

On the Construction of Efficient Data Gathering Tree in Wireless Sensor Networks

Niwat Thepvilojanapong
Department of Information and
Communication Engineering
University of Tokyo, Japan
Email: wat@mcl.iis.u-tokyo.ac.jp

Yoshito Tobe
Department of Information Systems
and Multimedia Design
Tokyo Denki University, Japan
Email: yoshito@unl.im.dendai.ac.jp

Kaoru Sezaki
Center for Spatial
Information Science (CSIS)
University of Tokyo, Japan
Email: sezaki@iis.u-tokyo.ac.jp

Abstract—A wireless sensor network can be an effective tool for gathering data in a variety of environments. The data gathering process must be designed to conserve the limited resources of the sensors. In this paper, we propose Efficient Data Gathering (EDGE) protocol which satisfies such requirement because it avoids both flooding and periodic updating of routing packets. The tree created by EDGE will be reconstructed upon node failures or adding of new nodes. The simulation results compared to Directed Diffusion, DSR, AODV, and OLSR demonstrated that EDGE achieves a higher delivery ratio and shorter delay on various scenarios.

I. INTRODUCTION

Recent advances in MEMS-based sensor technology and low-power RF and OS design have enabled the development of relatively inexpensive and low-power wireless sensors [1]. A great number of these sensors can coordinate their communications to achieve a larger sensing task, both in urban environments and in inhospitable terrain [2]. For example, we can use such wireless sensor networks for environmental and habitat monitoring, tracking systems, failure detection, intrusion detection, etc. [3], [4]. However, the known limitations of sensor networks are resource constraints, including memory storage, computational power, communication bandwidth, and energy resources, and these challenge the design of a routing protocol that fulfills the requirements of the sensor networks.

Generally, a large number of sensors are deployed in the remote terrain. These sensors coordinate to establish a communication network, monitor specified tasks, and report sensed data periodically or spontaneously to the base station or sink node. When the existing sensors are out of order for any reason, they reorganize by themselves to repair the failed routes. Alternatively, the user may deploy additional sensors to alleviate the severe effect of many failed nodes, thereby enforcing the sensors to reconstruct the network to take advantage of the additional system resources. Thus, we consider a protocol that is used for gathering sensed data and also robust to the dynamic natures of sensor networks.

The remainder of the paper is organized as follows. Section II describes the problem statement. Section III enumerates *Efficient Data Gathering* (EDGE) protocol in details. Section IV evaluates the performance of EDGE in the simulated networks. Section V discusses related work, and we conclude by summarizing our findings and identifying future research opportunities in Sect. VI.

II. PROBLEM STATEMENT

We consider a network composed of a small number of sink nodes and numerous wireless sensor nodes randomly distributed in an area of interest. These sensors have limited processing power, storage, bandwidth, and energy, whereas the sinks have powerful resources. In particular, the sensors have omni-directional antennas and use RF to communicate. We assume that the sensors are not mobile nodes,

i.e., all sensors are fixed for the duration of their lifetime. However, the sensor network we consider has a dynamic nature, as described in Sect. I.

We design a protocol for a wireless sensor network whose communication pattern differs from conventional mobile ad hoc networks. Let S and B be a set of sensors and base stations, respectively. EDGE is a multipoint-to-point protocol for a source s and a destination d , where $s \in \{S\}$ and $d \in \{B\}$, namely, every sensor tries to report sensed data to the sink. Unlike EDGE, previous works [5]–[10] are point-to-point routing protocols, i.e., $s, d \in \{S, B\}$.

III. EDGE DESIGN

EDGE is based on a tree topology rooted at a sink node. Every sensor must be a member of the tree, i.e., an internal or leaf node, in order to communicate with the sink.

The design of EDGE has been driven by the following goals.

- **Simplicity and Scalability.** Because unconstrained scale is an inherent feature of a sensor network, and the sensors have limited computing capability and memory resources, we seek to minimize the number of operations performed and the states maintained at each sensor. In particular, each node does not maintain information of its neighboring nodes, and the path calculation is not based on the complex algorithms such as Dijkstra's or the Bellman-Ford algorithm.

