Dynamic Data Forwarding in Wireless Mesh Networks

Tadashige Iwao, Kenji Yamada, Masakazu Yura, Yuuta Nakaya
Smart Network Division
Intelligent Society Business Unit
Fujitsu Limited
Fukuoka, Japan

Alvaro A. Cárdenas, Sung Lee, Ryusuke Masuoka
Trusted Systems Innovation Group
Fujitsu Laboratories of America
Sunnyvale, CA, USA

***Abstract*—Unreliable wireless links can cause frequent link (and route) failures, creating a major challenge for routing protocols who need to constantly repair routes and find alternate paths. In this paper we propose DADR (Distributed Autonomous Depth-first Routing), a new distributed distance-vector routing protocol designed to adapt quickly to changing link conditions while minimizing network control overhead. In our algorithm, when a link fails, data packets are rerouted through an alternate next hop, and the information about the failed link is propagated with the data packet; therefore, routes are updated dynamically and with little overhead. We have implemented DADR on several link-layer technologies and deployed it in different applications, including AMI deployments in Japan [1]; all implementations resulted in reliable networks that were easy to set up, maintain, and resilient to changing conditions.**

I. Introduction

Wireless mesh networks are becoming a key enabling technology to many smart grid initiatives, such as Advanced Metering Infrastructure (AMI) networks, home automation, Supervisory Control and Data Acquisition (SCADA) networks, and building Heating, Ventilating, and Air Conditioning (HVAC) systems. Many of these systems require reliable, efficient, and timely packet delivery in unreliable wireless environments.

During the last decade there have been several research efforts for designing wireless routing protocols for Mobile Ad Hoc Networks (MANETs) and sensor networks. Despite their contributions, there are concerns about applying many of these protocols in large and unreliable wireless mesh networks [2].

One of the main problems affecting many routing protocols in wireless networks is the unpredictable nature of wireless links, where there is no clear distinction between "working" and "non-working" links [3]. While using new routing metrics like the expected transmission count (ETX) [4] (instead of using the hop count) can alleviate some of the problems introduced by links with intermediate levels of loss, these links can still cause problems for traditional *two-phase routing algorithms*.

Most routing algorithms are generally composed of two independent phases: (1) route discovery and maintenance—coordinated by the control plane—and (2) data forwarding—coordinated by the data plane. The data plane is where data forwarding is done, and it is purposely kept simple. When a packet arrives at a node, a simple lookup operation sets the packet in the right path. The routing plane, on the other hand, is responsible for keeping the routing table accurate so that the data plane functions correctly.

While having independent routing and data forwarding planes creates a sound routing architecture, this design principle has limitations in addressing the following problems:

1. **Path quality depends on packet sizes.** The size of a packet can affect the reliability of a link due to the bit error rates of links. Sources generally send *data packets* of different sizes to their destinations. Because traditional routing protocols discover routes with smaller-sized *control packets*, the network topology seen by control packets may be different from the network topology seen by data packets.
2. **Path quality depends on time.** The reliability of links may be different at the time when the route was discovered, and the time when the data is forwarded. For example, a truck parked in front of a smart meter may sever the ability of this particular node to forward packets temporarily.
3. **Control plane overhead.** In networks with unreliable links, the control overhead of fixing a broken path must be paid often. This overhead is particularly problematic for routing algorithms that use flooding to search for new paths.

One-phase routing algorithms try to mitigate these problems by discovering routes during the data forwarding phase [5] [6] [7]. While these protocols can handle better dynamic topologies, they are inefficient in practical deployments; some of them even flood the network with data packets, creating an even larger overhead.

In this paper we introduce DADR (Distributed Autonomous Depth-First Routing), a new distributed wireless mesh routing protocol designed to improve how independent control and data planes handle unreliable links, while addressing the inefficiencies of one-phase routing schemes. Our solution is composed of two mechanisms: **1)** a light-weight control plane used to keep a "soft" routing table with redundant paths at each node, and **2)** a forwarding plane which can reroute data packets by doing a depth-first search guided by the "soft" routing table.

DADR has the following characteristics: **1)** if the network topology is static, and the wireless links are reliable, the light-weight control plane determines all the routes, and DADR behaves exactly the same as any other traditional distance vector routing algorithm. **2)** If the topology changes frequently, DADR does not flood the network in search for new routes; instead, the data forwarding plane uses the redundant paths in the routing table. Control information is maintained during data-forwarding so information learned from this search is used to update the routing tables.

