Controlled Flooding in Wireless Ad-hoc Networks

A. Rahman, W. Olesinski and P. Gburzynski

University of Alberta
Department of Computing Science
Edmonton, Alberta CANADA T6G 2E8

Abstract – **We show how flooding can be adopted as a reliable and efficient routing scheme in ad-hoc wireless mobile networks. It turns out that, with the assistance of some tunable heuristics, flooding is not necessarily inferior to sophisticated point-to-point forwarding schemes, at least for some classes of wireless applications. We discuss a reactive broadcast-based ad-hoc routing protocol in which flooding exhibits a tendency to converge to a narrow strip of nodes along the shortest path between source and destination. The width of this strip can be adjusted automatically or by the user, e.g., in response to varying node density and mobility patterns. Finally, we point out a certain deficiency inherent in the IEEE 802.11 family of collision avoidance schemes and show how to fix it to provide better service to broadcast-based routing schemes represented by our variant of controlled flooding.**

Index Terms – **Ad-hoc wireless routing, flooding, mobility, collision avoidance.**

I. Introduction

Routing protocols for ad-hoc wireless networks can be broadly divided into two groups. Proactive protocols try to maintain up-to-date routing information at every node in anticipation of demand [1], [2], while reactive protocols collect the necessary routing information only when it is explicitly needed to sustain an actual session [3], [4], [5], [6], [7]. Regardless of this high-level paradigm and the manner in which the routing information is acquired, most protocols try to identify the routes before forwarding packets. In source routing schemes, this means a precise identification of every single hop to be made by the packet before the packet departs at the source node. In hop-by-hop schemes, each node knows the identity of the next node along the packet's path from source to destination.

Despite the fact that the wireless environment is inherently multicast, this free *feature* is rarely exploited during the actual forwarding of session packets, although all protocols necessarily take advantage of broadcast transmission during various stages of route discovery, when the configuration of available routes or neighboring nodes is unknown or uncertain. Also, in many protocols, e.g., DSR [6] and AODV [4], a node is allowed to overhear (and cache) routing information exchanged by other nodes, which can be viewed as a form of turning the broadcast nature of the medium to the protocol's advantage.

However, once the route has been established, the forwarding of session packets is carried out using point-to-point transmission. At first sight this is obvious: the whole idea of establishing a route is to find out which nodes should be responsible for forwarding. Once this knowledge has been acquired, it only make sense to send the packets over the established path.

It is usually assumed that the cost of information exchange during route discovery, when the knowledge of the requisite elements of network configuration is imperfect, is higher than the cost of point-to-point forwarding after that knowledge has been acquired. But formally, from the viewpoint of raw bandwidth usage of the wireless channel, it makes no difference whether a transmitted packet is addressed to (and intended for) one specific neighbor, or whether it targets all/any neighbors that can hear it.

Of course, the assumption about the poorer performance of broadcasting compared to point-to-point transmission is not unwarranted. First, the family of collision avoidance schemes for wireless channels based on IEEE 802.11 [8], [9], [10], significantly favors point-to-point transmission with respect to the reliability of channel acquisition and effective bandwidth utilization. Second, from the viewpoint of reliability of the data-link layer, a point-to-point transmission can be easily acknowledged (with a special provision for acknowledgments in the IEEE 802.11 MAC layer [10]), which is more than can be said about a broadcast transmission with no clearly defined single recipient. Third, one of the primary objectives of an ad-hoc routing protocol is to minimize resource usage. In this context, it is natural to restrict the path traveled by session packets to a well-defined (optimum) sequence of nodes. In other words, there is no reason to broadcast if the naturally preferred approach is to learn the identity of the recipient first.

In this paper, we show that the operations of route discovery and forwarding can be combined and made indistinguishable, and the result of this amalgamation, in terms of end-to-end performance, need not be inferior to other ad-hoc routing schemes, e.g., based on point-to-point forwarding. We introduce an ad-hoc routing scheme in which forwarding is inherently broadcast-based in that a transmitting node never cares about the identity of its next-hop neighbor. What only matters, is the identity of the source and destination, i.e., packets are addressed exclusively within the transport layer. The proposed scheme is a refinement of straightforward flooding assisted with several heuristics that reduce its range to a narrow stripe of nodes along the shortest path between source and destination. The width of this stripe is adjustable with static and dynamic parameters that account for the expected or perceived mobility patterns, node density, and the required quality of service (QoS).

