff delay: We can also improve the efficiency of a reliable protocol while maintaining its reliability. Backoff delay is used in SBA and Stojmenović's algorithm to discover more black nodes and further reduce the number of forward nodes. We first examine the efficiency of SBA, Stojmenović's algorithm and Generic with the maximum backoff delay, D_{max}, varying from $0.001s$ to $1s$. As shown in Figure 8 (a), a relatively large D_{max} is essential to the efficiency of SBA, which performs poorly with a small backoff delay ($D_{max} = 0.001s$). Increasing D_{max} improves the efficiency of SBA significantly, until



(a) Delivery ratio



(b) Percentage of forward nodes

Fig. 9. Effects of the enhanced link failure detection (100 nodes, $10pps$).

D_{max} reaches $0.1s$. After this point, increasing D_{max} will not improve efficiency significantly. Stojmenović's algorithm is less sensitive to D_{max} than SBA, and Generic is barely affected by D_{max}.

An interesting observation is that the high reliability of SBA is not compromised by the higher efficiency achieved with larger D_{max}. As shown in Figure 8 (b), compared with Generic, variations of SBA with different values of D_{max} have almost the same high delivery ratio ($\geq 95\%$). With $D_{max} = 0.1s$, SBA is almost as efficient as Generic, as show in Figure 8 (c). In SBA, the number of forward nodes increases automatically to balance the increased mobility level. In the mean time, the number of forward nodes decreases in Generic, as more broadcast packets are lost under higher mobility. Unfortunately, this technique works only for SBA. Even with a very large backoff delay ($D_{max} = 1s$), SBA is still less efficient than both Stojmenović's algorithm and Generic with backoff delay. Another problem is the significant increase in end-to-end delay. With $D_{max} = 0.1s$, the overall end-to-end delay of SBA is about $0.2s$, while the typical end-to-end delay of Generic without backoff delay is about $0.01s$.

Link failure detection: In our implementation of self-pruning protocols, the link failure detection mechanism is designed to tolerate two consecutive loss of "Hello" messages. That is, node v considers node u as its neighbor if v has received a "Hello" message from u within the last three "Hello" intervals. This mechanism can reduce the number of false alerts caused by collisions of "Hello" messages. However, it makes the detection of a link failure very slow. On the other hand, an aggressive failure detector may cause false alerts, which leaves some neighbors uncovered and compromises the overall coverage.

Our solution is to use two expiration timers. After node v misses the first "Hello" message from node u, the failure of link (u, v) is advertised to v's neighbors. However, u is still viewed as a neighbor internally by v, until the loss of three consecutive "Hello" messages. Figure 9 (a) shows that the enhanced Generic achieves high delivery ratio. According to Figure 9 (b), the enhanced protocol has similar efficiency to the original protocol under low mobility, and the number of forward nodes increases automatically to balance higher mobility levels. In addition, this scheme does not use any location information or cause extra end-to-end delay.

We summarize our findings as follows:

1) Using a smaller "Hello" interval is more resilient to mobility. However, a very small ($< 0.1s$) "Hello" interval has a negative effect on delivery ratio.
2) Using node id as priority is more efficient and more reliable than node degree and NCR.
3) SBA achieves relatively high efficiency and high reliability with a large ($0.1s$) backoff delay.
4) Using accurate location information, self-pruning protocols are more resilient to mobility. However, location information may be inaccurate and causes extra cost.
5) Both high efficiency and high reliability can be achieved using the proposed fast link failure detection technique.

V. Conclusion

In the design of broadcast protocols, high reliability is as important as high efficiency. Low reliability (i.e., low delivery ratio) may be caused by contention, collision, or mobility. High efficiency relieves contention, but makes a protocol more vulnerable to collision and mobility. Our simulation result suggest that mobility is the major threat to efficient self-pruning protocols. We investigate several techniques that enhance the reliability of those protocols, and show that both high reliability and high efficiency in highly mobile networks. Our future work includes simulating self-pruning protocols under more realistic radio propagation models such as the "shadowing" model in *ns-2* and the real data set in [15].

References

[1] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wireless Networks*, vol. 8, no. 2/3, pp. 153–167, Mar.-May 2002.

[2] J. Wu and H. Li, "On calculating connected dominating set for effi cient routing in ad hoc wireless networks," in *Proc. of DiaLM*, 1999, pp. 7–14.

[3] F. Dai and J. Wu, "Distributed dominant pruning in ad hoc wireless networks," in *Proc. of ICC*, May 2003, p. 353.

[4] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-effi cient coordination algorithm for topology maintenance in ad hoc wireless networks," in *Proc. of MobiCom*, July 2001, pp. 85–96.

[5] M. Q. Rieck, S. Pai, and S. Dhar, "Distributed routing algorithms for wireless ad hoc networks using d-hop connected dominating sets," in *Proc. of HPC-ASIA*, Dec. 2002, pp. 443–450.

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

[7] W. Peng and X. Lu, "On the reduction of broadcast redundancy in mobile ad hoc networks," in *Proc. of MobiHoc*, 2000, pp. 129–130.

[8] I. Stojmenovi´c, M. Seddigh, and J. Zunic, "Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 1, pp. 14–25, Jan. 2002.

[9] J. Wu and F. Dai, "Broadcasting in ad hoc networks based on self-pruning," in *Proc. of INFOCOM*, Mar./Apr. 2003, pp. 2240–2250.

[10] B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks," in *Proc. of MobiHoc*, June 2002, pp. 194–205.

[11] Y.-C. Tseng, S.-Y. Ni, and E.-Y. Shih, "Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network," *IEEE Transactions on Computers*, vol. 52, no. 5, pp. 545–557, May 2003.

[12] K. Fall and K. Varadhan, "The *ns* manual," The VINT Project, http://www.isi.edu/nsnam/ns/doc/, Apr. 2002.

[13] D. B. Johnson, J. Broch, Y.-C. Hu, J. Jetcheva, and D. A. Maltz, "The CMU Monarch projects wireless and mobility extensions to ns," in *Proc. of 42nd Internet Engineering Task Force*, Aug. 1998.

[14] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483–502, 2002.

[15] D. Kotz, C. Newport, and C. Elliott, "The mistaken axioms of wireless-network research," Tech. Rep. TR2003-467, Dartmouth College Computer Science, July 2003.