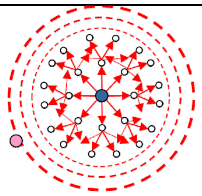
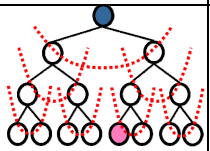
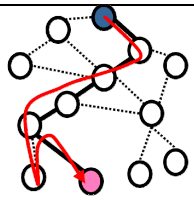
We have implemented our routing technology on IEEE 802.11, IEEE 802.15.4, and IEEE 802.3 (Ethernet). We have deployed DADR in different applications; in particular, DADR is currently being used in some AMI deployments in Japan [1]. We have conducted successful field deployments with up to 1500 nodes in a flat (non-hierarchical) wireless mesh network to test the scalability of the proposed technology. All implementations resulted in a network easy to set up, to maintain, and resilient to changing network conditions.

The paper proceeds in Section II with a review of previous work. Section III describes the routing algorithm, and in Section IV we show simulation and experimental results.

II. Related Work

The challenges in wireless networks are not new: there has been a lot of research in designing effective routing protocols for wireless networks. Within the IETF MANET working group, two popular proposals are Ad hoc On-Demand Distance Vector (AODV) [8] and Optimized Link State Routing (OLSR) [9].

Table 1. DADR attempts to minimize the control load overhead when repairing a route.

	Path recovery after link failures		Overhead
AODV	Breadth First Search		250,000 control packets for a 500-node network
OLSR	Modified Breadth First Search		25,000 control packets for a 500-node network
DADR	Guided Depth First Search (Greedy Search)		5,000 control packets for a 500-node network

While these algorithms introduced many good design principles for wireless networks, they have scalability problems in large, unreliable networks. In AODV, at the time of data packet transmission, a control packet is sent to the whole network concentrically, searching for the right path. When communication is down, the node tries to find a path again by flooding the network. If network links are highly dynamic, AODV can generate excessive flooding and degrade the performance of the network. Similarly, in OLSR, when a path breaks a node tries to find a path as soon as possible with limited flooding, leading again to a large number of control packets.

Scalability problems for low power and lossy networks originating from the large response overhead to link failures and route updates is shared by many other popular routing protocols [2]. To minimize the overhead for repairing routes, DADR reroutes data packets through alternate links by a modified depth-first search guided by a routing table which includes the cost of forwarding packets through alternate neighbors (Table 1).

The notion that flooding is counter-productive to networks with frequently changing topology is not new: in 1981, Gafni and Bertsekas were one of the first to identify this problem and propose a distributed solution [10]. Their original ideas were later extended by TORA [11]: a multi-path routing protocol that includes localized recovery from route failures. It was later shown that under frequent topology changes, TORA exhibits high-internodal coordination overhead [12], thus undermining the advantage of having multiple paths.

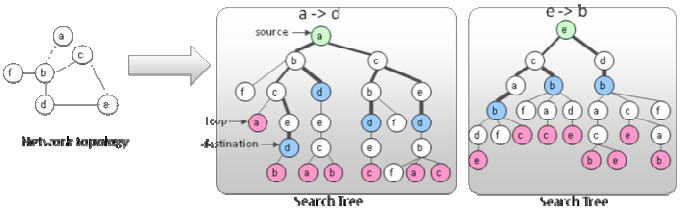


Fig. 1. The greedy search algorithm must be able to detect loops.

There are many other multipath routing algorithms proposed in the literature [13] [14] [15] [16] [17]. While multipath routing protocols minimize the end-to-end data packet delay, improve reliability and reduce the frequency of route discovery operations, they tend to increase the control packet overhead for maintaining multiple routes (in DADR there is no additional control packet overhead for maintaining a set of next hops).

In addition, when all the possible routes fail for an intermediate node, multipath algorithms must initiate a new route discovery. On the other hand, if all the possible next hops in DADR fail to deliver the packet, DADR simply returns the packet back the way it came so a node closer to the source can try an alternate route.

Furthermore, traditional multipath protocols were designed with independent control-plane and data-plane phases, and as mentioned in the introduction, this separation creates limitations during the data forwarding phase.

One-phase routing eliminates some of these problems. Geographical routing is an example of one-phase routing because there is no explicit discovery of routes [18] [19]. The main drawback of these schemes is the availability of a location service, which is not always a feasible or inexpensive solution.