The performance of our proposed scheme in terms of flooding confinement and convergence (i.e., global resource usage) depends on the amount of local resources (memory and processing speed) available at individual nodes. Notably, with a smaller amount of resources, the protocol still operates correctly, although it may yield suboptimal paths and slower convergence. In particular, owing to its essential simplicity, our

approach can be used in very inexpensive low-bandwidth devices (smart cards, sensors) as well as in complex and high-performance systems.

We also suggest an enhancement to the IEEE 802.11 collision avoidance scheme to improve its performance for the kind of multicast packet exchange needed in our protocol. Notably, besides increasing the reliability of forwarding, this new feature can also be used as a basis for new heuristics that facilitate path convergence and reduce global resource usage.

II. TARP: A TINY AD-HOC ROUTING PROTOCOL

The protocol presented in this section was conceived [11] as a proprietary solution of Olsonet Communications[1] and intended for small-footprint wireless devices organized into small to moderately sized ad-hoc networks, e.g., within the area of a golf course, shopping mall, or campus. The underlying assumptions were automatic configurability of nodes, maintenance-free operation, small to trivial memory requirements at a node, low power, and completely distributed operation. The protocol was implemented under PicOS [12] on an eCOG-based[2] microcontroller board.

A. An Overview

According to the simple idea of flooding, a node willing to send a packet to some destination simply broadcasts it to the neighboring nodes. A node receiving a packet checks its destination address. If the node is the intended recipient of the packet, the packet has reached the end of its path (it is received and passed to the Transport Layer). Otherwise, the node may decide to re-broadcast the packet.

To describe TARP we have to explain the meaning of the word "may" in the previous sentence. Clearly, the node should not forward blindly all received packets that happen to be addressed elsewhere. Even a most naive flooding protocol must take measures to limit the range of flooding. Among the simplest of those measures is restricting the number of hops that a single packet is allowed to travel. In addition to this obvious idea, TARP implements three more techniques (called rules) that heuristically limit the number of stray packets wandering in the network. The exact behavior of those rules is governed by a set of parameters that determine their focus (or fuzziness). For illustration, the SPD (Suboptimal Path Discard) rule acts to restrict all traffic between a pair of nodes to the proximity of the shortest path connecting them. Depending on its focus, the rule may allow some fuzziness by exploring a few (alternative) paths at the same time. While this approach uses more network resources, it provides for a better responsiveness to the dynamically changing configuration of intermediate nodes. The proper identification and implementation of this tradeoff is what ad-hoc routing is mostly about.

The rules of TARP can be viewed as a multi-part algorithm for determining whether a received packet that doesn't happen to be addressed to the current node should be retransmitted (forwarded) or dropped. The rules are executed in sequence, and the first one that says that the packet should be dropped terminates the execution of the chain. Thus, the rules are restrictive in nature: their role is to control (limit) the (otherwise unrestricted) flooding.

Once the packet has passed through all the rules, and none of them has decided that the packet should be dropped, the node will queue the packet for forwarding. We say that such a node is *eligible* to push the packet one hop forward on its way to the destination.

The concept of eligibility defines the major criterion of progress in TARP. The responsibility of a transmitting (forwarding) node is to pass the packet to at least one eligible neighbor. The node assumes that its forwarding task has been accomplished when it can tell with reasonable confidence that at least one eligible neighbor has successfully picked up the packet. For the sake of completeness of the progress criterion, we assume that the destination itself is also *eligible*, i.e., it provides the same kind of feedback to the neighbors as a forwarding node.

Packets in TARP are forwarded simply by being retransmitted. While there is no need to modify the addressing information in the headers of a forwarded packets, some header information gets updated at every hop. In addition to the obvious source/destination address pair $<S, D>$ (belonging to the transport layer), the TARP-specific (network layer) header components include: the session identifier (s) unique for a given $<S, D>$ pair, the sequence number of the packet within its session (n), the retransmission count of the packet (k), the maximum length of the path that the packet is allowed to travel expressed as the number of hops (r), the number of hops traveled by the packet so far (h_f), the total number of hops traveled by the last packet on the reverse path (h_b), the focus factor on the forward path from S to D (m_f), the focus factor on the reverse path from D to S (m_b).

