Energy Efficient and Reliable Data Routing Approach For WSN

Tintu Thampi P
M.Tech, Department of CSE
AMC Engineering College
Bangalore, India.
tintuthampi2004@gmail.com

Abstract **-Wireless Sensor Networks (WSNs) are usually large collection of sensor nodes. Sensor nodes are energy-constrained devices and the energy consumption is generally associated with the amount of gathered data, since communication is often the most expensive activity in terms of energy. Routing plays an important role in the data gathering process. Recently Data aggregation has emerged as an effective technique for saving energy in WSNs. Due to the inherent redundancy in raw data gathered by the sensor nodes, In-Networking aggregation can often be used to decrease the communication cost by eliminating redundancy and forwarding only smaller aggregated information. Since minimal communication leads directly to energy savings, which extends the network lifetime, In-Network Data aggregation is a key technology to be supported by WSNs. Data aggregation aware routing protocols present some desirable characteristics such as: reduced number of messages for setting up a routing tree, maximized number of overlapping routes, high aggregation rate, and also a reliable data transmission. This work proposes a novel Energy Conserving In-Network Data Aggregation Routing algorithm for WSNs, which is referred to as ECIDA. In this case, redundant data can be aggregated at intermediate nodes reducing the size and number of exchanged messages and, thus, decreasing communication costs and energy consumption. This maximizes network lifetime.**

IndexTerms—: Routing protocol, in-network aggregation, wireless sensor networks

I INTRODUCTION

Recent years have witnessed an increasing interest in using wireless sensor networks (WSNs) in many applications. A Wireless Sensor Network consists of spatially distributed autonomous devices that cooperatively sense physical or environmental conditions, such as temperature, sound, vibration, pressure, motion, or pollutants at different locations [1], [2]. WSNs have been used in applications such as environmental monitoring, homeland security, critical infrastructure systems, communications, manufacturing, and many other applications that can be critical to save lives and assets [3], [4], [5]. In these applications, tiny sensors are deployed and left unattended to continuously report parameters such as temperature, pressure, humidity, light, and chemical activity. Reports transmitted by these sensors are collected by observers (e.g., base stations). The dense deployment and unattended nature of WSNs makes it quite difficult to recharge node batteries. Therefore, energy efficiency is a major design goal in these networks. Several WSN applications require only an aggregate value to be reported to the observer.

Sensor nodes are energy-constrained devices and the energy consumption is generally associated with the amount of gathered data, since communication is often the most expensive activity in terms of energy. For that reason, algorithms and protocols designed for WSNs should consider the energy consumption in their conception [6], [7], [8]. Moreover, WSNs are data-driven networks that usually produce a large amount of information that needs to be routed, often in a multi hop fashion, toward a sink node, which works as a gateway to a monitoring center (Fig. 1).

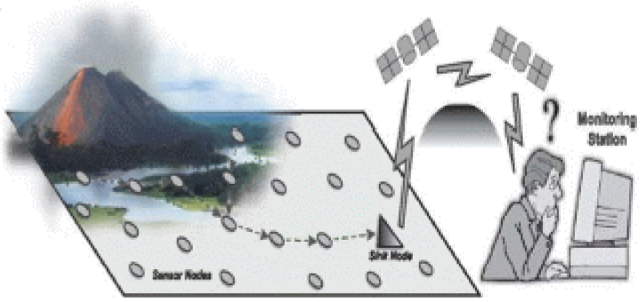

Fig. 1 Data aggregation aware routing, a key algorithm for data-driven WSNs.

Given this scenario, routing plays an important role in the data gathering process. A possible strategy to optimize the routing task is to use the available processing capacity provided by the intermediate sensor nodes along the routing paths. This is known as data-centric routing or In-network data aggregation. For more efficient and effective data gathering with a minimum use of the limited resources, sensor nodes should be configured to smartly report data by making local decisions [9] .For this, data aggregation is an effective technique for saving energy in WSNs. Due to the inherent redundancy in raw data gathered by the sensor nodes, in-networking aggregation can often be used to decrease the communication cost by eliminating redundancy and forwarding only smaller aggregated information. Since minimal communication leads directly to energy savings, which extends the network lifetime, in-network data aggregation is a key technology to be supported by WSNs.