To avoid depending on location services while maintaining a one-phase routing scheme, several other approaches have been proposed in which path discovery is made at the same time with data forwarding [5] [6] [7]. While these protocols can handle better dynamic topologies, they are inefficient in practical deployments; some of them [6] even flood the network with data packets.

In this paper we address the problems with one-phase routing schemes by proposing the use of a hybrid scheme composed of two mechanisms: (1) a "soft" proactive routing protocol used to keep a "soft" routing table in each node, and (2) a data forwarding algorithm that can use alternate paths in case of link failures, and which uses the information collected during the data forwarding phase to update its routing tables.

DADR is similar in spirit to FCP [20] (for wired networks with link or router failures) and BAF [21] (for static wireless networks). DADR, FCP, and BAF attempt to forward data packets through alternate routes, avoid the use of control packets to alert the network of a link failure, and use the information learned from data forwarding to update the view of the network.

The main difference between DADR and FCP and BAF is that the two latter are link-state routing protocols, and therefore every router has a consistent network map (which facilitates finding alternate paths). On the other hand, DADR is a distance-vector protocol: therefore, it has less overhead, but finding alternate routes while avoiding loops is a challenging problem.

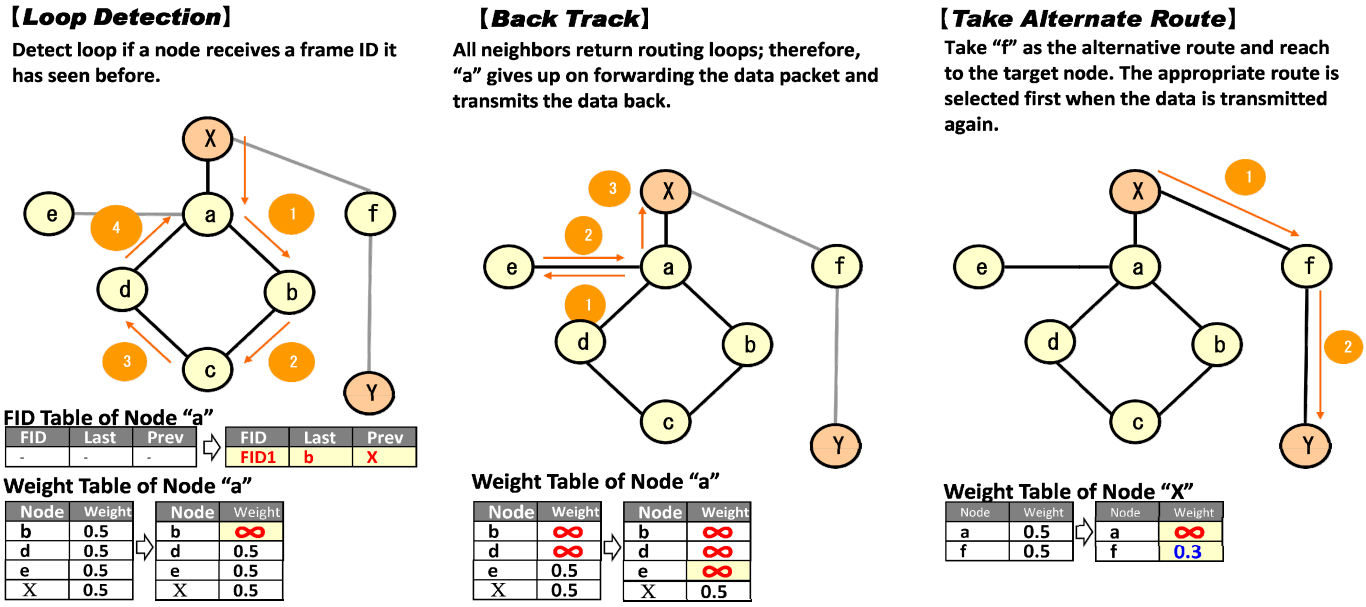


Fig. 2. Operation of data forwarding plane when node X attempts to send a data packet to node Y. Note that the routing table is updated with results from the data forwarding. The infinite symbols represent the maximum weight for a link, which can be reduced if future HELLO messages or successful data transmissions confirm the link can be used as a path towards the destination.

In addition (as far as we know) FCP and BAF have not been implemented and tested in a real deployment. DADR, on the other hand, has been deployed and is currently in use in several industrial scenarios. One of our smart grid AMI deployments currently supports 1500 smart meters in a flat topology reporting back to a single gateway. We are also currently working for a field test in the United States to obtain results we can share openly with the research community. In this test we will deploy 2107 nodes (smart meters) in a non-hierarchical topology with a single gateway (data aggregation point).