The sizes of those fields may depend on the application, but most of them can be very short. For example, s and k may use 4 bits each, n, r, h_f, h_b, may each fit into 5 bits (note that the range of packet sequence number depends on the ARQ scheme used by the transport layer and shouldn't be too large in a wireless environment), and both m_f and m_b may be stored into 2-bit fields (representing one of four quantized values). This yields 32 bits of the TARP-specific components, with the packet sequence number being in fact shared with the transport layer.

The role of all those fields will be explained in the next section. The tuple $<S, D, s, n, k>$ is called the *packet signature*. It uniquely identifies a single packet within a certain time frame. The focus factor is a parameter (passed by the respective endpoint of the session) that hints at the desired aggressiveness of the rules in their effort to eliminate the redundancy of paths. Note that in contrast to the traditional approach to implementing a hop number limit, whereby the remaining hop count of a packet is decremented toward zero, TARP uses two fields: the bound set by the source remains constant, while a separate field stores the increasing number of hops traveled by the packet. This is because both values are needed by the rules.

B. The First Two Rules

The first rule, called DD for Duplicate Discard, determines whether a packet with the same signature has been recently forwarded by the node. The rule uses a cache of signatures recycled in a FIFO manner with an additional timing out of old entries – to avoid a wrap-around of n. The simple formula used

1 See http://www.olsonet.com/.

2 See http://www.cyantechnology.com/.

by TARP to determine the amount of time after which a signature should expire from the DD cache is

$$T_r = F_c \times t_{avg} \times (r - h) \quad ,$$

where t_{avg} is the average transmission time (see below), and F_c is a parameter called the flooding constant (typically between 2 and 4). Note that T_r is proportional to the packet's expected distance from the destination. The signature of a packet that is near the end of its trip is not likely to be needed for very long. Also packet duplicates are less harmful at a node located close to the destination than far away from it.

While the premature removal of a DD cache entry may affect the efficiency of TARP (some packet duplicates may pass undetected and be unnecessarily forwarded), it does not affect the formal correctness of the scheme. Thus, the amount of memory allocated to the DD cache is flexible: in some applications it can be minimal – at the cost of increased flooding and suboptimal routing performance.

TARP estimates the average transmission time t_{avg} formally defined as the interval elapsing from the moment a packet becomes queued for transmission (the node considers itself eligible to forward the packet), until the node concludes that the packet has been passed to at least one eligible neighbor. The role of t_{avg} is to estimate the expected packet processing time (the cost of making a hop through the node) in a way that accounts for dynamic conditions affecting this cost, e.g., local congestion. For calculating t_{avg}, TARP samples the processing time of individual packets passing through the node. This is accomplished with a simple trick that requires a single signature buffer, regardless of the packet queue size.[3] The calculated value is an exponential moving average of the samples, i.e., it is updated as follows:

$$t_{avg} = C_a \times t_{avg} + (1 - C_a) \times t_t \quad ,$$

where t_t is the last sample, and C_a is a constant (typically 0.65).

The second rule is called SPD for Suboptimal Path Discard. Its objective is to avoid forwarding a packet via a route that takes it too far from the shortest path between source and destination. The rule uses its own cache (the SPD cache) storing tuples $<S, D, h_{DK}, h_{SK}, C_{DS}, C_{SD}>$ indexed by unordered pairs $\{S, D\}$. Note that only one such tuple is stored for a given pair of peers involved into (possibly multiple) sessions routed through the node. Let K be the node storing the tuple. Then h_{DK} is the number of hops between D and K made by the last packet seen by K and traveling from D to S, h_{SK} is the last-seen number of hops on the path from S to K, C_{DS} and C_{SD} are the *discard counts* calculated by the rule and applicable to the two forwarding directions.

Whenever K sees a packet traveling from S to D, it updates h_{SK} and sets

$$C_{DS} = m_b \times \left[(h_{SK} + h_{DK}) - h_b \right] \quad .$$

Similarly, for a packet traveling from D to S, K updates h_{DK} and sets

$$C_{SD} = m_b \times \left[(h_{SK} + h_{DK}) - h_b \right] \quad .$$

Note that the subscript b (in m_b and h_b) is interpreted relative to the actual source of the incoming packet and it refers to the *opposite* direction. In both cases, $h_{SKD} = h_{SK} + h_{DK}$ corresponds to the current length of the path connecting S and D and passing through K, as perceived (expected) by the incoming packet. If the value of h_b in its header is less than h_{SKD}, it means that there exists (or perhaps existed a short while ago) a path shorter than the one passing through K. Thus it is highly likely that the packet will reach its destination faster via another path, and its current copy will be discarded further along its route as a duplicate. Consequently, the node may consider dropping the packet, with the confidence of this decision being proportional to the difference between h_{SKD} and h_b.

