TARP: A Tiny Ad-hoc Routing Protocol for Wireless Networks

W. Olesinski, A. Rahman, and P. Gburzynski
Department of Computing Science, University of Alberta
Edmonton, Alberta, CANADA T6G 2E8
{wladek,ashikur,pawel}@cs.ualberta.ca

Abstract

We present TARP[1] —a low-complexity, small-footprint, tunable routing protocol for ad-hoc wireless networks. The protocol, roughly based on the idea of "controlled flooding" targets low-cost, medium-bandwidth networks built of maintenance-free "smart cards" with minimal cost, complexity, and power consumption. Owing to its (automatically) configurable modes of operation, TARP is highly flexible and can cater to a wide range of node densities and mobility scenarios.

1 Introduction

Generally, routing protocols for ad-hoc networks fall into two groups: *proactive* [12, 5] and *reactive* [13, 9, 10, 11]. Proactive protocols (e.g., DSDV [12]) try to maintain up-to-date routing information at every node—to make sure that it is readily available when needed to set up an end-to-end session. Reactive protocols, on the other hand, only collect the necessary routing information when it is explicitly required for communication. Examples of such protocols include DSR [9], AODV [13, 14], and TORA [11].

Our protocol belongs to the family of reactive protocols. Two main features distinguish it from the other solutions in this class: simplicity (implying small footprint and low power consumption) and flexibility (implying easy and automatic adaptation to diverse applications).

In this paper we describe TARP and discuss some of its properties. We also compare it to DSDV, DSR, and AODV, which can be viewed as the most popular "standard" ad-hoc routing solutions.

2 The Protocol

2.1 An overview

The underlying idea of TARP is controlled 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.

The description of TARP boils down to the meaning of the word "may" in the previous sentence. One can easily see that blind forwarding of all packets that are not received is not going to work. Thus, even the 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 limiting the number of hops, TARP employs three more techniques (rules) that can be viewed as separate components of the entire scheme. The role of the *Duplicate Discard* (DD) rule is to ignore multiple copies of a packet that has been seen and forwarded by the node in the past. The *Sub-optimal Path Discard* (SPD) rule eliminates packets that have deviated too far from a "reasonable" path between the source and the destination. Finally, the *Load Balancing* (LB) rule attempts to diversify the paths by exploring alternative routing opportunities. It counteracts SPD to some extent and accounts for a possible congestion along otherwise preferable paths. It also collaborates with SPD by responding to mobility, i.e., facilitating dynamic reconfiguration of "reasonable" routes.

The exact behavior of all three rules is governed by a set of parameters that determine their focus (or fuzziness). For example, the SPD rule can restrict all traffic between a pair of nodes to the shortest path connecting them, or it may allow some fuzziness by exploring a few (alternative) paths at the same time. While the second approach uses more network resources, it provides for a better responsiveness to the dynamically changing configuration of intermediate nodes. Generally, the proper identification and implementation of this tradeoff is what ad-hoc routing is mostly about.

2.2 Packet header

The network-layer packet header stores the following fields: the source and destination addresses (S and D), the session identifier (s) unique for a given pair $< S, D >$, the sequence number of the packet within

[1] TARP is a proprietary solution of Olsonet Communications, see http://www.olsonet.com.

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), the number of hops traveled by the last packet on the reverse path to S ($\overline{h}$), the mobility factor on the forward path from S to D (m), the mobility factor on the reverse path from D to S ($\overline{m}$). The sizes of those fields depend on the application, but most of them can be very short. For example, both S and D may occupy 16 bits each, s, and k may use 4 bits each, n, r, h, $\overline{h}$ 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 and $\overline{m}$ may be stored into 2-bit fields (representing one of four quantized values). This yields two 32-bit words, with one of those words used to store S and D. Thus, the TARP-specific components of the header reduce to 32 bits, with the packet sequence number being in fact shared with the transport layer.