NCITSF' 14

One of the main challenges in routing algorithms for WSNs is how to guarantee the delivery of the sensed data even in the presence of nodes failures and interruptions in communications. These failures become even more critical when data aggregation is performed along the routing paths since packets with aggregated data contain information from various sources and, whenever one of these packets is lost a considerable amount of information will also be lost. In the context of WSN, data aggregation aware routing protocols should present some desirable characteristics such as: a reduced number of messages for setting up a routing tree, maximized number of overlapping routes, high aggregation rate, and also a reliable data transmission. In order to overcome these challenges, in this work, we propose a novel Data Routing algorithm for In-Network Aggregation for WSNs, which we refer to as ECIDA algorithm. Our proposed algorithm was conceived to maximize information fusion along the communication route in reliable way, through a fault-tolerant routing mechanism.

II. RELATED WORK

In-network data aggregation in WSN refers to the different ways intermediate nodes forward data packets toward the sink node while combining the data gathered from different source nodes. A key component for In-network data aggregation is the design of a data aggregation aware routing protocol. Data aggregation requires a forwarding paradigm that is different from the classic routing, which typically involves the shortest path "in relation to some specific metric" to forward data toward the sink node. Differently from the classic approach in data aggregation aware routing algorithms, nodes route packets based on their content and choose the next hop that maximizes the overlap of routes in order to promote in-network data aggregation. A key aspect of In-network data aggregation is the synchronization of data transmission among the nodes. In these algorithms, a node usually does not send data as soon as it is available since waiting for data from neighboring nodes may lead to better data aggregation opportunities. This in turn, will improve the performance of the algorithm and save energy. Three main timing strategies are found [10], [11].

- *Periodic simple aggregation*

Requires each node to wait for a predefined period of time while aggregating all received data packet and, then, forward a single packet with the result of aggregation.

- *Periodic per-hop aggregation*

Quite similar to the previous approach, but the aggregated data packet is transmitted as soon as the node hears from all of its children. This approach requires each node to know the number of its children. In addition, a timeout may be used for the case of some children's packet being lost.

- *Periodic per-hop adjusted aggregation*

Adjusts the transmission time of a node according to this node's position in the gathering tree. Note that the choice of the timing strategy strongly affects the design of the routing protocol as well as its performance.

In-network data aggregation plays an important role in energy constrained WSNs since data correlation is exploited and aggregation is performed at intermediate nodes reducing size and the number of messages exchanged across the network. In data gathering-based applications, a considerable number of communication packets can be reduced by In-network aggregation, resulting in a longer network lifetime. In this case, the optimal aggregation problem is equivalent to a Steiner tree problem [12], [13].

- Definition 1 (Steiner Tree): Given a network represented by a graph G = (V, E), where V = {v1, v2, . . . , vn} is the set of sensor nodes, E is the set of edges representing the connections among the nodes, i.e., <i, j> ε E iff vi reaches vj , and w(e) is the cost of edge e, a minimal cost tree is to be built that spans all source nodes S = {s1, s2, . . . , sm}, S V , and the sink node s_0. The cost of the resulting Steiner tree (W) is the sum of the costs of its edges. This problem is a well-known NP-hard problem.

However, these solutions are not appropriate for resource-constrained networks, such as WSNs, since their distributed implementation require a large number of messages exchange when setting up the routing tree and, consequently, resulting in high energy consumption. Thus, various algorithms have been proposed to provide data aggregation during the routing in WSNs. Some of them are *tree-based* algorithms and try to solve some variation of the Steiner tree problem; others are *cluster-based* algorithms while others are simply *structure-less*.

A. Tree-Based Approaches

In these protocols, a tree structure is constructed first and then used later to either route the gathered data or respond to queries sent by the sink node. Aggregation is performed during the routing when two or more data packets arrive at the same node of the tree. This node then aggregates all received data with its own data and forwards only one packet to its neighbor that is lower in the tree. However, this approach has some drawbacks. For instance, when a packet is lost at a certain level of the tree (e.g., due to channel impairments), data from the whole sub tree will be lost as well. Thus, tree-based approaches require a mechanism to reliably forward the aggregated data.

