The Sink-based Anycast Routing Protocol for Ad Hoc Wireless Sensor Networks

Chalermek Intanagonwiwat* and Dante De Lucia**

* USC/Information Sciences Institute
** HRL Research Laboratories

Abstract

The first anycast routing protocol for ad hoc wireless sensor networks is presented and referred to here as the Sink-based Anycast Routing Protocol (SARP). Instead of sending a packet to a specific destination, the packet is delivered to the nearest sink. In some domains of applications, which sink obtains data is not crucial. Nearest-sink delivery consumes minimal bandwidth and power, causes smallest delay, and is best suited for resource location applications. The proposed SARP is highly adaptive, efficient, and suitable for dynamic networks, as shown here by a detailed packet-level simulation on the NS-2 network simulator.

I. INTRODUCTION

In the near future, processor-embedded devices, including sensors, actuators, and tunable components, will have some type of external communication capabilities. Network connection of such systems clearly enables local and global coordination, and provides a new array of sensor applications.

In hostile environments where there is no network infrastructure, network connectivity must be formed on the fly at low cost. Such networks are known as “ad hoc networks”.

Deploying networks in these environments must be rapid and simple. Sensors could be randomly distributed, communicate with each other via wireless communication, and coordinate to provide a local or global view of the environment, or to react to environment changes. Detected information at each sensor could be processed locally, or sent to sink points. A sink could be a data collection point, an actuator, a processing unit, a server or even a sensor.

For data collection applications, which sink obtains data is not crucial. In the other words, data could be anycasted to a sink. Due to limited power and network bandwidth, the data should be delivered to the nearest sink to minimize network bandwidth and power consumption, and experience the smallest delay.

Likewise, in distributed control applications where a sink could be an actuator, the closest actuator reacts to environment changes reported by sensors, and shows a local adaptation behavior. In a sense, an actuator could be viewed as a server reacting to a request from a sensor. This makes anycast mechanisms best suited to resource location applications.

During the past 4 years, a number of multi-hop wireless ad hoc network routing protocols have been proposed, including DSDV (Destination Sequenced Distance Vector) [4], TORA (Temporally-Ordered Routing Algorithm) [5], DSR (Dynamic Source Routing) [2] and AODV (Adhoc On-Demand Distance Vector) [3], to name only several.

All of above are unicast routing protocols, and none of them focuses on anycast routing issues. However, since they are designed to be deployed in hostile environments as ours, some of their techniques can be borrowed and applied in our environment.

We propose the first anycast routing protocol for ad hoc wireless sensor networks called the Sink-based Anycast Routing Protocol (SARP). The features of the proposed protocol are summarized in Table 1 and compared with several of the unicast routing protocols mentioned above.

The key technical issues are: 1) how to route a packet on an optimal path to consume minimal network bandwidth and power, 2) how to achieve a high packet delivery ratio under the presence of route failures caused by node mobility, and 3) how to maintain routes using minimal routing overhead.

The SARP as described above has been simulated on the NS-2 network simulator [6] with CMU wireless and mobility extensions [7] to obtain a realistic, quantitative performance evaluation.
The complete details of the SARP algorithm will be described in the next section. Then, the simulation results will be shown in the Section III. Section IV discusses future work. Concluding remarks are given in Section V.

### TABLE I
Comparison of routing protocols for ad hoc networks

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Routing Type</th>
<th>Protocol Type</th>
<th>On Demand/Periodic</th>
<th>Promiscuous Mode</th>
<th>Route Repair</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSR</td>
<td>Unicast</td>
<td>Source Routing</td>
<td>On Demand</td>
<td>Suggested</td>
<td>At Source</td>
</tr>
<tr>
<td>DSDV</td>
<td>Unicast</td>
<td>Distance Vector</td>
<td>Periodic</td>
<td>NO</td>
<td>Everywhere</td>
</tr>
<tr>
<td>AODV</td>
<td>Unicast</td>
<td>Distance Vector</td>
<td>Both</td>
<td>Not Required</td>
<td>At Source</td>
</tr>
<tr>
<td>TORA</td>
<td>Unicast</td>
<td>Link Reversal</td>
<td>On Demand</td>
<td>NO</td>
<td>At Failure Point</td>
</tr>
<tr>
<td>SARP</td>
<td>Anycast</td>
<td>Distance Vector</td>
<td>On Demand</td>
<td>Suggested</td>
<td>At Failure Point</td>
</tr>
</tbody>
</table>

### II. SARP

In this section, system assumptions and an intuitive idea behind our design decisions will be given, followed by a detailed discussion of the algorithm.