Suppose that m_b is 1 and the packet travels from S to D. Then if $C_{SD} > 0$, the route leading through K can be suspected of being superfluous and suboptimal. A careful analysis of the possible scenarios leads us to the observation that it makes sense to follow up on this suspicion and drop the packet only if both C_{SD} and C_{DS} are greater than zero. Otherwise, the neighbors of K, which may also be located on a superfluous path, will not learn about that fact and may keep forwarding other copies of the packet received from elsewhere.

The two values, C_{SD} and C_{DS} are viewed by the rule as counters. Whenever a packet traveling from S to D (the other direction is symmetric) finds both counters positive, the rule decrements C_{SD} and indicates that the node is not eligible (the packet is dropped). Thus, the higher the value of C_{SD}, the more consecutive packets trying to reach D via K will be dropped before forwarding in that direction is (tentatively) resumed by the node. By using a factor m_b, which may be less or greater than 1, the rule may be more aggressive with avoiding suboptimal routes, or more fuzzy, i.e., allow alternative routes and drop fewer packets. Note that regardless of the actual value of m_b, as long as it is greater than zero, drastically suboptimal routes are discouraged more than those that are only slightly suboptimal. Also, regardless how suboptimal the route appeared at some point, it will be tried again at some later time, with less suboptimal routes being reconsidered more often. Additionally, the frequency of those attempts depends globally on the focus factor m_b. They are critical from the viewpoint of responding to the changing routing opportunities in a dynamic network.

As implemented in TARP, the SPD rule is augmented by a simple load balancing (LB) mechanism that makes the values of the discard counters depend not only on the suspected deviation of the path length from the optimum, but also on the intermittent congestion level at the node. The rationale behind this approach is that it may be sensible to forward a packet via a longer path, if the shorter path appears to be congested. This way, the rule attempts to balance the load among the paths that are close (albeit not necessarily equal) in terms of the number of hops. With this modification, the formulas for calculating the discard counters take the shape

$$C_{SD/DS} = (m_b + d \times t_t) \times \left[(h_{SK} + h_{DK}) - h_b \right] \quad ,$$

where d is a constant coefficient dubbed the *diversity factor*, and t_t is the last sample of the packet processing time. By making the overall factor depend on t_t rather than the average packet processing time t_{avg}, the rule is intentionally "jumpy" with interpreting the load fluctuations at intermediate nodes. Depending on the setting of d, this approach tends to balance the routing among close paths and also helps in those situa-

3What we mean here is the data-link layer queue. TARP does not queue packets in the network-layer.

tions where there are multiple shortest paths between a given source and destination. Instead of using all those paths at once (which would happen with unmodified SPD), the new rule may alternatively choose only one (or some) of them.

C. Comments on Performance

Figure 1 illustrates the performance of TARP, in terms of the packet delivery fraction, and compares it to that of some other popular routing protocols. Two versions of TARP are considered: the lower of the two lines corresponds to the SPD rule without the load balancing mechanism (our intention was to demonstrate the relevance of the LB modification to SPD).

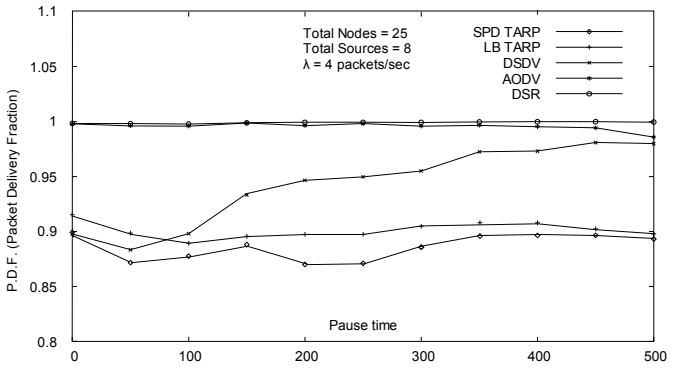

Fig. 1. Comparison of TARP with other protocols.