In most cases, tree-based protocols build a traditional short-est path routing tree. For instance, the Shortest Path Tree (SPT) algorithm [12] uses a very simple strategy to build a routing tree in a distributed fashion. In this approach, every node that detects an event reports its collected information by using a shortest path to the sink node. Information fusion occurs whenever paths overlap (opportunistic information fusion). The Directed Diffusion algorithm is one of the earliest solutions to also propose attribute-based routing. In these cases, data can be opportunistically aggregated when they meet at any intermediate node. Based on Directed Diffusion, the Greedy Incremental Tree (GIT) approach was proposed. The GIT algorithm establishes an energy-efficient path and greedily attaches other sources onto the established path. In the GIT strategy, when the first event is detected, nodes send their information as in the SPT algorithm and, for every new event, the information is routed using the shortest path to the current tree. There is a new aggregation point every time a new branch is created. Some practical issues make GIT not appropriate in WSNs. For example, each node needs to know the shortest path to all nodes in the network. The communication cost to create this infrastructure is $O(n^2)$, where n is the number of nodes. Furthermore, the space needed to store this information at each node is O(D n), where D is the number of hops in the shortest path connecting the farthest node v ε V to the sink node (network diameter). After the initial phase the algorithm needs O(m n) messages to build the routing tree, where m is the number of source nodes.

Data aggregation is implemented in a real world test-bed and the Tiny Aggregation Service (TAG) framework is introduced. TAG uses a shortest path tree, and proposes improvements, such as snooping-

NCITSF'14

based and hypothesis testing based optimizations, dynamic parent switching, and the use of a child cache to estimate data loss. In the TAG algorithm, parents notify their children about the waiting time for gathering all the data before transmitting it so the sleeping schedule of the nodes can be adjusted accordingly. However, like most of the cited tree-based data aggregation aware routing algorithms, the TAG algorithm needs a considerable number of message exchange to construct and maintain the tree.

B. Cluster-Based Approaches

Similarly to the tree-based approaches, cluster-based schemes [14], [15] also consist of a hierarchical organization of the network. However, in these approaches, nodes are divided into clusters. Moreover, special nodes, referred to as cluster-heads, are elected to aggregate data locally and forward the result of such aggregation to the sink node.

In the Low Energy Adaptive Clustering Hierarchy(LEACH) algorithm clustered structures are exploited to perform data aggregation. In this algorithm, cluster heads can act as aggregation points and they communicate directly to the sink node. In order to evenly distribute energy consumption among all nodes, cluster-heads are randomly elected in each round. LEACH-based algorithms assume that the sink can be reached by any node in only one hop, which limits the size of the network for which such protocols can be used.

The Information Fusion-based Role Assignment (InFRA) algorithm builds a cluster for each event including only those nodes that were able to detect it. Then, cluster-heads merge the data within the cluster and send the result toward the sink node. The InFRA algorithm aims at building the shortest path tree that maximizes information fusion. Thus, once clusters are formed, cluster-heads choose the shortest path to the sink node that also maximizes information fusion by using the aggregated coordinators distance. A disadvantage of the InFRA algorithm is that for each new event that arises in the network, the information about the event must be flooded throughout the network to inform other nodes about its occurrence and to update the aggregated coordinators-distance. This procedure increases the communication cost of the algorithm and, thus, limits its scalability.

Our proposed the ECIDA algorithm, presented in detail is also a cluster-based approach. In our algorithm, for each new event, it is performed the clustering of the nodes that detected the same event as well as the election of the cluster-head. After that, routes are created by selecting nodes in the shortest path, to the nearest node that is part of an existing routing infrastructure, where this node will be an aggregation point. Our ECIDA routing infrastructure tends to maximize the aggregation points and to use fewer control packets to build the routing tree. Also, differently from the InFRA algorithm, ECIDA does not flood a message to the whole network whenever a new

C. Structure-Less Approaches

Few algorithms for routing aware of data aggregation have been proposed that use a structure-less approach. The Data-Aware Anycast (DAA) algorithm, a structure-less data aggregation algorithm, uses anycast to forward packets to one-hop neighbors that have packets for aggregation. It has mechanisms for increasing the chance of packets meeting at the same node (spatial aggregation) and at the same time (temporal aggregation). Since the approach does not guarantee aggregation of all packets, the cost of transmitting packets with no aggregation increases in larger networks.