### Assumptions

The first assumption is the symmetric link requirement. If node A can hear node B, node B will be able to hear node A. This assumption is reasonable because the widely used MAC protocol for wireless networks, IEEE 802.11, requires an RTS/CTS/DATA/ACK exchange for all unicast packets. The exchange will not be possible unless the link is symmetric. The purpose of this assumption is to enable link reversals. The uplink and downlink do not necessary have the same bandwidth.

In our protocol design, the reliability issues have been separated out from routing issues. There is no guarantee that a packet will reach a sink. It is more reasonable to ensure only that if there exists a route, our routing protocol is aware of it. Moreover, the reliability issues are more relevant to data types and applications. Although, under route failures, it will perform retransmission after a new route is found, this capability is very limited due to the buffer size. This is done simply to relieve the problem, not to solve it.

Our protocol goal is to deliver data to the nearest sink. There is no exact destination for a packet. Thus, routing decisions must rely on hop count and data type instead.

Nevertheless, overhearing, an optimization technique suggested by DSR, will be the key to learning potentially useful information without causing any additional overhead on the limited network bandwidth. Nodes operating in promiscuous mode will receive all packets that the interface overhears by disabling address-filtering mechanism. If the forwarder of a packet always puts its hop count in the packet, that packet will act as a route update due to promiscuous mode. Therefore, a shorter route can be detected, and the shortest path in active regions will be found eventually.

Additionally, nodes must not move too fast, compared to their transmission range. This assumption is needed to enable a successful communication in this network.

### Algorithm

Before a detailed discussion of the algorithm is given, let us clarify a confusing word here. In wireless communications, a broadcast can be local or global.

A local broadcast means that only nodes within the transmission range obtain the broadcast packet. All broadcasts mentioned in this paper are local.

For a global broadcast, all nodes in the network, without network partitioning, receive the packet. In this paper, this kind of broadcast is called flooding. In flooding, a packet is propagated or broadcast locally by all nodes hearing the packet at the first time.

However, some flooding may be performed in a controlled manner or have a limited scope of propagation. Therefore, only some nodes will receive the packet. This kind of flooding is not a global broadcast.

The SARP protocol consists of three mechanisms: Route Establishment, Route Recovery and Hop Count Update. All are described as follow.

### Route Establishment

Route Establishment is the mechanism for a new sink to express its interest and for sources to set up routes to that sink.

A new sink will initiate communication by broadcasting its interest (data type). An interest packet can have a TTL to limit the scope of propagation if only local interest is required.

The sink’s neighbors receive the interest and learn that they are 1 hop away from a sink. Then, they put their hop count in the interest packet, and rebroadcast it.

Nodes having already received the packet will ignore it. Other nodes receiving this packet for the first time will learn their distance from a sink by looking at the forwarder’s hop count in the packet. If the forwarder’s hop count is $n$, they know that they are $n+1$ hops away, and their next hop is that forwarder. This is possible due
to the symmetric link assumption. Again, they will put their hop count in the packet, and rebroadcast it. Eventually, all nodes in the network receive the interest packet. They know how far they are from a sink and how to route a packet to a sink.

In case of multiple sinks, nodes will keep only the information for the sink having minimum hop count. They choose to route a packet to the nearest sink.

A source can be any node in the network. Before sending a packet to its next hop, a source will put its hop count in the packet. A node receiving a data packet, again, will put its hop count in the packet before forwarding it. This process continues until the packet reaches a sink or it is dropped. By turning off address filtering, all packets will behave as virtual route updates in active areas.

When a sink receives a data packet, it broadcasts a one-hop acknowledgement with hop count 0 because a passive acknowledgement is not applicable for a sink. (A passive acknowledgement is used to show that the link is still alive in the second option of the next subsection.) This single hop acknowledgement packet will not be forwarded, and, again, it serves as a route update.

**Route Recovery**

Under presence of mobility, two nodes in a route may move out of range of each other and cause a route failure. To detect route failure, link breakage detection feedback from the IEEE 802.11 MAC layer indicating when a packet could not be forwarded to its next hop can be used.

Another option is to take advantage of promiscuous mode and a passive acknowledgement. When forwarding a data packet, a forwarder will set up a timer slightly more than the round trip time (RTT) to its next hop. If it does not hear any action from its next hop before the timer expires, it will assume a route failure, and Route Recovery mechanism will be deployed to determine a new route. (It is noted that the first option, link breakage detection from the IEEE 802.11 MAC layer, is used to detect route failure in our simulation.)