- **Robustness.** EDGE provides self-organized mechanisms to handle the dynamic nature of sensor networks, i.e., joining and leaving scenario.

This section describes the details of EDGE which satisfies the above mentioned goals. The cycle of all sensors, except the sinks, start with the joining procedure (Sect. III-B) where the node attempts to attach itself to the tree.

A. Constructing the Tree

A base station initiates the tree construction by broadcasting a *child request* (CRQ) packet. A *Nonmember*, a node that is not attached to the tree, decides on its parent from the received CRQ packets by waiting for a short period (T_{crq}) to collect a number of candidates and choosing a node whose defined metric is the best. The candidates are kept in a *parental candidate* (PC) table, which maintains the pairs of candidate IDs and metrics. A *Member* that is an internal, or leaf, node of the tree also updates its PC table according to the incoming CRQ packets. Assume crq_time and $joined_time$ denote the time that a node first receives a CRQ packet and the time that a node joins the tree, respectively. The nonmember chooses a node whose crq_time is the minimum, and if many nodes have the same value of this metric, it will choose a node whose $joined_time$ is the minimum. Less time implies a node is nearer the base station, which results in the shorter path.

After choosing the parent, the nonmember sends a *child reply* (CRP) packet to the selected parent. Upon receiving the CRP packet, the parent confirms an acceptance of a new child by replying with a *child acceptance* (CAC) packet. If the child node does not receive the CAC packet within a period of T_{cac} , it will retransmit the CRP packet. This retransmission is performed N_{crp} times. After such attempts, it chooses a new parent from the PC table. If the PC table is empty, it will use the joining procedure (Sect. III-B) to discover a new parent. After receiving the CAC packet from the parent, the child node performs the same process as its parent, i.e., broadcasts a CRQ packet to discover its own child nodes. Note that the *joined.time* is the time when a node receives the CAC packet.

B. Adding the Nodes

A newly deployed sensor finds a parent by using a *joining procedure* as follows. The joining node broadcasts a *parent request* (PRQ) packet, thereby making the neighboring nodes aware of its existence. All members of the tree that hear this packet reply by unicasting a CRQ packet to the joining node. Then, the process follows the tree construction procedure, i.e., the joining node sends a CRP packet to the selected parent, and waits for a CAC packet as a confirmation of their relationship. If the joining node does not receive any CRQ packet after broadcasting the PRQ packet, it infers that no node is within its radio coverage or that all of its neighboring nodes are not members. In this case, it waits for an incoming CRQ packet after one of its neighbors has become a member. As an optional feature, the joining node can broadcast the PRQ packet periodically until it receives a CRQ packets.

C. Dealing with Node Failures

A *leaving procedure* described in this section is used to reconstruct the tree if an internal node has failed due to one of many possible reasons. For example, the battery of the node has discharged with time, or the node may be damaged by the harsh environment or by the enemy. The tree in EDGE is self-organized and it is reconstructed on demand, i.e., whenever the node has data to send. A detection of failed nodes relies on the underlying MAC layer protocol. If the acknowledgement on MAC layer does not arrive, the node infers that a communicating party has left from the network.

When an *orphaned node* is aware of the absence of its parent, it immediately switches to a new parent by choosing the most appropriate parent from its PC table. This can be done by sending a CRP packet to a newly selected parent and waiting for a CAC packet. If there is no any candidate in the PC table, it use the joining procedure as a newly deployed node. However, its child and grandchild nodes will not reply to this PRQ packet to prevent routing loop. Based on all of the simulations done in Sect. IV, two-hop information is sufficient for dealing with the routing loop problem. In the worst case where the orphaned node does not have any candidate in the PC table and there is no response to the PRQ packet, it sends a *parent query* (PQR) packet to its child nodes asking whether they have any candidate for a parent. The child nodes reply with a *parent reply* (PRP) packet containing such information. Then, the orphaned node randomly chooses the child that has at least one candidate as its new parent by sending a *reverse* (REV) packet to inform it of the reverse relationship, i.e., the selected child becomes a parent of the orphaned node, and switches to a new parent, chosen from its PC table. The relationships between other children of the orphaned node and the orphaned node do not change. If no child node has any candidate for a parent, the orphaned node still randomly chooses one child as a new parent by sending a REV packet and will let this

selected child find a new parent using the joining procedure. Note that the last scenario is quite rare but may occur in very sparse networks.