III. ENERGY CONSERVING IN-NETWORK DATA AGGREGATION FOR WSNs

The main goal of our proposed the ECIDA algorithm is to build a routing tree with the shortest paths that connect all source nodes to the sink while maximizing data aggregation. The proposed algorithm considers the following roles in the routing infrastructure creation:

Collaborator. A node that detects an event and reports the gathered data to a coordinator node.

Coordinator. A node that also detects an event and is responsible for gathering all the gathered data sent by collaborator nodes, aggregating them and sending the result toward the sink node.

Sink. A node interested in receiving data from a set of coordinator and collaborator nodes.

Relay. A node that forwards data toward the sink.

The ECIDA algorithm can be divided into three phases. In Phase 1, the hop tree from the sensor nodes to the sink node is built. In this phase, the sink node starts building the hop tree that will be used by Coordinators or data forwarding purposes. Phase 2 consists of cluster formation and cluster-head election among the nodes that detected the occurrence of a new event in the network. Finally, Phase 3 is responsible for both setting up a new route for the reliable delivering of packets and updating.

A. Phase 1: Building the Hop Tree

In this phase, the distance from the sink to each node is computed in hops. This phase is started by the sink node sending, by means of a flooding, the Hop Configuration Message (HCM) to all network nodes. The HCM message contains two fields: ID and HopToTree, where ID is node identifier that started or retransmitted the HCM message and HopToTree is the distance, in hops, by which an HCM message has passed.

The HopToTree value is started with value 1 at the sink, which forwards it to its neighbors (at the beginning, all nodes set the HopToTree as infinity). Each node, upon receiving the message HCM, verifies if the value of HopToTree in the HCM message is less than the value of HopToTree that it has stored and if the value of First Sending is true, as shown in Algorithm 1 - Line 3. If that condition is true then the node updates the value of the Next Hop variable with the value of the field ID of message HCM, as well as the value of the HopToTree variable, and the values in the fields ID and HopToTree of the HCM message. The node also relays the HCM message, as shown in Algorithm 1 - Line 8. Otherwise, if that condition is false, which means that the node already received the HCM by a shortest distance, then the node discards the received HCM message, as shown in Algorithm 1 - Line 12. The steps described above occur repeatedly until the whole network is configured.

Before the first event takes place, there is no established route and the HopToTree variable stores the smallest dis-tance to the sink. On the first event occurrence, HopToTree will still be the smallest distance; however, a new route will be established. After the first event, the HopToTree stores the smaller of two values: the distance to the sink or the distance to the closest already established route.

NCITSF`14

Algorithm 1: Hop Tree Configuration Phase

```
1  Node sink sends a broadcast of HCM messageswith the value
   of HopToTree = 1;
   // R_u is the set of nodes that received the message
      HCM
2  foreach u ∈ R_u do
3      if HopToTree(u) > HopToTree(HCM) and
       FirstSending(u) then
4          NextHop_u ← ID_HCM;
5          HopToTree_u ← HopToTree_HCM + 1 ;
           // Node u updates the value of the ID field in
              the message HCM
6          ID_HCM ← ID_u ;
           // Node u updates the value of the HopToTree
              field in the message HCM
7          HopToTree_HCM ← HopToTree_u ;
8          Node u sends a broadcast message of the HCM with
           the new values;
9          FirstSending_u ← false ;
10     end
11     else
12         Node u discards the received message HCM;
13     end
14 end
```

Algorithm 2: Cluster formation and leader election

```
1  Input: S // Set of nodes that detected the event
2  Output: u // A node of the set S is elected leader of
       the group
3  foreach u ∈ S do
4      role_u ← coordinator;
       // Node u sends message MCC in broadcast
5      Announcement of event detection ;
       // N_u is the set of neighbors of node u ∈ S
6      foreach w ∈ N_u do
7          if HopToTree(u) > HopToTree(w) then
8              role_u ← collaborator ;
9              Node u retransmits the MCC message received
               from node w ;
10         end
11         else if HopToTree(u) = HopToTree(w) ∧
           ID(u) > ID(w) then
12             role_u ← collaborator ;
13             Node u retransmits the MCC message received
               from node w;
14         end
15         else
16             Node u discards the MCC message received from
               w;
17         end
18     end
19 end
```