Unlike DSR and AODV, a Route Recovery will be performed at a node that detects a route failure. The source of the packet will not be informed. Clearly, this can save the network bandwidth and power required to inform the source. However, the key factor is that the node detecting this route failure may be on a route of other sources as well. Solving the route failure at this node now can prevent potential future route failures for other sources having this node on their route.

To perform Route Recovery, the node detecting the route failure floods a ROUTE REQUEST packet through the network in a controlled manner. A ROUTE REQUEST packet contains the requester’s hop count to specify repliers’ qualification.

Only nodes with lower hop count than the requester’s are qualified and can reply. Unlike TORA, this ensures a progress, at least one hop toward a sink. Moreover, it will not have the looping and counting-to-infinity problem.

In Route Recovery, an expanding ring search can be used as an optimization technique to limit a scope of flooding. A requester first sends a ROUTE REQUEST with the maximum hop limit set to zero to prevent its neighbors from rebroadcasting it.

If there is no reply received before its nonpropagating search timer expires, it expands the ring search by setting the maximum hop limit to N. This N is not specified in SARP yet because the right value of N is not obvious and depends on the network size. One possible value of N is the average hop count.

If, still, no answer is received, the new ROUTE REQUEST packet will be generated and propagated globally by setting the maximum hop limit to infinity.

Since N is not decided yet, in our current simulation, if nonpropagating search fails, a propagating search is performed with the maximum hop limit set to infinity.

When a node receives a ROUTE REQUEST, if it cannot reply and the request’s maximum hop limit is higher than zero, it rebroadcasts the request and decrement the maximum hop limit by 1. If it has already experienced this request packet, it ignores the packet.

Moreover, it keeps state about who rebroadcasts this request. Therefore, if it is qualified to reply or receives a ROUTE REPLY, it can send or forward it toward the requester by using this state to reverse the path.

In order to limit the number of ROUTE REPLY packets, if a node has already sent or forwarded a reply for a request, it ignores the other replies of that request.

Again, it can stress this idea even more by taking an advantage of promiscuous mode. A qualified replier, overhearing a ROUTE REPLY for a request, will suppress its own reply.

**Hop Count Update**

An important SARP rule is that a data packet is always forwarded from a node with higher hop count to a node with lower hop count to ensure a loop-free delivery. Therefore, maintaining the correctness of a node’s hop count is essential.

This rule is subjective to violation mostly when a node performs Route Recovery. All nodes along the new route potentially have higher hop count than their
previous value. Therefore, a node having any of these nodes as its next hop is likely to violate the rule.

By operating in promiscuous mode, a node can easily detect if it violates the rule. Upon detection, the violating node corrects its hop count. Then, it broadcasts a hop count update packet containing its new hop count. Therefore, a neighbor having that node as its next hop can check if the new hop count makes it become a violating node. If it does, that neighbor node will correct its hop count and send its own hop count update packet. The process continues until there is no violating node. This type of packet will not be rebroadcast.

III. SIMULATION

In this section, the methodology to simulate and evaluate performance of SARP will be described first. Then, simulation results of SARP will be shown.

Although all simulation parameters here are very similar to those used in [1], the simulation results are not meant to be compared with unicast routing protocols. Since they attempt to achieve different goals, making a comparison would not make sense.

Our performance evaluation is based on simulations using the NS-2 network simulator with the CMU wireless and mobility extensions. The physical radio characteristics were chosen to approximate the 914 MHz Lucent Wave LAN DSSS (Direct Sequence Spread Spectrum) radio.

The wireless nodes form an ad hoc network, and move according to the random waypoint model over a rectangular (1500m x 300m) flat space for 900 seconds of simulation time. Initially, all nodes remain stationary for pause time seconds. (A pause time of 0 second represents constant mobility.) Each node then moves to a random destination at a uniformly distributed speed between 0 and 20 m/s. After reaching the destination, a node pauses again for pause time seconds, and selects a new destination. This is repeated until the end of the simulation.

The simulation scenarios consist of 2 different network sizes (50 and 60 nodes), 5 different pause times (0, 30, 60, 120, and 300 seconds), 3 different numbers of sources (10, 20, and 30 sources), and 2 different numbers of sinks (1 and 5 sinks). In total, there are 60 scenarios.

