An Efficient Data Collection Approach for Wireless Sensor Networks

Hanieh Alipour and Alireza Nemaney Pour

Abstract—One of the most important applications of wireless sensor networks is data collection. This paper proposes as efficient approach for data collection in wireless sensor networks by introducing Member Forward List. This list includes the nodes with highest priority for forwarding the data. When a node fails or dies, this list is used to select the next node with higher priority. The benefit of this node is that it prevents the algorithm from repeating when a node fails or dies. The results show that Member Forward List decreases power consumption and latency in wireless sensor networks.

Keywords—Data Collection, Wireless Sensor Network, Sensor Node, Tree-Based

I. INTRODUCTION

WIRELESS Sensor Networks (hereinafter called WSNs) consists of a large number of smart sensor nodes which connect to each other in a wireless network. Each node receives the data from the environment and forwards it to its base station (technically called sink) (Fig. 1). Energy is the main important factor in the WSN because each sensor works with its non-rechargeable battery. Data collection is the principle application in the WSN. The applications consist of wildlife habitat monitoring, environmental research, volcano monitoring, civil engineering and wild land fire forecast/detection. If data collection is not performed efficiently, the sensors will have a lot of traffic and energy consumption. Consequently, the life time of the sensors will be short.

Figure 1 illustrates a simple structure of WSNs in general. First the Sink broadcasts the request, when one node finds the data it tries to transmit this with the help of other nodes to the Sink. WSN is classified into three categories:

(1) Cluster-Based: sensors in the network are divided into clusters. The nodes transmit data to the cluster head then cluster heads aggregate and compress the data, and forward it to the sink [1, 2].

(2) Chain-Based: a chain is formed to serve as a network structure. Data transmission is divided into multiple levels. Only subsets of nodes communicate with neighboring nodes at each level [3, 4].

Hanieh Alipour is with Dept. of the IT Engineering, Sharif University of Technology, International Campus, Kish Island, IRAN (e-mail: h.alipour@sharif.kish.ac.ir).

Alireza Nemaney Pour is with Dept. of the IT engineering Sharif University of Technology, International Campus, Kish Island, IRAN (e-mail:, pour@sharif.edu).

(3) Tree-Based: all nodes are organized in the form of a logical hierarchical tree. In this model the leaf node senses and forwards the data to the intermediate node. These inter- mediate nodes play the role of aggregators. Finally the Sink node serves as the root of the tree [3, 5].
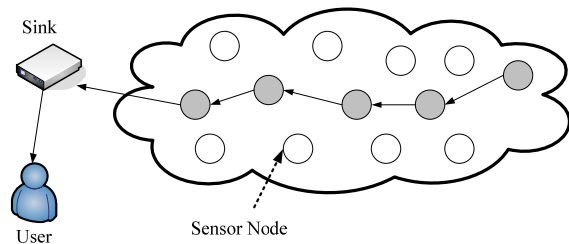

Fig. 1 Wireless Sensor Network in general

A problem with chain-based structure happens when one neighbor fails and consequently the chain for that data transmission is lost. In cluster-based structure, the cluster head or aggregator node may be attacked by malicious attacker. The common issue with all of these structures is that when a forwarding node fails to transmit the received data to its neighbor or a node in a higher position, the whole structure is lost. Consequently, the algorithm to construct the structure again needs to be repeated. This challenging point causes to use more energy, and leads to latency in data forwarding.

This paper proposes a tree-based algorithm for data forwarding to improve the above stated common issue. This improvement is possible with the help of using a Member Forward List. This list helps the other nodes to find the route for data forwarding when a previous forwarder node has failed.

This paper is organized as follows: Section 2 discusses the the related works. The proposed model and the related algorithm are shown in section 3 and 4 respectively. In section 5, we compare our model with the previously proposed ones. The conclusion is given in section 6.

II. RELATED WORK

In this section, we describe some of protocols related to our work. The main purpose of these designs is how to collect data and forward it to the sink efficiently in order to save the energy of the sensors. Considering energy efficiency, Low-Energy Adaptive Clustering Hierarchy (LEACH) [1] is the first energy conserving cluster formation protocol. Figure 2 illustrates the structure of this protocol in which all sensors are divided into several clusters for fusion. A cluster head collects all data from the sensor nodes and sends it to the sink.