B. Cluster Formation

When an event is detected by one or more nodes, the leader election algorithm starts and sensing nodes will be running for leadership (group coordinator); this process is described in Algorithm 2. For this election, all sensing nodes are eligible. If this is the first event, the leader node will be the one that is closest to the sink node. Otherwise, the leader will be the node that is closest to an already established route (Algorithm 2, Lines 7 to 9). In the case of a tie, i.e., two or more concurrent nodes have the same distance in hops to the sink (or to an established route), the node with the smallest ID maintains eligibility, as shown in Lines 11 to 13 of Algorithm 2. Another possibility is to use the energy level as a tiebreak criterion. At the end of the election algorithm only one node in the group will be declared as the leader (Coordinator). The remaining nodes that detected the same event will be the Collab-orators. The Coordinator gathers the information collected by the Collaborators and sends them to the Sink. A key advantage of this algorithm is that all of the information gathered by the nodes sensing the same event will be aggregated at a single node (the Coordinator), which is more efficient than other aggregation mechanisms (e.g.,opportunistic aggregation).

C. Routing Formation and Hop Tree Updates

The elected group leader, as described in Algorithm 2, starts establishing the new route for the event dissemination. For that, the Coordinator sends a route establishment message to its NextHop node. When the NextHop node receives a route establishment message, it retransmits the message to its NextHop and starts the hop tree updating process. These steps are repeated until either the sink is reached or a node that is part of an already established route is found. The routes are created by choosing the best neighbor at each hop. The choices for the best neighbor are twofold: 1) when the first event occurs, the node that leads to the shortest path to the sink is chosen (Fig. 2a); and 2) after the occurrence of subsequent events, the best neighbor is the one that leads to the closest node that is already part of an established route (Fig. 2c). This process tends to increase the aggregation points, ensuring that they occur as close as possible to the events.

The resulting route is a tree that connects the Coordinator nodes to the sink. When the route is established, the hop tree updating phase is started. The main goal of this phase is to update the HopToTree value of all nodes so they can take into consideration the newly established route. This is done by the new relay nodes that are part of an established route. These nodes send an HCM message (by means of a controlled flooding) for the hop updating (Fig. 2b). The whole cost of this process is the same of a flooding, i.e., each node will send only one packet. Outside the cluster, aggregation is performed by the relay nodes when two or more events overlap along routing (cluster outside aggregation).

D. Route Repair Mechanism

The route created to send the data toward the sink node is unique and efficient since it maximizes the points of aggregation and, consequently, the information fusion. However, because this route is unique, any failure in one of its nodes will cause disruption, preventing the delivery of several gathered event data. Possible causes of failure include low energy, physical destruction, and communication blockage.

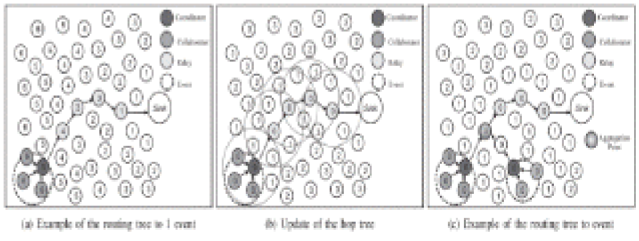

Fig. 2. Example of establishing new routes and updating the hop tree.

Our ECIDA algorithm offers a piggybacked, ACK-based, route repair mechanism, which consists of two parts: failure detection at the NextHop node, and selection of a new NextHop. When a relay node needs to forward data to its NextHop node, it simply sends the data packet, sets a timeout, and waits for the retransmission of the data packet by its NextHop. This re-transmission is also considered an ACK message. If the sender receives its ACK from the NextHop node, it can infer that the NextHop node is alive and, for now, everything is ok. However, if the sender node does not receive the ACK from the NextHop node within the predetermined timeout, it considers this node as offline and another one should be selected as the new NextHop node. For this, the sender chooses the neighbor with the lowest hop-to-tree level to be its new NextHop; in case of a tie, it chooses the neighbor with the highest energy level. After that, the sender updates its routing table to facilitate the forwarding of subsequent packets.