Note that in contrast to the traditional approach whereby the remaining hop count of a packet is decremented toward zero, TARP uses two fields to implement the hop count limit: the bound set by the source remains constant, while a separate field stores the (increasing) number of hops traveled by the packet. Both values are needed by the SPD rule.

The tuple $< S, D, s, n, k >$ is called the *packet signature*. It uniquely identifies a single packet (at least within a certain time frame).

2.3 The rules

Before a packet is queued for forwarding, the DD rule consults its cache to determine if a packet with the same signature has not been forwarded already. The DD cache stores the signatures of those packets that were recently forwarded by the node.

Note that in addition to following the natural FIFO policy for removing old entries from the cache (as new entries are needed to accommodate more recent signatures), it makes sense to use some other criteria as well. First, depending on the hop count information extracted from the header (the packet appears to be close to the destination), it may be reasonable to discard the packet signature from the cache after a relatively short time. Second, even if there is no demand for new entries (the neighborhood is quiet), old entries should be erased to avoid a wrap-around of n, which would result in mistaking new packets for old ones. The simple formula used by TARP to determine the amount of time after which a signature should expire from the DD cache is $T_r = F_c \times (t_{avg} \times (r - h))$, where t_{avg} is the average transmission time (see below), and F_c is a parameter called the *Flooding Control Constant*.

Note that while the premature cleanup of the DD cache may affect the efficiency of TARP (in the sense of resource usage), it does not affect the formal correctness of the scheme. The worst that can happen is an unnecessary (and wasteful) forwarding of a packet that was already forwarded by the node. 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 attempts to estimate the average transmission time t_{avg} understood as the interval elapsing from the moment a packet becomes queued for transmission at the data-link layer, until the network layer receives a signal from the data-link layer indicating that the packet has been received by (passed to) at least one neighbor. The protocol abstracts from the underlying implementation of the data-link layer (the MAC scheme) and does not by itself specify the exact meaning/implementation of that signal. The role of t_{avg} is to estimate the expected packet processing time at the node (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.[2] The average is calculated as an exponential moving average of the samples, i.e., it is updated as follows: $t_{avg} = C_a \times t_{avg} + (1 - C_a) \times t_t$, where t_t is the last sample, and C_a is a constant (set to 0.65 in our experiments).

The purpose of the SPD rule is to avoid forwarding a packet via a route that takes it away from the destination. The rule uses its own SPD cache storing tuples $< S, D, h_{DK}, h_{SK}, C_{DS}, C_{SD} >$ indexed by unordered pairs $\{S, D\}$. 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 (intermediate) 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 between S and 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 $C_{DS} = \overline{m} \times [(h_{SK} + h_{DK}) - \overline{h}]$. Similarly, for a packet from D to S, K updates h_{DK} and sets $C_{SD} = \overline{m} \times [(h_{SK} + h_{DK}) - \overline{h}]$. Note that $h_{SKD} = h_{SK} + h_{DK}$ 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 $\overline{h}$ 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 route, and its current copy will be

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

discarded further along its path as a duplicate (at the destination or possibly earlier). Consequently, the node may consider dropping the packet, with the confidence of this decision being proportional to the discrepancy between h_{SKD} and $\overline{h}$.

Suppose that $\overline{m}$ is 1. Then if $C_{SD} > 0$ (or $C_{DS} > 0$, depending on the packet direction), 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 drop a packet only if both C_{SD} and C_{DS} are greater than zero.[3] The two values 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 drops the packet without further processing. 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 $\overline{m}$, 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 value of $\overline{m}$, 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 re-tried more often. Additionally, the frequency of those attempts depends globally on the mobility factor $\overline{m}$. They are critical from the viewpoint of responding to the changing routing opportunities in a dynamic network.