D. Data Gathering and Discussions

To collect the data, each node forwards its data and all of the received data to its parent. If it is not attached to the tree, it keeps the data in the buffer and sends them later. Each node in EDGE relies only on the knowledge of a parent, a grandparent, and a PC table which should be small enough to keep in the node itself. However, the PC table is an optional storage. The nodes can attach to a new parent faster with the help of the PC table but the information in the PC table may be stale, i.e., newly selected parent is also a left node. If the nodes always broadcast the PRQ packet to discover a new parent without relying on the PC table, it will get fresh information. Because the number of states required on each node is constant (a fix-sized PC table), and they are independent of node density and network size, EDGE is highly scalable. Every node in EDGE broadcasts only once to discover the route, thereby no propagation of routing packets. In other words, the routing packets are limited to one-hop neighboring nodes. Moreover, EDGE does not apply periodic updating that reduces the traffic load so much. Furthermore, geographical information is also unnecessary.

IV. PERFORMANCE EVALUATION

To evaluate the performance of EDGE, we use the *ns-2* [11] simulation tool to run a number of simulations. We then compare the performance with the following protocols: (1) a proactive ad hoc routing protocol, i.e., OLSR [6], (2) two reactive ad hoc routing protocols, i.e., DSR [7] and AODV [8], and (3) a communication protocol for sensor networks, i.e., Directed Diffusion (DD) [12]. We also ran the simulations on DSDV [5] which shows worse performance than AODV and DSR as also reported in [13]. Therefore, we do not include the results of DSDV in this paper.

A. Methodology

The *ns-2* simulator includes full simulation of the IEEE 802.11 physical and MAC layers. Our simulations used this MAC layer and assumed symmetric links. Using this MAC layer did not affect the evaluation because we needed to evaluate the network layer of five protocols. Actually, this MAC protocol is designed for ad hoc networks, therefore, ad hoc routing protocol may take the advantage from MAC layer used in the simulations. We randomly placed 50 sensors in a 200 m by 200 m square region. Each node had fixed radio coverage of 50 meters. All nodes had fixed positions, with no movement for the entire simulation. We used a constant bit rate (CBR) as our traffic sources. These CBR sources sent 64-byte data packets at the rate of 0.25, 0.5, 1, and 2 packets per second, i.e., 128, 256, 512, and 1,024 bps, respectively, while the bandwidth of the sensors was set to 19.2 kbps. The CBR agent was attached to a UDP agent, which in turn attached to the source node. For all of the simulations, the communication patterns were peer-to-peer and the starting time of each connection was randomly chosen. One node from each simulation was randomly chosen as a base station and the only destination for all traffic sources, while all the remaining nodes (49 nodes) were the source nodes (one flow per one source). Each simulation was run for 500 simulated seconds. Each result in the following graphs and tables represents an average of five runs with identical traffic model, but different randomly generated topologies. The parameters of EDGE used in the simulations were set as follows: $T_{crq} = 0.1$ sec., $T_{cac} = 0.3$ sec., and $N_{crp} = 2$.

B. Performance Metrics

We studied our protocol using the following three metrics.

- *Packet delivery ratio* (PDR): The ratio between the number of data packets received by the destination and the number of data packets sent by the source.

- *Average delay*: Average end-to-end delay observed between transmitting a data packet and receiving it at the destination.

- *Average path length*: Average number of hops a data packet took to reach the destination.

PDR is important because it shows the loss rate seen by the transport protocols. It also affects the maximum throughput that the network can support. This throughput can be investigated by increasing transmission rate. Therefore, this metric characterizes both the completeness and correctness of the protocol.