IV. PERFORMANCE EVALUATION

In this section, we evaluate the proposed ECIDA algorithm and compare its performance to two other known routing protocols: the InFRA and SPT algorithms. These two algorithms were chosen for being well known in the literature and have the same goals that the proposed ECIDA algorithm. Table 2 shows the basic characteristics of SPT, InFRA, and ECIDA algorithms. We evaluate the ECIDA performance under the following metrics:

1. Packet delivery rate.
2. Control overhead.
3. Efficiency (packets per processed data).
4. Routing tree cost.
5. Loss of raw data.
6. Loss of aggregated data.
7. Transmissions number.

A. Methodology

The performance evaluation is achieved through simulations using the SinalGo version v.0.75.3 network simulator. In all results, curves represent average values, while error bars represent confidence intervals for 95 percent of confidence from 33 different instances (seeds). The default simulation parameters are presented in Table 1. For each simulation set, a parameter shown in Table 1 will be varied as described in the evaluated scenario. The first event starts at time 1,000 s and all other events start at a uniformly distributed random time between the interval [1,000, 3,000] seconds. Also, these events occur at random positions. The network density is considered as the relation $n\pi r_c^2/A$, where n is number of nodes, r_c is the communication radius, and A is the area of the sensor field. For each simulation in which the number of nodes is varied, the sensor field dimension is adjusted accordingly in order to maintain the node density at the same value. Sensor nodes are uniformly and randomly deployed. To provide a lower bound to the packet transmissions, an aggregation function was used that receives p data packets and sends only a fixed size merged packet. However, any other aggregation function can be used to take advantage of ECIDA features. This function is performed at the aggregation points whenever these nodes send a packet. The evaluated algorithms used periodic simple aggregation strategy in which the aggregator nodes transmit periodically the received and aggregated information.

The following metrics were used for the performance evaluation:

Data packet delivery rate. Number of packets that reach the sink node. This metric indicates the quality of the routing tree built by the algorithms—the lower the packet delivery rate, the greater the aggregation rate of the built tree

TABLE 1
Simulation Parameters

Sink Node	1 (Top left)
Network Size	1024
Communication Radius	80(m)
# of events	3
Event Radius(m)	50
Event Duration(hrs)	3
Loss Probability %	0
Simulation Duration(hrs)	4
Notification Interval(sec)	60
Sensor Field(m)	700x700

TABLE 2
Summary of the Basic Characteristics of Assessed Algorithms

Scheme	Objective	Overhead	Scalability	Drawback
SPT	Shortest Path	Low	High	Data Redundancy
Infra	Max overlap routes	Very high	Low	Low scalability, high cost
ECIDA	Max overlap routes	Medium	Medium	Static routes

Control packet overhead. Number of control messages used to build the routing tree including the overhead to both create the clusters and set up all the routing parameters for each algorithm.

Efficiency. Packets per processed data. It is the rate between the total packets transmitted (data and control packets) and the number of data received by the sink.

Routing tree cost. Total number of edges in the routing tree structure built by the algorithm.

Loss of aggregated data. Number of aggregated data packets lost during the routing. In this metric, if a packet contains X aggregated packets and if this packet is lost, it is accounted the loss of X packets.

Number of transmissions. Sum of control overhead and data transmissions, i.e., the total packets transmitted.

Number of Steiner nodes. Number of Steiner nodes in the routing structure, i.e., the number of relay nodes.

V. CONCLUSION AND FUTURE WORK

Aggregation aware routing algorithms play an important role in event-based WSNs. In this work, we presented the ECIDA algorithm, a novel and reliable Data Aggregation Aware Routing Protocol for

WSNs. Our proposed ECIDA algorithm was extensively compared to two other known routing algorithms, InFRA and SPT, regarding scalability, communication costs, delivery efficiency, aggregation rate,and aggregated data delivery rate. By maximizing the aggregation points and offering a fault tolerant mechanism to improve delivery rate, the obtained results clearly show that ECIDA outperformed the InFRA and SPT algorithms for all evaluated scenarios. Also, we show that our proposed algorithm has some key aspects required by WSNs aggregation aware routing algorithms such as a reduced number of messages for setting up a routing tree, maximized number of overlapping routes, high aggregation rate, and reliable data aggregation and transmission.

As future work, spatial and temporal correlation of the aggregated data will also be taken into consideration as well as the construction of a routing tree that meets application needs.

We also plan to modify the ECIDA algorithm to stochastically select nodes that will be part of the communication structure. The goal is to find a balance between the overhead and the quality of the routing tree. In addition, new strategies will be devised to control the waiting time for aggregator nodes based on two criteria: average distance of the event coordinators, and spatial and semantics event correlation.