This protocol, LEACH, is a good solution for data transmission to the sink compared to direct data transmission.
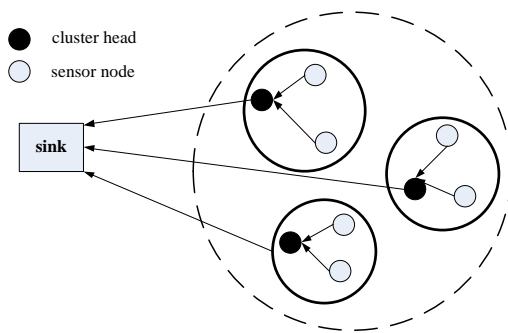
Fig. 2 Network structure of LEACH

The drawback of this protocol is that, it suffers from extra overhead because of its dynamic clustering. Moreover, when a cluster head dies, the cluster member nodes fail to forward the collected data. To solve this problem, the algorithm needs to be repeated to find the next cluster head.

Power Efficient Data-Gathering Protocol for Sensor Information Systems (PEGASIS) [2] is a chain-based data collection protocol. Figure 3 illustrates data transmission with eight nodes. All nodes are structured into a linear chain. In each step the closest neighbor of a node is selected. The selected node receives a data packet from one of its neighbors and forwards it to the next node. This process continues until the leader node sends the data to the sink. At the initial phase, a chain head is chosen with the following algorithm.
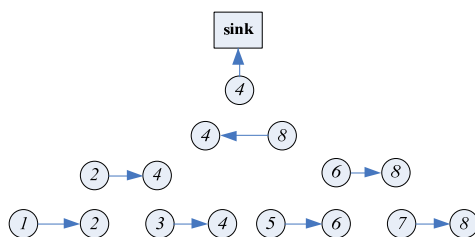

Fig. 3 Chain in PEGASIS

The performance of PEGASIS is better than LEACH when the product of *energy*delay* for energy consumption is considered as performance metrics. In special cases when the nodes are capable to communicate directly with the sink, it is not efficient to send data through a chain or a cluster head.

Tree Based Data Collection Scheme (TBDCS) [5] is a tree-based protocol. This protocol uses a logical hierarchy tree to forward the collected data from the leaf node to its root, the sink. This protocol runs an algorithm by a query message to find the shortest path length between the nodes and the sink (Fig. 4). This mechanism of forwarding the collected data is efficient because the data is forwarded through the shortest path length between the sink and the other nodes. Though efficient, the drawback of this protocol is that when an intermediate node fails to forward the data because of its lack of energy, the data cannot be sent to the sink.
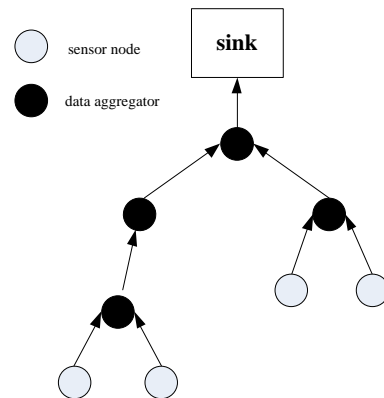

Fig. 4 Tree-based structure

As noticed, the common issue with all these protocols is that when a cluster head in cluster-based protocol, a chain member in chain-based protocol, and an intermediate node in tree-based protocol has failed to forward its data, the algorithm needs to be repeated again to deliver the data to its sink. This causes energy consumption which leads to shorten the life time of the nodes. In order to solve this common issue and to save the energy, in this paper, we propose an efficient protocol which improves the drawback of the tree-based protocol.

III. DESIGN PRINCIPLE

This section describes the design principle of our proposal method for data collection in WSNs. Figure 5 illustrates the physical structure of our proposal. There are two types of nodes in this network, The Sink and the wireless sensor nodes. The sink is responsible to inform the sensor nodes that what kind of data is needed, and to gather the collected data from them. The sensor nodes gather the information from the environment and forward it to the other nodes which are closer to the sink from hop point of view.
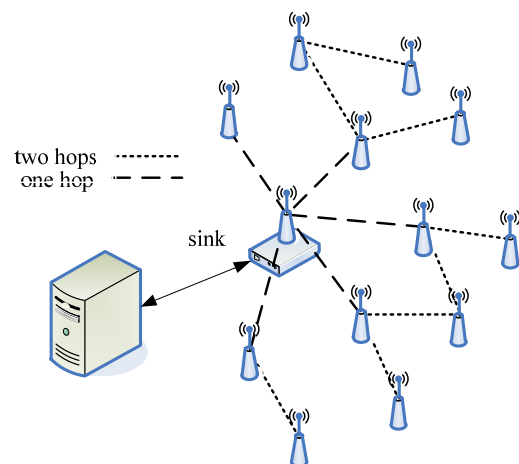