Average delay is an important metric for comparison as it measures the quality of path determined by the routing algorithm. Protocols that send a large number of routing packets can also increase the probability of packet collisions and may delay data packets by queuing them in the buffer.

Average path length measures the ability of the protocol to use network resources efficiently by selecting the shortest path from the source to the destination. Shorter path means fewer transmissions, which in turn implies lower dissipated energy.

C. Performance Comparison in Static Networks

The comparisons of five protocols for three metrics are illustrated in Fig. 1. EDGE is the only protocol that can deliver nearly 100% of the originated packets for all offered loads (Fig. 1(a)). PDRs of DD and DSR are similar with those of EDGE at light load but it can deliver only 89% and 44% at 1,024-bps load, respectively. PDRs of AODV and OLSR start to drop at 512-bps load and both can deliver only a half of originated packets at the extreme load. With high offered load, most of the data packets drop at the nodes around the sink due to high congestion. Because the sink in DD *periodically floods* interest messages to the entire network, the probability of channel contention increases, resulting in dropped packets when the buffer is overflow. Although OLSR uses multipoint relaying technique to suppress some redundant floodings, every node is still required to *periodically* exchange control messages which incurs a large volume of traffics. The congestions due to high offered load also force a node to find a new available route. When a node in DD detects failed or degraded paths, all nodes downstream of the lossy link (all nodes towards the sink) will apply the reinforcement rules, i.e., flooding the interest messages. For four left protocols, when a node cannot deliver a packet to the next hop due to congestion, it implies the absence of the next-hop node. In such case, DSR and AODV will flood a route request (RREQ) packet to discover a new route. These control packets incurred by DD, OLSR, DSR, and AODV lead to a chaos of the network, which in turn incurs more congestions and more dropped packets. In contrast, EDGE discovers a new route by broadcasting which is limited to only one hop. Thereby, EDGE is more robust to higher offered load than other four protocols.

EDGE has shorter delay than other four protocols for all offered loads (Fig. 1(b) and 1(c)) because it encounters less contention as discussed above, and it also takes shorter paths (Fig. 1(d)). The path length of AODV and OLSR decreases as the load is increased because most of dropped packets are the packets destined to a distant destination. DD uses the longest path because it selects an empirically low delay path which may not be the shortest path. OLSR takes the shortest path (only 0.05 hop shorter than EDGE) because any link state protocols know the topology of the entire network so that it can

TABLE I
PERFORMANCE COMPARISON IN JOINING SCENARIOS.

Load (bps)	PDR		Delay (ms)		Path length	
	512	1024	512	1024	512	1024
AODV	0.8753	0.36	129	912	4.02	2.93
DSR	0.9988	0.35	17	3737	3.24	3.40
OLSR	0.9605	0.48	44	1277	3.03	2.46
DD	0.9649	0.82	87	1345	5.18	4.58
EDGE	0.9980	0.98	13	516	3.33	3.30

TABLE II
PERFORMANCE COMPARISON IN LEAVING SCENARIOS.

Load (bps)	PDR		Delay (ms)		Path length	
	512	1024	512	1024	512	1024
AODV	0.9656	0.44	44	720	4.00	2.91
DSR	0.9435	0.33	279	4830	3.35	3.29
OLSR	0.9619	0.42	37	1261	3.10	2.37
DD	0.7794	0.65	44	394	4.19	3.82
EDGE	0.9999	0.98	13	360	3.28	3.24

calculate an optimal path. However, this advantage of OLSR comes with a large amount of control traffics. We conclude that the tree created by EDGE is nearly optimal in the viewpoint of path length, and it is quite efficient when considering PDR and delay.