In our simulation experiments, we used a *random way point* mobility model parameterized by the *pause time* [13] whereby every node acts in a cyclic fashion, remaining stationary for *pause time* seconds, then selecting a random destination and moving to that destination at a speed uniformly distributed between 0 and 10 meters per second. The network model was implemented in *ns-2*[4] using wireless extensions [13].

The gap separating TARP from the other protocol becomes narrower for smaller node populations and tends to widen for larger node densities. Essentially, there are two reasons why TARP yields to the competition. First, the path convergence of TARP to the single "best" path between source and destination is not perfect, regardless of the setting of the focus factor *m*. While TARP is good at identifying shortest paths, it does not cope well with multiple shortest paths, if they happen to be present. The LB addition to the SPD rule exhibits a tendency to switch among multiple shortest paths instead of using them simultaneously, but this isn't perfect. The second problem is the inadequacy of the collision avoidance scheme in the MAC layer. As all communication is inherently broadcast, a forwarding node cannot take advantage of data-link acknowledgments to improve the reliability of communication. This is the issue that we shall address in the next section.

III. Fuzzy Acknowledgments

A forwarding node in TARP would like to know whether the packet has been picked up by any eligible node in the neighborhood. It is also OK if several nodes considering themselves eligible pick up the same packet. The identity of those nodes is of no direct importance to the forwarding node. However, with the traditional collision avoidance scheme of IEEE 802.11, the forwarding node has no means of determining or even guessing at the success of its broadcast transmission in the data-link layer. The approach used in our first implementation of TARP was to listen for a copy of the transmitted packet (forwarded by an eligible node) and use it as an indication of success – in addition to timers used to diagnose failures. There are two problems with this solution. First, depending on the load at the eligible node, there can be a significant delay between packet reception and retransmission. Second, to make this idea work, the destination itself has to "forward" (i.e., retransmit) all received packets, which creates unnecessary noise in its neighborhood.

A. Deficiencies of IEEE 802.11

The IEEE 802.11 family of MAC schemes is reasonably well equipped to handle point-to-point communication in the face of multiple parties trying to talk at about the same time [8]. With the four-way handshake, RTS/CTS/DATA/ACK, the sending node first verifies that the other party is present and willing to receive, and reserves bandwidth for the actual data exchange, and then, following the packet transmission, it receives an acknowledgment from the recipient. The RTS/CTS part of the handshake also accounts for the "hidden terminal" problem by including the recipient in the bandwidth reservation part of the complete exchange.

Unfortunately, none of these features is available for broadcast transmission, particularly in its flavor needed in TARP. First, the RTS/CTS part makes no sense because 1) the recipient's identity is unknown and unimportant, 2) there can be multiple legitimate recipients that do not know about each other. Second, even the two-way handshake, DATA/ACK, is not possible because of 2. Consequently, the only available option is to transmit blindly, following the standard DIFS delay and backoff procedure prescribed by the scheme. This completely ignores the hidden terminal problem, greatly increases the likelihood of a collision, and renders the data exchange highly unreliable.

B. The Proposed Mechanism

We propose the following simple solution as an extension of the IEEE 802.11 MAC protocol. When a node receives a packet for which it considers itself eligible, it waits for a short amount of time, defined by the short inter-frame space (SIFS), and then sends an acknowledgment. When multiple recipients send their acknowledgments at (almost) the same time, the sender will not be able to recognize them as valid packets. However, the sender can interpret any activity (of a certain bounded duration) that follows the end if its last transmitted packet as an indication that the packet has been successfully forwarded. Although the value of this indication is inferior to that of a "true" acknowledgment, it may provide the kind of feedback needed by the data-link layer to assume that its responsibility for handling the packet has been fulfilled.

Thus, having completed a packet transmission, the sender immediately switches to listening mode and awaits a period of silence (of duration comparable to SIFS) followed by a burst of activity (of duration comparable to the duration of an acknowledgment packet). If such an event occurs within the prescribed interval, the sender assumes that the packet has been passed over; otherwise, it schedules a retransmission. With this ap-

4 See http://www.isi.edu/nsnam/ns/.

proach, the acknowledgment packet (which carries no information other than its presence) can be made very short and consist of some characteristic pattern unlikely to be encountered in a regular packet.