Fig. 5 physical model

Figure 6 illustrates the logical model of our proposal for data forwarding. This model is based on tree structure in which the root represents the sink. The intermediate nodes are the nodes which are selected for data forwarding. These nodes gather the information from the environment or forward the collected information for their child nodes. There are two kinds of intermediate nodes. The nodes with high priority are used for data forwarding. The other intermediate nodes with lower priority are the nodes which are reserved for the time when the node with higher priority has failed to forward the collected data. Finally, the leaf nodes are the nodes which just collect information from the environment and forward it to intermediate nodes.
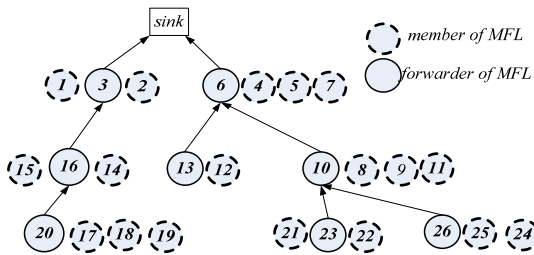

Fig 6 logical structure

In this model a list called Member Forward List (MFL) is introduced. MFL is used to find the suitable node for data forwarding. When a forwarder node is missed, the next node with higher priority from the MFL is selected. This property prevents the algorithm from repeating to find the forwarder node because always there are some nodes with different priority listed in the MFL. This list is created by the nodes under the following procedure:

(1) The nodes with one hop distance from the sink, broadcast the ID numbers.
(2) The IDs is checked by these nodes to find the smallest ID. This node is selected as the node which has the responsibility to create MFL.
(3) Next, this node broadcast MFL to two hop nodes.
(4) Steps 1, 2, are 3 are repeated between the nodes with two and more hops.

IV. DETAILED DESIGN

This section presents the details of our proposal. First, we explain the structure of MFL. This list includes node number, member ID number, and the priority. We assume that it is unique ID number. Each node with minimum ID number has higher priority, and when one node dies, the next node with higher priority in the list gets the responsibility to broadcast the updated list to the neighbors with one more hops, and to forward the collected data (Table I).

TABLE I
MEMBER FORWARD LIST

Member Forward List		
Node	ID Number	Priority

For simplicity, we consider a simple example (Fig. 6) with 26 nodes, {*1, 2, 3, 4,..., 26*}, to explain the detailed design of our proposal method as follows.

(1) Before the Sink broadcasts the request to all nodes, MFLs are created by the nodes.
(2) The nodes with one hop distance {*1, 2, 3, 4, 5, 6, 7*} broadcast their ID number.
(3) The nodes in the same signal range decide which has the highest priority based on the smallest ID number. In this example, we assume that node *3* has the smallest ID number between the nodes {*1, 2*}, and node *6* has the smallest ID number between the node {*4, 5, 7*}.
(4) Nodes *3* and *6* create MFL_1 and MFL_2, respectively (Tables II and III).

TABLE II
EXAMPLE OF MFL_1

Node Number	ID Number	Priority
3	*1001001*	1
1	*1101001*	2
2	*1101011*	3

TABLE III
EXAMPLE OF MFL_2

Node Number	ID Number	Priority
6	*10010101*	1
7	10010111	2
4	10011111	3
5	11010000	4

(5) Nodes *3* and *6* forward MFL_1 and MFL_2 to the nodes with two hop distance, and this algorithm is repeated for nodes with two or more hops. For example, node *16*, *13*, and *10* are nodes with higher priority with two hop distance, and the nodes *20*, *23*, and *26* and nodes with higher priority with three hops distance.

Now we assume that the sink has broadcasted the request, and the node *24* has found the requested data, and wants to forward it to the sink as the follows.

(1) Node *24* looks up its MFL to find the forwarder node. Assuming that node *10* is in the list with higher priority, node *24* forwards the collected data to node *10*.
(2) Node *10* looks up its MFL to find the next forwarder node. Assuming that node *6* is in the list with higher priority, node *10* forwards the received data to node *6*.
(3) Finally, node *6* forwards the received data to the sink.