REFERENCES

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cyirci,"Wireless Sensor Networks: A Survey," Computer Networks, vol. 38, no. 4, pp. 393-422, Mar. 2002.

[2] K. Romer and F. Mattern, "The Design Space of Wireless Sensor Networks," IEEE Wireless Comm., vol. 11, no. 6, pp. 54-61, Dec. 2004.

[3] G. Anastasi, M. Conti, M. Francesco, and A. Passarella, "Energy Conservation in Wireless Sensor Networks: A Survey," Ad Hoc Networks, vol. 7, no. 3, pp. 537-568, http://dx.doi.org/10.1016/j.adhoc.2008.06.003, May 2009.

[4] A. Boukerche, R.B. Araujo, and L. Villas, "Optimal Route Selection for Highly Dynamic Wireless Sensor and Actor Networks Environment," Proc. 10th ACM Symp. Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM '07), pp. 21-27, 2007.

[5] O. Younis, M. Krunz, and S. Ramasubramanina, "Node Clustering in Wireless Sensor Networks: Recent Developments and Deployment Challenges," IEEE Network, vol. 20, no. 3, pp. 20-25, Dec. 2006.

[6] S. Olariu, Q. Xu, and A. Zomaya, "An Energy-Efficient Self-Organization Protocol for Wireless Sensor Networks," Proc. IEEE Intelligent Sensors, Sensor Networks and Information Processing Conf. (ISSNIP), pp. 55-60, Dec. 2004.

[7] H.S. AbdelSalam and S. Olariu, "A Lightweight Skeleton Construction Algorithm for Self-Organizing Sensor Networks," Proc. IEEE Int'l Conf. Comm. (ICC), pp. 1-5, http://dblp.uni-trier.de/db/conf/icc/icc2009.html#AbdelSalamO09, 2009.

[8] L. Villas, A. Boukerche, R.B. de Araujo, and A.A.F. Loureiro, "Highly Dynamic Routing Protocol for Data Aggregation in Sensor Networks," Proc. IEEE Symp. Computers and Comm. (ISCC), pp. 496-502, http://dx.doi.org/10.1109/ISCC.2010.5546580, 2010.

[9] L.A. Villas, D.L. Guidoni, R.B. Arau´ jo, A. Boukerche, and A.A. Loureiro, "A Scalable and Dynamic Data Aggregation Aware Routing Protocol for Wireless Sensor Networks," Proc. 13th ACM Int'l Conf. Modeling, Analysis, and Simulation of Wireless and Mobile Systems, pp. 110-117, http://doi.acm.org/10.1145/1868521.

[10] F. Hu, X. Cao, and C. May, "Optimized Scheduling for Data Aggregation in Wireless Sensor Networks," Proc. Int'l Conf. Information Technology: Coding and Computing (ITCC '05), pp. 557-561, 2005.

[11] I. Solis and K. Obraczka, "The Impact of Timing in Data Aggregation for Sensor Networks," IEEE Int'l Conf. Comm., vol. 6, pp. 3640-3645, June 2004.

[12] B. Krishnamachari, D. Estrin, and S.B. Wicker, "The Impact of Data Aggregation in Wireless Sensor Networks," Proc. 22nd Int'l Conf. Distributed Computing Systems (ICDCSW '02), pp. 575-578, 2002.

[13] J. Al-Karaki, R. Ul-Mustafa, and A. Kamal, "Data Aggregation in Wireless Sensor Networks—Exact and Approximate Algorithms," Proc. High Performance Switching and Routing Workshop (HPSR '04), pp. 241-245, 2004.

[14] A.P. Chandrakasan, A.C. Smith, and W.B. Heinzelman, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," IEEE Trans. Wireless Comm., vol. 1, no. 4, pp. 660-670, Oct. 2002.

[15] L.A. Villas, A. Boukerche, R.B. Araujo, and A.A. Loureiro, "A Reliable and Data Aggregation Aware Routing Protocol for Wireless Sensor Networks," Proc. 12th ACM Int'l Conf. Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM), pp. 245-252, http://doi.acm.org/10.1145/1641804.1641846, 2009.

NCITSF' 14