C. Incorporation into TARP: the Third Rule

As the role of a fuzzy acknowledgment in TARP is to tell the sender that its packet has been forwarded towards the destination, it is important that only those recipients that are actually going to forward the packet (or the destination itself) send the acknowledgments. Consequently, acknowledgments cannot be sent mechanically in the data-link layer, and the incorporation of fuzzy acknowledgments into TARP requires some cross-layer coordination. For illustration, consider the configuration of nodes shown in Figure 2. Assume for simplicity that the network is static and that there is a session in progress between nodes 1 and 2, with the current converged path passing through nodes <1,10,15,20,22,2>. When node 1 sends a packet in the first hop, it will be received by nodes 6,7,10,12, and 13.

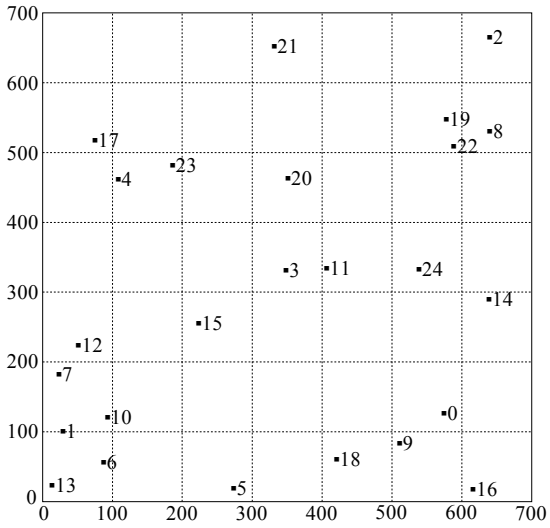

Fig. 2. A sample configuration of nodes.

In the next hop, node 10 is going to rebroadcast the packet because it lies on the converged path. But what will happen if the packet is received by some neighbors of node 1 but not by node 10, e.g., because of an interference. If any of those neighbors sends an acknowledgment that is subsequently received by node 1, then node 1 will conclude that the packet has been forwarded, which, of course, is not the case.

To see another problem, suppose that a packet sent from node 1 to node 2 has arrived at node 4, which, according to its current perception of path convergence, considers itself eligible. The recipients of the transmission of node 4 are nodes 17 and 23. If node 17 decides to forward this packet again, it will arrive back at 4 and 23, where it will be recognized as a duplicate. Thus, neither of the two nodes will find itself eligible and they will send no acknowledgments. Consequently, node 17 will keep retransmitting the packet over and over until it finally decides that the optimal path for the session lies elsewhere. This will create unnecessary activity in the neighborhood of node 17 contributing to the overall interference and reducing the amount of usable bandwidth.

By blurring the distinction between layers a bit further, we can avoid this problem and turn it into one more heuristic facilitating path convergence in TARP. Consider three nodes: the source *S*, the destination *D*, and an intermediate node *N*. Suppose that Δ refers to a time interval and define

$$RF_{SND}(\Delta) = n^{ack}_{SND}(\Delta) / n^{fwd}_{SND}(\Delta) \quad ,$$

where n^{fwd}_{SND} is the total number of packets between *S* and *D* passed through *N* within time Δ, and $n^{ack}_{SND} \le n^{fwd}_{SND}$ represents the portion of those packets for which *N* has received (fuzzy) acknowledgments. Depending on the setting of the interval Δ, *RF* (called the *relevance factor*) can be viewed as a measure of *N*'s relevance in forwarding packets between *S* and *D*. Alternatively, we can view *RF* as a measure of probability that *N* lies on the optimal path between the two end-nodes, at least as long as the configuration of nodes remains static.

With the mobility included, the indications of *RF* become less accurate. This is not solely a problem of TARP: any information related to the configuration of paths tends to become outdated, if the nodes are allowed to change their location. Thus, the proper way to interpret the value of RF should be determined experimentally. One natural idea is to use a threshold. A value of *RF* above the threshold indicates that the node is relevant in sustaining the session.

A straightforward way to implement the new rule is to take advantage of the DD cache and flag those packets for which acknowledgments have been received with one extra bit. This approach automatically equates Δ with the DD cache expiration interval and rids the protocol of one parameter.