The last rule, LB, is a modification of SPD in which the values of the discard counters depend not only on the perceived path length (its discrepancy from the optimum), but also on the intermittent congestion level at the node. The rationale behind LB is that it may be sensible to forward a packet via a suboptimal path (based on its length), if the optimal (shortest) 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 LB, the straightforward factor $\overline{m}$ used in calculation of the discard counters is redefined as $\overline{m} + \overline{d} \times t_t$, where d is a constant dubbed the *diversity factor*, and t_t is the last sample of the packet processing time. With the modification, the formulas for calculating the discard counters take this shape: $C_{SD/DS} = (\overline{m} + \overline{d} \times t_t) \times [(h_{SK} + h_{DK}) - \overline{h}]$.

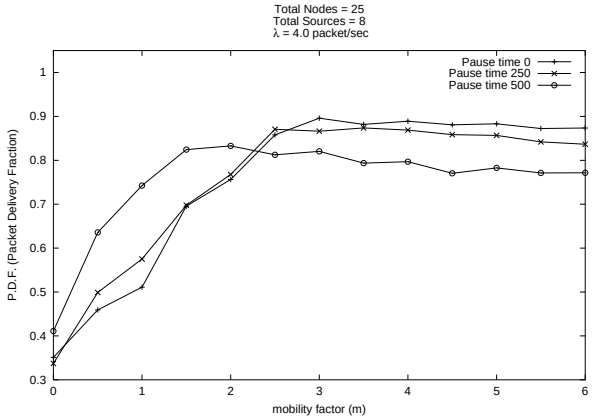
By making the overall factor depend on t_t rather than the average packet processing time t_{avg}, the rule is intentionally "jumpy" with interpreting the load fluctuations at intermediate nodes. Depending on the setting of $\overline{d}$, this approach tends to balance the routing among close paths and also helps in those situations 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.

[3] Otherwise, the neighbors of K, which may also be located on a superfluous route, will not learn about that fact and may keep forwarding other copies of the packet received from elsewhere.

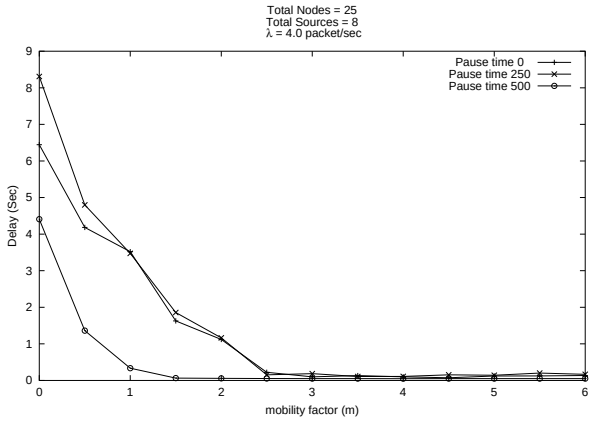
3 Selected Results

3.1 Network model

To evaluate the performance of TARP, we used a detailed simulation model based on *ns*-2 [1] with wireless extensions [4]. The distributed coordination function (DCF) of the IEEE standard 802.11 [6], was used as the MAC layer. The radio model characteristics were similar to Lucent's WaveLAN [7, 16]. Measurements were based on simulations of 25 nodes moving over a flat square area ($670m \times 670m$) for 500 seconds of simulated time.



(a)



(b)

Figure 1: Impact of mobility factor on (a) packet delivery fraction, (b) average delay

For TARP, all MAC-layer packets were broadcast and,

according to 802.11, they were not acknowledged in the MAC layer. To determine whether a packet was successfully forwarded, the network layer would listen for the first copy of the packet retransmitted by a neighbor. This simple and (almost) free acknowledging technique works for all hops except the last one, because no packet is forwarded beyond its destination. Thus, in our implementation, the destination would re-broadcast every received packet—to make sure that its last-hop sender receives the same kind of feedback as every other intermediate node.

We used a *random way point* mobility model parameterized by the *pause time* [4]. According to this model, every node acts in a cyclic fashion whereby it remains stationary for *pause time* seconds, then selects a random destination and moves to that destination at a speed distributed uniformly between 0 and 10 meters per second. We ran our experiments with movement patterns generated for 11 different pause times: 0, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500 seconds (with 0 describing a continuous motion and 500 representing a stationary case).