Our traffic sources are constant bit rate sources (CBR) sending with a rate of 4 packets per second, and the packet size is 64 bytes. Each source starts generating packets at a time uniformly distributed between 0 and 180 seconds. Each sink broadcasts its interest once at a time uniformly distributed between 0 and 10 seconds.

To evaluate SARP performance, three metrics are used: packet delivery ratio, routing overhead and path optimality.

The packet delivery ratio is the fraction of data packets successfully delivered. It demonstrates the loss rate seen by the transport protocols, effects maximum network throughput, and characterizes the completeness and correctness of SARP.

The routing overhead is measured by counting the number of routing packets sent during the simulation. This metric shows scalability, and efficient usage of network bandwidth and battery power. The larger number of routing packets introduces more chances of packet collisions, longer delay, and more packets dropped.

The last is path optimality. The simulator determined the optimal hop for each packet and the average optimal hop beforehand. Then, the real hop taken for each packet is recorded and the average real hop taken is computed. The smaller the difference between the average optimal hop and the average taken hop, the more efficient the use of network bandwidth and power.

Figure 1: Comparison between numbers of sources and the packet delivery ratio as a function of pause time.

In this simulation, we do not use pause times of more than 300 seconds because SARP will perform perfectly, and will not give any interesting results.

Figure 1 shows packet delivery ratio of 3 different number of sources in a network of 50 nodes and 1 sink. All of them perform well, and get packet delivery ratio higher than 86 %. In case of 10 and 20 sources, SARP performs even better with a packet delivery ratio of over
94%. With traffic loads of 30 sources, it seems to experience difficulty when nodes move constantly. The main reason is that all traffic loads must go to the same sink. Due to node mobility, some links may fail. Traffic loads must concentrate on only a few links or nodes hence causing congestion.

**Figure 2:** Comparison between numbers of sinks and packet delivery ratio as a function of pause time.

By increasing the number of sinks from 1 to 5, SARP performs perfectly as shown in Figure 2. This occurs because traffic loads have been balanced amongst 5 sinks, and so will not experience the same problem described above. It also supports the fact that the results in this paper should not be compared to [1]. In testing unicast routing protocols, 30 connections usually have more than 1 sink.

**Figure 3:** Comparison between numbers of total nodes and packet delivery ratio as a function of pause time.

Nevertheless, increasing the number of total nodes in the network from 50 to be 60 nodes also eliminates this difficulty as shown in Figure 3. Since the area size is still the same, 60 nodes introduce more network connections. Thus, traffic loads are less concentrated to each node and link.

**Figure 4:** Comparison between numbers of sources and routing overhead packets sent as a function of pause time.

Moreover, increasing the number of total nodes in the network from 50 to be 60 nodes also eliminates this difficulty as shown in Figure 3. Since the area size is still the same, 60 nodes introduce more network connections. Thus, traffic loads are less concentrated to each node and link.

**Figure 5:** Comparison between number of sinks and routing overhead as a function of pause time.

Figure 4 shows the routing overhead of 3 different number of sources in a network of 50 nodes and 1 sink. As expected, the higher number of sources generates more routing packets. However, it should be noted that
one-hop acknowledgements have been counted as routing packets as well. In case of 10 sources, it generates about 32,000 data packets. About 64,000 and 96,000 data packets are sent by 20 and 30 sources, respectively. Hence, most part of routing overhead comes from one-hop acknowledgements.

Figure 6: Comparison between number of total nodes and routing overhead as a function of pause time.

Since we used link breakage detection feedback to detect route failures in this simulation, a single hop acknowledgement was not necessary. However, it was still used due to its functionality as a route update. We could turn it off, and eliminate a significant portion of the routing overhead, but the higher average taken hop was expected. Without a single hop acknowledgement, the number of packets sent by each sink is smaller. Since each packet has a secondary role as a route update, the less the packets are sent, the less frequently the routes are updated, resulting in the higher average taken hop.

Again, since a Route Recovery is performed at a node detecting a route failure, there is no routing overhead to inform the source. It can prevent future route failures of the other sources having this node on their route, and significantly reduce routing overhead.

As shown in Figure 5, when increasing the number of sinks from 1 to 5, routing overhead is reduced significantly. Due to more sinks, the average number of hops from a source to the nearest sink is lower. It increases a chance that the nonpropagating search will successfully find a new route in a Route Recovery. Since the number of propagating search is likely reduced, the routing overhead is smaller.