D. Performance Comparison in Dynamic Networks

To simulate the joining scenarios, 10 nodes were randomly deployed halfway through the simulations, with 50 nodes being deployed from the beginning of the simulation. For the leaving scenarios, we deployed 60 nodes at the beginning of the simulations and used an energy model that provided sufficient energy for 50 nodes, making the battery of 10 random nodes discharge at approximately the half-way point of the simulation. We chose heavy offered loads, i.e., 512- and 1,024-bps load, to make the situations fairly challenging for the protocols. These experiments were used to evaluate the robustness and resilience of the protocols against network dynamics. Note that 10 nodes means approximately 20% of the nodes in the network have a dynamic nature.

Table I presents the results of the joining scenarios. With 512-bps load, PDRs of EDGE, DSR, and OLSR are consistent with those of static networks, while PDRs of DD and AODV drop 3% and 6%, respectively. When we offer 1,024-bps load, EDGE still achieves an excellent PDR, i.e., 98%, whereas PDRs of other four protocols are lower than those of static scenarios because additional 10 nodes increase the total traffic loads, which incur more congestion in the networks. The average delays of all protocols also increase as a result of additional data traffics. The average path lengths slightly decrease as the node density increases. The exception is DD because it chooses a path according to the delay.

Table II shows the results of the leaving scenarios. EDGE still has a superior PDR when some nodes left the networks, whereas PDRs of other protocols drop when comparing to those of static scenarios. PDRs of DD decrease more than 20% because local repair is done by all nodes downstream of the left node. Instead of using all downstream nodes, only one upstream node, which is employed by other four protocols, is better to initiate the local repair as proved by the simulations. Consequently, the delay of DD at 1,024-bps load is shorter than those of static case due to highly dropped packets destined to a distant destination. The path length also affirms this fact, i.e., the packets in DD take shorter path compared to the results of static network. The average delays of all cases, except DD at 1,024-bps load, are higher than those of static scenarios because every

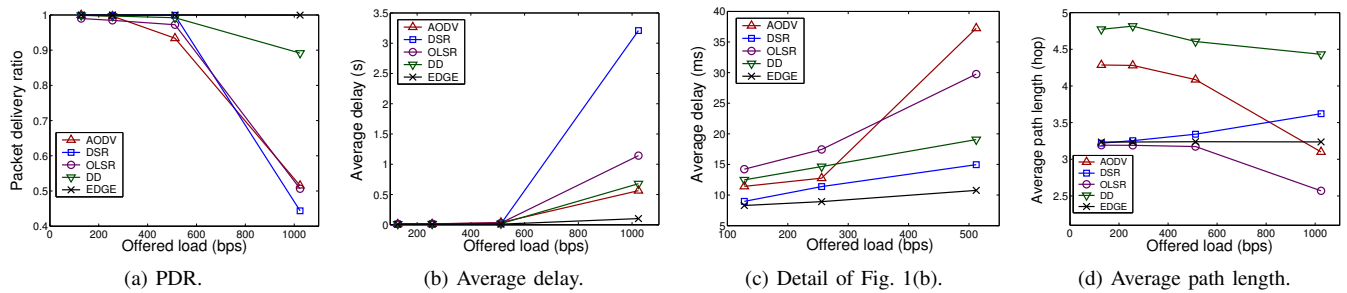


Fig. 1. Performance comparison in static networks.

protocol must take an amount of time in detecting a left node and finding a new path, while the average path lengths are a little shorter than static networks due to higher node density. We conclude that EDGE still achieves a good performance in dynamic sensor networks.

V. RELATED WORK

A number of routing protocols [14] has been developed for mobile ad hoc network (MANET) which differs from wireless sensor network (WSN) in an issue of node mobility. The proactive protocols (*e.g.*, DSDV [5], OLSR [6]) must periodically update the states to follow current physical topology. Although, OLSR uses multipoint relaying technique to economically flood control messages, it still employs periodic exchange of messages which is a main cause of congestion. Moreover, each node must maintain the routing table for the entire network which is not possible in limited-resource sensors. Despite the fact that the reactive protocols (*e.g.*, DSR [7], AODV [8]) discover the route on-demand, flooding still incurs congestion as shown in our simulations, and it also takes time to discover a new available route. Therefore, we propose an alternative which is more light-weight and tailored to a specific communication required in WSNs. Every node in EDGE broadcasts only once to discover a route which obviously incurs less overhead than flooding and periodic updating.