D. The Improvement

Figure 3 illustrates how the relevance factor *RF* affects the performance of TARP in terms of the packet delivery fraction. It shows that *RF* does influence the quality of routing and hints at the range between 0.6 and 0.75 as the suggested setting. Notably, the same range of values seems to be adequate for different mobility levels, which allows us to make *RF* a constant rather than a dynamically tunable parameter.

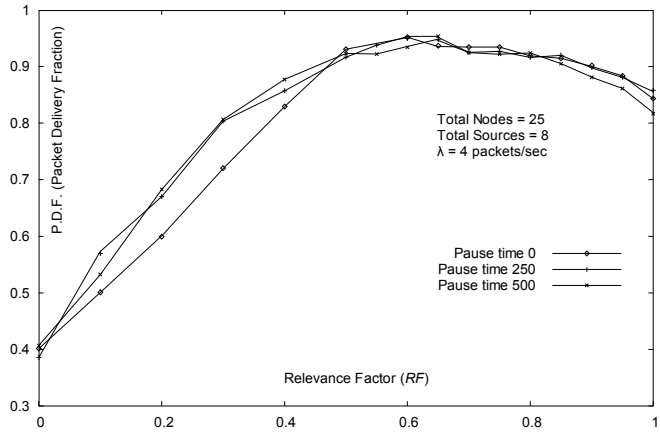

Fig. 3. Impact of RF threshold on packet delivery fraction.

In Figure 4, we show how much improvement has been brought into TARP by the addition of the new rule. The upper portion presents a magnification of the two TARP curves from Figure 1 with the inclusion of a new curve reflecting the fuzzy-acknowledgment version with *RF* = 0.7. The magnitude of the observed improvement has been consistent across different

node densities and traffic levels. The bottom portion of Figure 4 compares the three variants of TARP for a larger number of sessions.

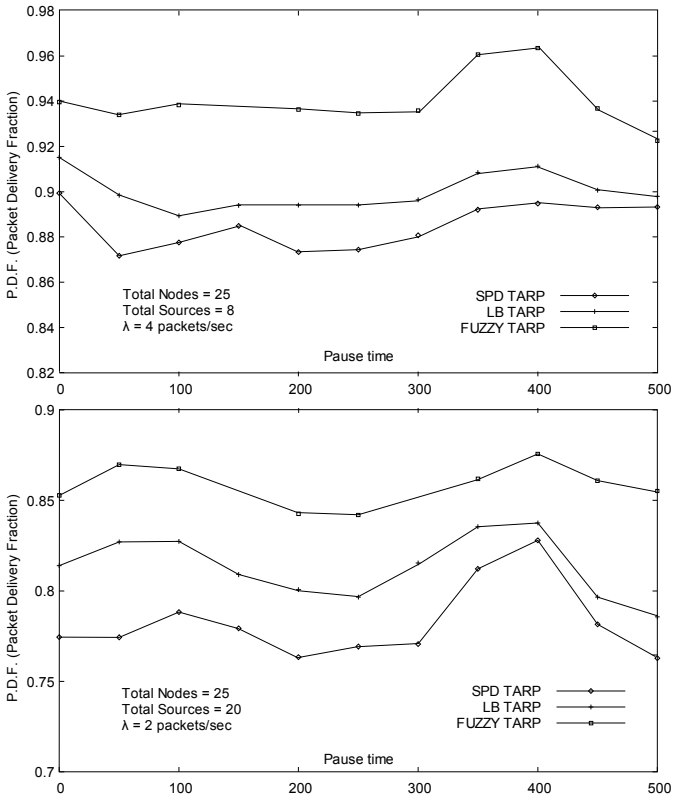

Fig. 4. The magnitude of performance improvement with the addition of fuzzy acknowledgments.

IV. Conclusions and Future Work

The simple routing protocol discussed in this paper appeals to us as a promising avenue for deploying maintenance-free ad-hoc networks based on inexpensive and small hardware. Despite its simplicity, TARP, in terms of its performance, can be compared to serious routing protocols with complex route discovery/maintenance mechanisms. With the addition of fuzzy acknowledgments that compensate for the poor handling of broadcast transmissions by the IEEE 802.11 MAC scheme, the gap separating TARP from AODV, DSDV, and DSR does not look insurmountable at all, especially that several other enhancements are in store.