III. DADR Protocol Description

DADR is composed of a "soft" routing plane algorithm, and a data forwarding plane where learned information (such as bad signal quality, congested link, etc.) is reflected in the routing metrics.

A. Control Plane

For the routing plane in DADR we select distance-vector routing over link-state routing or source-based routing because we want to keep the protocol as lightweight as possible. Distance vector routing requires very little information stored at the network nodes and less processing power when compared to link-state routing. Similarly, source-based routing will incur in significant overhead for large networks consisting of routes with several hops (e.g., 60) before reaching the destination.

Besides selecting distance-vector routing, another choice is the selection between proactive and on-demand routing. In principle, DADR can use either technology depending on the deployment characteristic and requirements. In on-demand routing, for example, once a route is discovered, it can be maintained with the help of the data forwarding plane. For most applications, however, we found that there is a clearly defined set of global destinations that nodes have to contact regularly; therefore, having a "soft" proactive routing protocol is a good choice

We use the term "soft" to refer to the use proactive routing without control messages for path maintenance or path repair. By avoiding the frequent control overhead required for path maintenance, we eliminate one of the main drawbacks of proactive routing.

In DADR, the only control messages are periodic HELLO messages (containing routing table information) exchanged among neighboring nodes. Each node maintains forwarding knowledge in a "soft"-state routing table. The routing table for each node contains a list of all known destinations, the cost to reach each destination, and a set of link-layer and network-layer parameters used to estimate the cost.

Because we do not use a route discovery phase in the control plane when paths break, we keep a set of at most k (when available) possible next hops for each destination. When a path breaks during the data forwarding phase, the data forwarding algorithm will select one of these alternate paths dynamically, in search for an alternate way to reach the destination.

B. Data Forwarding Plane

If there are no path breaks or link failures while forwarding a data packet, the data forwarding plane will work like any other protocol, forwarding data packets along the preferred minimum cost route; however, if the data forwarding plane cannot send a data packet via the preferred route (as defined by the routing table), it will attempt to forward the packet along an alternate neighbor.

In order to search for a path, DADR employs a depth-first search. In principle the data forwarding plane can route packets without the need of the control plane; however, it may take a long time to find the final destination depending on where the search begins. To guide the search algorithm, we use guide the search with the routing table.

One issue with applying a greedy search to a network is that the search algorithm expects the search space to be tree-based. On the other hand, a network may be formed with multiple loops; as an example, the network topology shown in the left of Fig. 1, can be traversed via the search-tree shown in the right of the figure. A link is terminated when it hits either a loop or a leaf node.

To detect forwarding loops, each data packet has a unique Frame ID (FID), and each node forwarding a packet, stores the FID of the packet in a data structure. The FID table for $k=3$ can be seen in Table 2.

The forwarding algorithm performs the following steps:

1) Basic Forwarding: A node receiving a data packet with a FID not currently stored, creates a FID table for the packet (to be deleted after a FID_timer expires indicating that the data packet is

assumed to be delivered correctly), and then forwards the packet to a neighboring node (except to the sender) to determine the next path in a greedy search. If a node does not receive FID

2) Loop Detection: A loop is detected if a node receives a FID that it has seen before (a FID that is currently stored in the node).

3) Backtrack: When a node has attempted to forward the packet over all possible next hops and determines that there is no available route (only loops and packet drops), then the packet is sent back to the sender with that information.

4) Route Avoidance: The routing table is updated from the information gathered during data forwarding in different ways:

a) When a node detects a loop, it performs a route poisoning to indicate that the path is no longer available and should be removed from the routing tables.

b) If a node **does not** receive an acknowledgment for the packet it tried to forward, it decreases the **forward delivery ratio** metric (one of the key components of the link cost) to indicate that the link is less reliable than our previous estimate.

c) If a node receives an acknowledgment, it increases the **forward delivery ratio** to indicate that the link is more reliable than our previous estimate.

Table 2. The FID table is used for detecting ACK timeouts, for rerouting data packets, and for detecting routing loops. A FID table is created (or updated) whenever a DATA packet is transmitted.

Parameter	Description
Frame ID	Sequential number given to each data frame. Used to identify previously seen data packets.
Previous hop	If forwarding fails, return data packet to this node.
Source	Origin of data frame.
Destination	Final recipient of data frame.
ACK. Flag	Flag representing the status of DATA ACK. reception.
Next hop 1	Neighbor with the minimum routing cost to reach the destination.
Next hop 2	Neighbor with the second lowest routing cost to reach the destination.
Next hop 3	Neighbor with the third lowest routing cost to reach the destination.
Last	Address of the last neighbor through which we attempted to forward the packet.