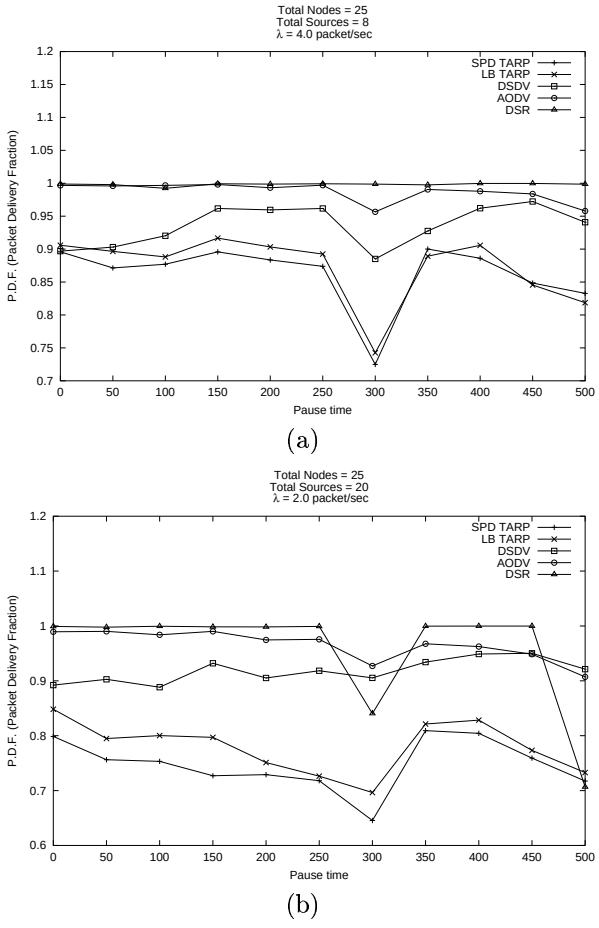


Figure 2: Comparison of TARP with other protocols (a) 8 sources (b) 20 sources

Traffic sessions were bidirectional CBR with the packet size of 512 bytes and the source-destination pairs spread randomly over all nodes. The number of source-destination pairs and the packet generation rate were varied to represent the offered load in the network.

3.2 Performance

The limited size of this paper makes it impossible to include here a complete set of results from our experiments. Figure 1 illustrates the most relevant properties of TARP at a glance by showing the impact of the mobility factor $\overline{m}$ on two performance measures: the *packet delivery fraction* (PDF) (i.e., the ratio of the number of packets reaching the destination to the number of packets transmitted at the source) and the end-to-end delay averaged over all received packets. Three mobility scenarios are considered: continuous movement (pause time = 0), average mobility (pause time = 250), and the stationary case (pause time = 500).

Note that the "best" value of $\overline{m}$ need not be hardwired into the protocol or even into the application. A source can initially set m to a small value (e.g., 0) and increase it gradually while monitoring the quality of service received by the session. Depending on the dynamic configuration of the network, different values of $\overline{m}$ may provide the best performance. Figure 1 indicates that the best value of $\overline{m}$ typically lies between 1 and 3.

Figure 2 shows the PDF in TARP, DSDV, AODV and DSR for two different numbers of sources (sessions). Two versions of TARP are shown: the version labeled as SPD operates without the LB rule, while the LB version augments the SPD rule by the load balancing extension. Thus, the two graphs also illustrate the impact of the LB extension on the PDF in TARP. While TARP yields to the other protocols, its performance is at least comparable to that of the established and considerably more complex solutions.

4 Conclusions

The simple routing protocol presented in this paper appeals to us as a promising avenue for deploying maintenance-free ad-hoc networks based on inexpensive and small hardware. Even in its present preliminary version, TARP can be compared to serious routing protocols, with the gap (Figure 2) appearing hardly unsurmountable. There exist a few directions for further work, which may bridge and possibly eliminate this gap in a future version of TARP.