Our current work on TARP is focused on three issues. First, we would like to eliminate the detrimental diversity of alternative paths with the same shortest length, which is the primary source of resource wastage in the present version of TARP. As it turns out, there is a way to provide the requisite feedback to a forwarding node without affecting the spirit of the protocol and giving up its underlying forwarding paradigm. We are currently experimenting with the *Fourth Rule* aimed at selecting one of the multiple shortest paths – the one that provides locally best service, as viewed by an eligible node.

Another promising avenue is to try a fuzzy variant of the RTS/CTS handshake, whereby the forwarding node announces its intentions with a brief RTS-like packet that contains, instead of the recipient address, the signature of the session packet about to be forwarded. In response to this packet, all eligible nodes would send a *fuzzy* CTS response. Intuitively, this solution may work better than the fuzzy acknowledgments discussed in this paper because it would also account for the hidden terminal problem.

The third issue with TARP is power management. Again, the right way to tackle it within our paradigm is to implement a rule that would turn received power indications into heuristics facilitating path identification. We are currently investigating the idea of using power level feedback for this purpose, in addition to power savings, which are extremely important in a wireless ad-hoc environment.

Although at the present stage of its development TARP may appear slightly inferior to protocols based on the point-to-point forwarding paradigm, our work at least suggests that controlled flooding may offer a viable alternative to explicit route discovery and maintenance. One can think of numerous applications where the trivially low cost of nodes, simplicity, node scalability, and the completely automatic configurability outweigh the performance penalty, which we are very likely to further reduce in the next version. Among those applications are sensor networks, smart badges, profile matchers, and all those areas in which the networking component must be implemented in a tiny, disposable, and inconspicuous device.

References

[1] Perkins, C.E. and Bhagwat, P., "Highly Dynamic Destination-Sequenced Distance Vector Routing (DSDV)," Proceedings of SIGCOMM'94, pp. 234-244, August, 1993.

[2] Chen, T-W. and Gerla, M., "Global State Routing: a New Routing Scheme for Ad-hoc Wireless Networks." Proceedings of ICC'98, June, 1998.

[3] Perkins, C.E. and Royer, E.M., "Ad-hoc On-demand Distance Vector Routing (AODV)," Proceedings of WMCSA'99, pp. 90-100, 1999.

[4] Perkins, C.E., Royers, E.M., and Das, S.R., "Ad-hoc On-demand Distance Vector Routing (AODV)," Internet Draft: draft-ietf-manet-aodv-13.txt, IETF, February, 2003.

[5] Li, J., Jannotti, J., De Couto, D. and Karger, D., and Morris, R., "A Scalable Location Service for Geographic Ad-hoc Routing," Proceedings of MOBICOM'00, pp. 120-130, 2000.

[6] Johnson, D., Maltz, D., and Hu, Y.C., "The Dynamic Source Vector Routing Protocol for Mobile Ad-hoc Networks (DSR)," Internet Draft: draft-ietf-manet-dsr-09.txt, April, 2003.

[7] Park, V.D. and Corson, M.S., "A performance comparison of {TORA} and ideal Link State routing," Proceedings of IEEE Symposium on Computers and Communications, IEEE Computer Society Press, June, 1998.

[8] IEEE Standards Department, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY)," IEEE Standard 802.11-1997, 1997.

[9] Weinmiller, J., Schlager, M., Festag, A., and Wolisz, A., "Performance Study of Access Control in Wireless LANs IEEE 802.11 DFWMAC and ETSI RES 10 HIPERLAN," Mobile Networks and Appliations, vol. 2, pp. 55-67, 1997.

[10] Bianchi, G., "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," IEEE Journal on Selected Areas in Communications, vol. 18, no. 3, pp. 535-547, 2000.

[11] Olesinski, W., Rahman, A., and Gburzynski, P., "TARP: a Tiny Ad-hoc Routing Protocol for Wireless Networks," Proceeding of ANTAC'03, Melbourne, Australia, December, 2003.

[12] Akhmetshina, E., Gburzynski, P., and Vizeacoumar, F., "PicOS: A Tiny Operating System for Extremely Small Embedded Platforms," Proceedings of ESA'03, pp. 116-122, Las Vegas, June, 2003.

[13] Broch, J. et. al, "A Performance Comparison of Multi-hop Wireless Ad-hoc Network Routing Protocols," Proceedings of MOBICOM'98, pp. 85-97, October, 1998.