An example of the data forwarding algorithm can be seen in Fig. 1.

While we have found that this data forwarding algorithm performs reliably in practice, there are a couple of disadvantages. First, we are introducing state in the data forwarding phase, which increases the CPU and memory overhead of intermediate nodes. Second, if acknowledgments are lost, there can be multiple packets with the same FID circulating in the network, and therefore, there might be some loop detection false positives. Finally, if the FID timer expires too early and the FID table is deleted, there may be some data forwarding loops that are not detected (false negatives).

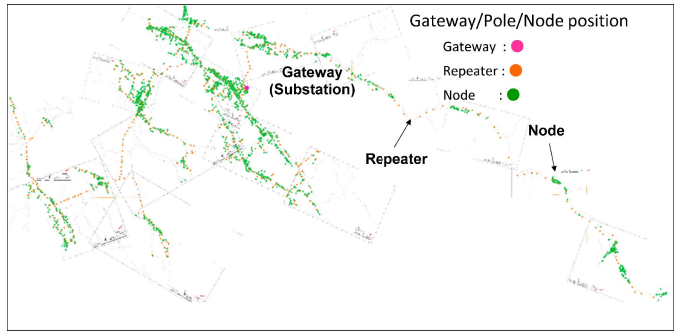


Fig. 3. AMI network.

We have not seen these problems degrading the performance of DADR in our deployments, in particular, the lack of acknowledgments is a good indication of packet drops; however, we are actively investigating new ways to optimize the parameters, timers, decisions, and data structures used in the data forwarding phase.

We are currently working in minimizing the FID processing and storage overhead, and in improving our loop detection mechanisms. For example, one option is to sample a subset of packets by adding a flag to the packets we want to keep in the FID table. This flag can be generated upon a route failure (i.e., we only start collecting FID data after a route fails). We are also currently investigating different strategies on how to modify the routing table after a loop is detected, or after we fail to receive a data ACK. How we modify the routing table after these events will determine the side effects from loop detection false positives and false negatives. One advantage we have is that DADR is a time-synchronized routing protocol, so we can use time to live timers for data frames and select FID timers based on them. Similarly, after a lost ACK, a node can wait until FID timers expire (for applications that can tolerate delays) before retransmitting the packet in order to avoid false positives. Finally, data headers carry a flag indicating the number of times a data packet has been retransmitted; therefore nodes receiving these packets can assume the packet is a possible duplicate, and not penalize the routing table with false routing loops.

IV. DADR Performance Results

We have implemented and deployed DADR in several environments. In particular, we have deployed DADR in AMI networks in Japan [1]. One of our deployed AMI networks consists of 1500 smart meters connected in a flat topology and reporting to a single gateway. Unfortunately, due to contractual agreements with our customers, we cannot show data collected from these deployments. Therefore, in this paper we can only show experimental data of smaller deployments we conducted in our laboratories. In addition, we show simulations we have performed in preparation for a field test we are currently deploying in the state of New Mexico (USA). This field test consists of a 2107 smart meter network in a flat topology. The results of this field trial will be open and we will present them to the research community as soon as possible.

A. AMI field trial preparation

We are preparing for a field trial in New Mexico in an area of 304 square miles containing mountainous and residential terrains. After an onsite load test of the radio wave environment we created a radio propagation model to represent accurately real conditions in our simulations.

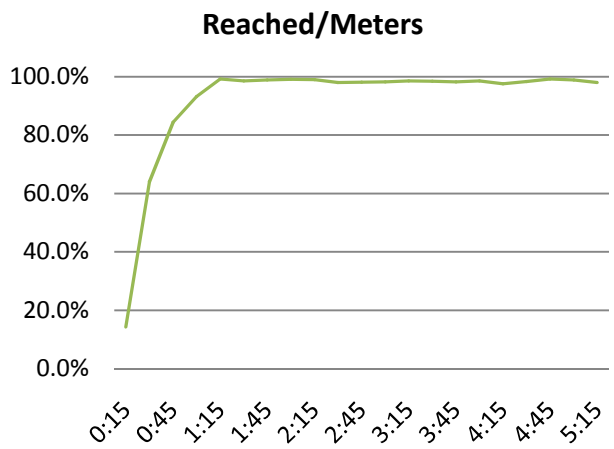


Fig. 4 Reachability to all smart meters after the gateway is turned on.