One such direction involves experimenting with other MAC schemes. Forwarding in TARP is inherently broadcast in nature, and 802.11, which (as a collision avoidance scheme) shows its best at point-to-point communication, is not necessarily the most adequate MAC protocol for TARP. In particular, its RTS/CTS hand-

shake is useless. On the other hand, one would like to see a simple MAC-level acknowledging mechanism applicable to a situation in which several neighbors may receive the same packet (at the same time) and acknowledge their reception synchronously. We are currently experimenting with the idea of fuzzy acknowledgments and a collision avoidance scheme better suited for the kind of forwarding technique used in TARP.

At the same time, we are working on a real-life implementation of TARP in a smart card environment, on our experimental platform consisting of an inexpensive microcontroller, a small operating system [2], and an unorthodox protocol stack developed to support small-footprint wireless applications. We hope that those experiments will bring about new insights which will translate into further enhancements of TARP.

Acknowledgments

The authors would like to thank all subscribers to the *NS* user mailing list <ns-users@ISI.EDU>, who helped us with valuable comments and suggestions during various stages of the development of our simulation model.

References

[1] The Network Simulator: NS-2: notes and documentation. http://www.isi.edu/nsnam/ns/.

[2] E. Akhmetshina, P. Gburzynski, and F. Vizeacoumar. Picos: A tiny operating system for extremely small embedded platforms. In *Proceedings of ESA'03*, pages 116–122, Las Vegas, jun 2003.

[3] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A media access protocol for wireless LAN's. In *Proceedings of SIGCOMM'94*, pages 212–225, August 1994.

[4] J. Broch, D.A. Maltz, D.B. Johnson, Y-C Hu, and J. Jetcheva. A performance comparison of multihop wireless ad-hoc network routing protocols. In *Proceedings of MOBICOM'98*, pages 85–97, October 1998.

[5] T-W. Chen and M. Gerla. Global state routing: a new routing sceme for ad-hoc wireless networks. In *Proceedings of ICC'98*, Atlanta, GA, June 1998.

[6] IEEE Standards Department. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, 1997. IEEE standard 802.11-1997.

[7] D. Eckhardt and P. Steenkiste. Measurement and analysis of the error characteristics of an in-building wireless network. In *Proceedings of the ACM SIGCOMM'96 Conference*, pages 243–254, October 1996.

[8] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark. Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In *Proceedings of MOBICOM'99*, Seattle WA, August 1999.

[9] D. Johnson, D. Maltz, and Y.C. Hu. The Dynamic Source Vector routing protocol for mobile ad-hoc networks (DSR), April 2003. Internet Draft: draft-ietf-manet-dsr-09.txt.

[10] J. Li, J. Jannotti, D. De Couto, D. Karger, and R. Morris. A scalable location service for geographic ad-hoc routing. In *Proceedings of MOBICOM'00*, pages 120–130, 2000.

[11] V.D. Park and M.S. Corson. A performance comparison of TORA and ideal link state routing. In *Proceedings of IEEE Symposium on Computers and Communiactions '98*, June 1998.

[12] C.E. Perkins and P. Bhagwat. Highly dynamic Destination-Sequenced Distance Vector routing (DSDV) for mobile computers. In *Proceedings of SIGCOMM'94*, pages 234–244, August 1993.

[13] C.E. Perkins and E.M. Royer. Ad-hoc On-demand Distance Vector Routing (AODV). In *Proceedings of the IEEE workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 90–100, 1999.

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

[15] F.A. Tobagi and L. Kleinrock. Packet switching in radio channels: Part-ii - the hidden terminal problem in carrier sense multiple-access models and the busy-tone solution. *IEEE Transactions on Communications*, COM-23(12):1417–1433, 1975.

[16] B. Tuch. Development of WaveLAN, an ISM band wireless LAN. *AT&T Technical Journal*, 72(4):27–33, July/Aug 1973.