Notice that, when a node dies or fails to forward the received data, the other nodes in the list with the highest priority is replaced as a forwarder. In this example, assuming that the node *6* has died, node *7* has the highest priority and is selected as a forwarder node (Fig. 7). Moreover, in some cases one or two nodes are shared in two areas, consequently, in two separate MFLs. In such case, when the node wants to send the collected data, it just looks up its MFL for the node with minimum ID number as a forwarder.
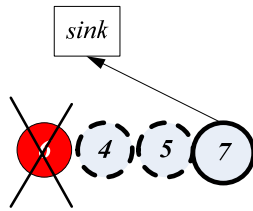
Fig. 7 when the forwarder node dies

V. EVALUATION

In this section, we compare our proposed protocol with two of typical previously proposed protocols. Table IV illustrates this comparison focusing on the following metrics:

- Energy Efficiency: the increase in the functionality of the WSN.
- Latency: the time required to collect data, and forward it from the sensor node to the sink.

Table 3 summarizes the comparison between previously proposed protocols, LEACH, PEGASIS, and our proposal. "X" represents the corresponding metric for evaluation. α and β denote the proportional values for improvement assuming that $\alpha < \beta$. When the energy efficiency of LEACH is X, the energy efficiency of PEGASIS and our protocol are αX and βX, respectively. By this values we can observe that the energy efficiency of our protocol is the best compared with two other protocols, or *our protocol > PEGASIS > LEACH*. This is because, the cluster size is not optimal in most cases, and the power consumption is high to forward data to the cluster head. PEGASIS optimizes the power consumption by chain path compared with LEACH but when the length of path is long, PEGASIS is affected with power consumption as well. Compared with these two typical protocols, our proposal method optimizes the pad length by introducing MFL.

TABLE IV
PERFORMANCE COMPARISONS

protocols	energy efficiency	latency
LEACH	X	X/α
PEAGASIS	αX	X
Our method	βX	X/α

When the latency of PEGASIS is X, the latency of LEACH and our protocol becomes X/α. By this values we can observe that the latency of our protocol is the best compared with two other protocols, or *our protocol < LEACH < PEGASIS*. This is because, in some cases in PEGASIS that the node is close to the sink, it has to forward the data based on the chain policy which leads to time waste. LEACH solves this issue by introducing cluster head. A cluster head can forward the received data directly to the sink. As well as LEACH, our protocol does not suffer from the stated issue about PEGASIS.

The major issue with LEACH and PEGASIS is that when a data forwarder node has failed or died for some reasons, the algorithm to find a new forwarder node needs to be repeated. But, our protocol improves this issue by introducing MFL

which prevents the algorithm from repeating. This property decreases the energy consumption and the latency. Consequently, our protocol is an efficient protocol compared with LEACH and PEGASIS.

Although we have compared the above protocols objectively by quantitative comparison criteria, the decision of which protocol is suitable for a particular application must also depend on the requirements of the application.

VI. CONCLUSION

In this paper, we introduced an efficient protocol to detect forwarder node for saving energy. In this protocol we used Member Forward List (MFL) to find an efficient and the shortest path for forwarding data to the sink.

At the end, we conclude our proposal with some of its contributions:

- Our protocol decreases the latency
- Our protocol increases energy efficiency.
- Our protocol prevents the algorithm from repeating when a node has failed or died for some reasons.

REFERENCES

[1] K. Maraiya, K. Kant, and N. Gupta, "Architectural Based Data Aggregation Techniques in Wireless Sensor Network: A Comparative Study," *International Journal on Computer Science and Engineering (IJCSE)*, vol. 3, no. 3, pp.1131-1134, 2002.

[2] S. Lindsey, C.S. Raghavendra, "PEGASIS: Power-Efficient Gathering in Sensor Information Systems," *Proceedings of IEEE on Conference Aerospace*, Los Angeles, CA. 2002, pp. 3-8.

[3] X. ZHANG, *Data Collection in Wireless Sensor Networks*, PhD thesis Electrical and Computer Engineering in the Graduate College of the University of Illinois at Chicago, 2009, pp. 11-20.

[4] A. Förster, and A.L. Murphy, "Optimal Cluster Sizes for Wireless Sensor Networks: An Experimental Analysis," *In Proceedings of ADHOCNETS*, 2009, pp.49-63.

[5] H. Li, H. Yi Yu, and A. Liu, "A Tree Based Data Collection Scheme for Wireless Sensor Network" Proceedings of *IEEE International Conference on Mobile Communications and Learning Technologies, (ICN/ICONS/MCL2006)*, Mauritius, 2006, pp.119-124.