Our test consists of 2107 smart meters, 500 relay nodes, and a single gateway. The layout of the network can be seen in Fig. 3. The network has an isolated area of 185 nodes that need multiple repeaters to connect to the main network. In addition, in our simulation we found 6 isolated nodes that cannot reach any neighboring nodes. We use IEEE802.11g at the link layer.

In our simulations, the gateway is turned on at time 0, and every node also transmits HELLO messages every minute (proactive routing table buildup). There are no control messages for route repairs. Every meter transmits a 111 byte packet (a meter reading) every 15 minutes to the gateway. The time to live of data packets is 250 hops.

Over five hours of simulation time, there were 35251 unique data packets sent to the gateway. The total number of packets dropped in a single hop was 9198. In total, 680 data packets did not reach the destination. 211 data packets were dropped (one or more times) on their way to the gateway, but they were rerouted in the data forwarding plane to arrive at the gateway.

Fig. 4 shows the reachability of smart meters over time. Because smart meters only start transmitting data once the gateway appears in their routing tables there are several smart meters that are not reachable while the routing tables are initialized. As Fig. 5 shows, about 2.2% of the smart meters are between 51 and 60 hops away (on average) from the gateway, so HELLO messages propagating at a rate of one minute per hop tend to arrive one hour after the gateway started advertising its availability (HELLO messages are not retransmitted, so a lost packet is not sent again until a minute after, when a new routing table update is available). Fig. 6 shows that the worst delay experienced at a node is (on average) 1.2s, while most of the packets are delivered to the destination in less than 500ms.

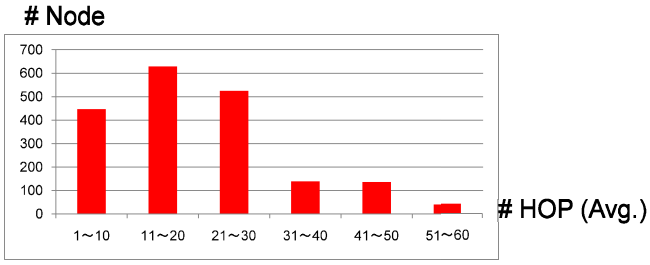


Fig. 5. Average number of hops in the AMI network.

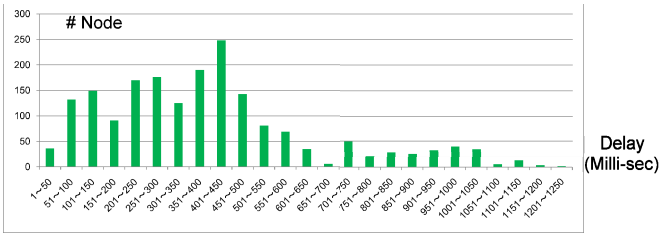


Fig. 6. Average data packet delay (per node).

B. *Experiments*

While we are unable to show the performance of our current large-scale deployments, we can show the performance of small-scale deployments within our installations.

1) *Indoor Deployment*

We used twelve nodes to form a wireless mesh network over two floors at a Fujitsu office in the Nagoya Prefecture. The center frequency for the link layer was 2.405GHz (IEEE 802.15.4, Channel 11) with transmission power at 3dBm, and there was interference from IEEE 802.11b.

Each floor had six nodes; one of them acting as a gateway for the network. Of these twelve nodes, one node was added half way through the experiment and one node in the critical path was taken out in the latter half of this experiment. As shown in Figure 7, these nodes were placed amongst office objects (no special placement).