Location-based routings (LAR [10], GPSR [9], *etc.*) can also apply well to WSNs. Unfortunately, such approaches may not implement into many sensor networks because each sensor requires exact geographic location. Current methods of determining geographic location [15] consume much energy and may not be possible in many sensor network scenarios. Hence, we do not rely on location information.

Directed Diffusion [12] is a data-centric routing protocol based on the name of data. The sinks draw interesting information by periodically flooding interests and setting up gradients within the network. These periodic interests waste the resources and disturb the delivery of data packets as done by periodic updating in proactive protocols. Moreover, Directed diffusion requires location information, which may not be possible as described above.

VI. CONCLUSIONS

This paper demonstrates that EDGE efficiently gathers data across multi-hop, wireless sensor networks while maintaining constant local state and making only local decisions. EDGE also provides a solution against dynamic natures of sensor networks. We have examined the effects of EDGE in terms of packet delivery ratio, end-to-end delay, and path length. The results show that EDGE achieves notably higher delivery ratios, shorter delays, and comparable path lengths compared to the existing protocols including proactive and reactive protocols for MANET and a routing protocol for WSN. Besides, it is more robust against high offered loads than the existing solutions. One

application in sensor networks that incurs high traffic load is structure health monitoring (SHM) [4]. Nonetheless, the actual traffic rate in any applications is greater than the data generation rate of a node due to forwarded traffic, especially around the sink.

We plan to study EDGE in many issues. We will study the improvement of performance when employing an aggregation scheme in EDGE. The aggregation is an important technique in sensor networks because it greatly reduces the number of data packets. We plan to study reliability as a factor in the future. This will be an end-to-end reliability employed in the transport layer. We also plan to implement our protocol in Mica Mote to study its performance in a real world environment.

REFERENCES

- [1] J. Hill and D. Culler, "Mica: a wireless platform for deeply embedded networks," *IEEE Micro*, vol. 22, no. 6, pp. 12–24, 2002.
- [2] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proc. of ACM MOBICOM*, Seattle, Washington, Aug. 1999, pp. 263–270.
- [3] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proc. of ACM WSNA*, Atlanta, GA, Sept. 2002, pp. 88–97.
- [4] K. Mechitov, W. Kim, G. Agha, and T. Nagayama, "High-frequency distributed sensing for structure monitoring," in *Proc. of INSS*, Tokyo, Japan, June 2004, pp. 101–104.
- [5] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proc. of ACM SIGCOMM*, London, UK, Sept. 1994, pp. 234–244.
- [6] T. Clausen and P. Jacquet, "Optimized link state routing protocol (OLSR)," IETF, RFC 3626, Oct. 2003.
- [7] D. B. Johnson, D. A. Maltz, and J. Broch, "DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks," in *Ad Hoc Networking*, C. E. Perkins, Ed. Addison-Wesley, 2001, pp. 139–172.
- [8] C. E. Perkins, E. M. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," IETF, RFC 3561, July 2003.
- [9] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proc. of MOBICOM*, Aug. 2000, pp. 243–254.
- [10] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," in *Proc. of ACM MOBICOM*, Oct. 1998, pp. 66–75.
- [11] "Network simulator – ns-2," <http://www.isi.edu/nsnam/ns/>.
- [12] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proc. of ACM MOBICOM*, Aug. 2000, pp. 56–67.
- [13] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proc. of MOBICOM*, Dallas, Texas, Oct. 1998, pp. 85–97.
- [14] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Ad Hoc Networks*, vol. 2, no. 1, pp. 1–22, Jan. 2004.
- [15] L. Doherty, K. S. J. Pister, and L. E. Ghaoui, "Convex optimization methods for sensor node position estimation," in *Proc. of IEEE INFOCOM*, Anchorage, Alaska, Apr. 2001, pp. 1655–1663.