In Figure 6, we increase the total number of nodes from 50 to 60. The results show slightly more routing overhead. In a global propagating search of a Route Recovery, a ROUTE REQUEST packet is flooded through the network. In the case of flooding, routing overhead depends on the total number of nodes. The higher total number of nodes undoubtedly causes more routing overhead.

Figure 7: Comparison between average optimal hops and average hops taken as a function of pause time.

Figure 8: Comparison between number of sinks and average hop difference from the optimal path as a function of pause time.
Figure 7 compares the average hop count of optimal paths with the average hop count of actual paths. The difference between them is less than 0.1 hops in all cases here. It means that, in every 10 packets, there is one taking a path with one hop more than the optimal path length. This is acceptable since this network is highly dynamic.

Nevertheless, since Route Recovery is performed at a node detecting a route failure, the number of hops before and after a Route Recovery are included. Therefore, this number is realistic. In some unicast routing protocols, a Route Recovery is performed at the source not at a node detecting a route failure. After finding a new route, the source retransmits the packet. The number of taken hops before Route Recovery must be taken into consideration. Otherwise, it will not be a valid comparison between protocols.

In Figure 8, by increasing a number of sinks from 1 to 5, the average hop difference between the optimal path and the taken path is reduced significantly. In SARP, a node learns a shorter route by overhearing a packet from an active region. Since more sinks allow better load balancing, more regions are active. Most routes in the network are always updated.

Figure 9: Comparison between numbers of total nodes and average hop difference from the optimal path as a function of pause time.

With 10 more nodes in the network, the number of network connections is higher and the traffic concentration at each node and link is more balanced. It leads to larger active regions. More routes are up to date. Consequently, the average hop difference between the optimal path and the taken path is slightly lower as shown in Figure 9.

IV. FUTURE WORK

In wireless sensor networks, the major concerns are power limitation and scalability. To make efficient use of scarce power and achieve the smallest delay, most routing protocols are based on finding the shortest path. Traffic loads tend to concentrate on particular nodes and links. Hence, some nodes become more important in routing packets, and they tend to run out of power before the others, resulting in node and route failures. The probability of network partitioning becomes higher with a lower number of operating nodes. Moreover, traffic loads may concentrate even more on the remaining nodes and links, and worsen the situation. Therefore, a desirable behavior of an ad hoc routing protocol is a low variance of node lifetime. Ideally, all nodes in the network should run out of power about the same time.

By using power-aware metrics suggested in [8], for instances, cost/packet and maximum node cost, along with delay and distance, node and network life could be extended. In SARP, a node with low power could avoid routing packets by increasing its hop count as a simple power routing mechanism. We will focus more on power routing in future work.

In order to achieve high scalability, we need to minimize routing overhead. Promiscuous mode and expanding ring search are optimization techniques used for this purpose. Some future work could be done in this area, for example, determining the right value of N in propagating search which can potentially reduce flooding. However, more intelligent flooding scheme needs to be explored as well. In the other words, if flooding is unavoidable, the best way to do it is to use a minimum number of rebroadcasts. Techniques used in Zone Routing Protocol [9] could be helpful in this context.
V. CONCLUSIONS

A new anycast routing algorithm for ad hoc wireless sensor networks, SARP, has been presented, and simulated on the NS-2 network simulator.

Operating in promiscuous mode, the proposed protocol takes advantages of useful information in overheard packets to detect route failures, shorter routes and hop count violations, and to suppress route replies at no additional cost.

Local repairs for route failures can reduce routing overhead spent to inform the sources, and prevent potential future route failures.

All packets in SARP, except route requests, have a secondary role as a route update. Combining all these ideas with promiscuous mode and expanding ring search has helped eliminate a large number of routing packets to achieve scalability, path optimality and high packet delivery ratio at low cost.

The simulation results show good adaptation to topology changes and efficient use of network bandwidth and power.

One interesting observation is that scalability of the system can easily be improved by adding more sinks.

The future work in this area will be on power-aware anycast routing and intelligent flooding schemes. Power remaining in each node should be taken into consideration in routing a packet to increase network life. Intelligent flooding could reduce unnecessary rebroadcasts and results in less routing packets, more scalability, higher packet delivery ratio, and longer network lifetime.

ACKNOWLEDGEMENTS

The idea of interest propagation is inspired by Directed Diffusion, an outcome of the 1998 DARPA ISAT study, chaired by Deborah Estrin. We would like to thank especially Deborah Estrin, Ramesh Govindan, and Nirupama Bulusu for their useful comments. This work is supported by USC/ISI and HRL.

REFERENCES