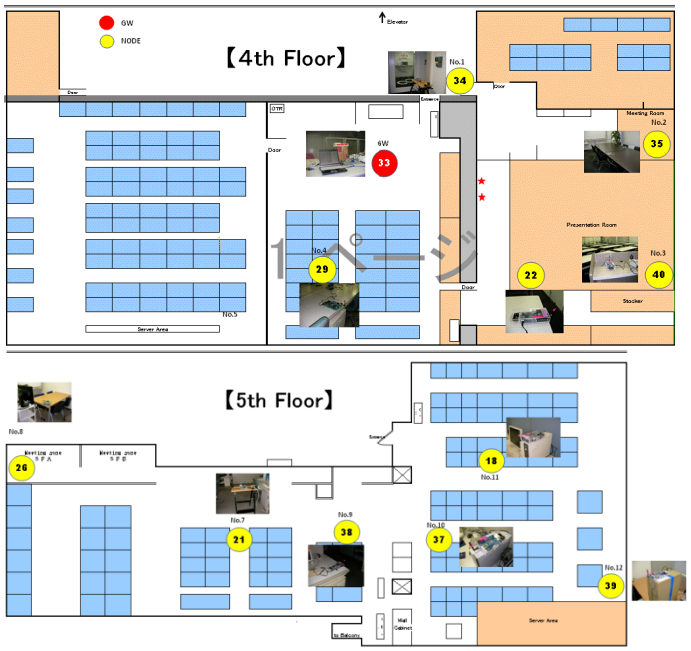


Figure 7 IEEE 802.15.4 Indoor Experiment: Red dot indicates the gateway and yellow dot indicates a node.

Each node sent roughly 90 packets during this period (roughly 13-15 seconds apart). Although the experiment went on for a longer period, the data presented here is based on data collected over 20-minutes. The majority of packets arrived with between 60 and 140 msec delay. Nearly 100% of packets reached the gateway once nodes formed the network. When a node that was being used by the majority of the routes was taken out, the reliability was reduced to 96%; however, within a minute, the reliability went back to 100% after a new route was learned via the data forwarding phase.

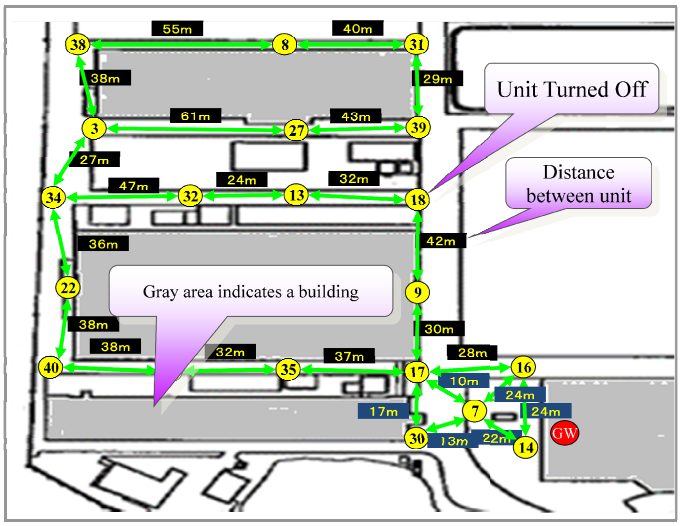


Fig. 8. Upper nodes were using the path 18-17 to route packets to the gateway. After node 18 is turned off, the data forwarding plane finds a route through nodes 39-27-3-22-40-35-17.

2) *Outdoor Experiment*

We also set up twenty nodes an outdoor environment at Fujitsu's factory in Kawasaki. Fig. 8 shows the layout of the experiment. The results were similar to those in the indoor experiment (See Fig. 9). While routing involved more hops, most packets arrived again with a maximum delay of 140ms.

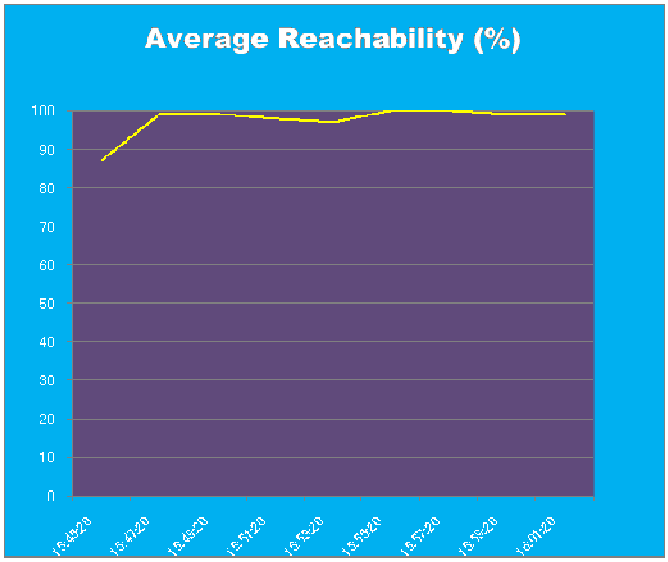


Fig. 9 Reliability results for the outdoor experiment. At time 3:54 node 18 was turned off.

V. Conclusions

We have presented a new routing and data forwarding paradigm for wireless sensor networks. We have shown that DADR can scale to provide flat connectivity to networks of thousands of nodes, with routes of more than 50 hops, while maintaining minimum control overhead. We have also shown that the data forwarding plane is able to reroute and converge to new routes when packets are lost due to intermittent or permanent link failure conditions.

This paper is an introduction to DADR. In the future we plan to evaluate and study in detail how the parameters affect performance. We also plan to continue improving the performance of DADR by exploring alternate responses to loop detection and packet drops, improve loop detection and FID table overhead, and present the results of our field trial in New Mexico. We also plan to evaluate the performance of DADR when compared to newly emerging wireless mesh networking standards such as the RPL routing protocols being currently drafted by the IETF ROLL working group.

References

[1] FUJITSU. (2010, May) Advanced Metering Infrastructure. [Online]. http://www.fujitsu.com/global/services/solutions/sensor-network/casestudies/ami/ Accessed 7/29/10

[2] P. Levis, A. Tavakoli, and S. Dawson-Haggerty, "Overview of existing routing protocols for low power and lossy networks," Internet Engineering Task Force, Internet Draft draft-ietf-roll-protocols-survey-07, 2009.

[3] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11b mesh network," in *ACM Sigcomm*, Sept. 2004, pp. 121-132.

[4] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, pp. 419-434, 2005.

[5] Y. Ganjali and N. McKeown, "Routing in a highly dynamic topology," in *IEEE SECON*, 2009, pp. 164-175.

[6] N. Garepalli, K. Gopalan, and P. Yang, "Control Message Reduction Techniques in Backward Learning Ad Hoc Routing Protocols," in *International Conference on Computer Communication Networks*, U.S. Virgin Islands, Aug. 2008.

[7] S.H. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali, "Routing Without Routes: The Backpressure Collection Protocol," in *ACM/IEEE International Conference on Information Processing in Sensor Networks*, Stockholm, 2010.

[8] C. Perkins, E. Belding-Royer, and S.R. Das. (2003, July) Ad hoc on-demand distance vector (AODV) routing. [Online] http://www.ietf.org/rfc/rfc3561.txt.

[9] T. Clausen and P. Jacquet. (2003, October) Optimized Link State Routing Protocol (OLSR). [Online]. http://www.ietf.org/rfc/rfc3626.txt.

[10] E. Gafni and D. Bertsekas, "Distributed Algorithms for Generating Loop-Free Routes in Networks with Frequently Changing Topology," *IEEE Transactions on Communications*, vol. 29, no. 1, pp. 11-18, Jan. 1981.

[11] V.D. Park and M.S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *IEEE Infocom*, 1997.

[12] J. Broch, D. Maltz, D. Johnson, Y-C Hu, and J. Jetcheva., "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *IEEE/ACM MobiCom*, 1998.

[13] Lee S.J. and Gerla M., "AODV-BR: backup routing in ad hoc networks," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2000.

[14] M. K. Marina and S. R. Das, "Ad hoc on-demand multipath distance vector routing," *Wireless Communications and Mobile Computing*, vol. 6, pp. 969-988, 2006.

[15] K. S. J. Pister and L. Doherty, "TSMP: Time synchronized mesh protocol," in *IASTED International Symposium. Distributed Sensor Networks (DSN)*, Orlando, FL, USA, 2008, pp. 391-398.

[16] S. Mueller, R.P. Tsang, and D. Ghosal, "Multipath routing in mobile ad hoc networks: issues and challenges," in *Performance Tools and Applications to Networked Systems*, 2004, pp. 209-234.

[17] J. Tsai and T. Moors, "A Review of Multipath Routing Protocols: From Wireless Ad Hoc to Mesh Networks," in *ACoRN Early Career Researcher Workshop on Wireless Multihop Networking*, 2006.

[18] R. Jain, A. Puri, and R. Sengupta, "Geographical routing using partial information for wireless ad hoc networks," *IEEE Personal Communications*, vol. 8, no. 1, pp. 48-57, Feb. 2001.

[19] J.C. Navas and T. Imielinski, "Geocastgeographic addressing and routing," in *ACM/IEEE international conference on mobile computing and networking*, 1997, pp. 66-76.

[20] K. Lakshminarayanan et al., "Achieving convergence-free routing using failure-carrying packets," in *ACM SIGCOMM*, Kyoto, Japan, 2007.

[21] S. Nelakuditi et. al., "Blacklist-Aided Forwarding in Static Multihop Wireless Networks," in *SECON*, 2005.