LBN: Load-balancing Network for Data Gathering Wireless Sensor Networks

Wenlu Yang[1, 2], Chongqing Zhang[2], Minglu Li[2]
[1]Department of Electronic Engineering,
Shanghai Maritime University, Shanghai, China
[2]Department of Computer Science and Engineering,
Shanghai Jiaotong University, Shanghai, China
wenluyang@online.sh.cn, zhangchongqing@sjtu.edu.cn

Abstract. Hotspots of energy consumption and network congestions can be caused by load imbalance among sensor nodes in wireless sensor networks. This may lead to loss of data packets and premature death of sensor nodes which may cause the premature death of entire network. Load-balancing techniques can prolong the lifetime of sensor networks and avoid the occurrence of packet congestions. This paper proposes an approach that using load-balancing network to balancing the load in a static data gathering wireless sensor network. Experiments show the effectiveness of our approach.

1 Introduction

Many wireless sensor networks are deployed for monitoring applications. Examples of such kind of applications include monitoring the temperature of a vineyard, monitoring the pH value of a farmland's soil [1], etc. In such applications, sensor nodes are required to sense the monitored objects periodically and send the sensed data to the base station. The base station serves as the data aggregation point or the data sink of the WSN. Such wireless sensor networks are characterized by many-to-one traffic patterns [2] and called data gathering wireless sensor networks [8].

A Typical WSN may consist of a large number of sensor nodes. As the size of a WSN scales up, load unevenness may happen. Load unevenness may cause network congestion and even loss of data packet. What is more, some nodes with heavy load burden may run out of their energy rapidly and make the network become disconnected. By averaging the energy consumption of sensor nodes, load balancing can prolong the expected lifetime of the WSN. In addition, load balancing is also useful for avoiding congestion in network, thereby reducing wireless collisions [3].

Previous work has researched load balancing issue in WSNs. For example, R. C. Shah et al in [3] proposed an energy-aware routing multiple paths routing mechanism to balancing the load of sensor nodes. However, the WSN model they used is not many-to-one model. In [4], M. Perillo et al tried to solve unbalanced load distribution by transmission range optimization technique. Yet in their WSN model all sensor nodes can communication with the base station directly.

Constructing a WSN into a load-balancing tree is a way to solve load unevenness problem in WSNs for monitoring applications. P. H. Hsiao et al in [5] designed a load-balancing routing algorithm which achieves the balancing goal by constructing top balancing trees for wireless access networks. Because the flow in a WSN is totally different from the flow in a wireless access network, their work cannot be applied to wireless sensor networks. And their algorithm is designed for networks with mesh topology which is not common in most applications. In [6], H. Dai et al designed a node-centric algorithm that constructs a load-balancing tree fro WSNs. However, their algorithm is a centralized algorithm and is only applied to WSNs with grid topology.

Load-balance cannot be achieved by constructing only one static load-balancing tree in many cases. The can be illustrated by a simple example presented in Fig. 1. As for such a WSN, two load-balancing trees, as shown by Fig. 1 (a) and Fig. 1 (b) respectively, can be constructed. Yet none of them is a load-balancing tree. To solve such a problem, in Refs [7], H. Yang et al propose DQEB (Dynamic Query-tree Energy Balancing) protocol to dynamically adjust the tree structure to balance energy consumption for cluster-based WSNs. By their approach, a WSN will change its tree structure when working. As a result, their approach is energy-consumed because of the readjustment of the tree structure.

In this paper, we try to solve the load-balancing problem by constructing a load-balancing supply and demand network for a WSN. The routing structure that our algorithm builds is not a tree, but a network. The idea of our approach comes from the supply and demand network of commodity. In our approach a WSN is regarded as a market composed of a buyer and many producers and conveyancers. The base station which acts as the data collector is the only buyer and nodes are data producers and conveyancers. The buyer buys data from all the data producers. We propose a distributed algorithm to organize these buyer and conveyancers into a load-balanced network.

The remainder of this paper is organized as follows. In Section 2, WSN model and an overview of the approach are presented. In Section 3, we present the algorithm. In Section 4, the approach is evaluated and compared with several other load-balancing approaches. Finally, we conclude and give directions for future work in Section 5.

2 Network Model and Overview of LBN

2.1 Wireless Sensor Network Model

The WSN model used in this paper is based on following assumptions:

1) A WSN is composed of a base station and a large number of sensor nodes that are uniformly scattered on a plane. Each node is assigned a unique ID.

2) Sensor nodes are energy-constrained; while the base station is not energy-constrained.

3) The communication range of sensor nodes is fixed. After being deployed, all the base station and sensor nodes remain stationary. The nodes may organize into a flat WSN or a hierarchical WSN, for example, a cluster-based WSN.

4) A sensor node sense the monitored objects periodically every a fixed interval, and this is called a round. In each round a data packet is generated by a node and sent to the base station [8].

2.2 Overview of LBN

The main idea of LBN comes from the market mechanism, that is, the supply and demand network of commodity. A WSN can be regarded as a market composed of a buyer and many producers and conveyancers. In our approach, the measurement data is commodity. The base station which acts as the data collector is the only buyer and the nodes are data producers and conveyancers. The buyer buys data from all the data producers. All nodes are classified into different levels according to the least hop count that a node takes to send the data generated by this node to the base station.

The base station can only communicate directly with its direct neighbors. In this paper, the nodes that can communicate with the base station are called critical nodes. Correspondingly, the nodes that cannot communicate with the base station are called non-critical nodes. In a WSN, the critical nodes are most heavily burdened because these nodes need to relay data packets generated by other nodes to the base station. The base station can only make deals with the critical node and it pays same amount of money to the critical nodes. After that, the neighbors of the base station use the money it receives from the base station to buy data from their sons. Using proper strategy, we can ensure that the sons are assigned nearly same amount of money. In succession, a node uses the money it receives from its topper layer nodes to buy data from its lower layer nodes. In such a way, all nodes at the same level cost nearly identical amount of money. And as a result, the loads of the nodes at the same level are nearly identical. By making deals, the energy cost of the nodes can be balanced.

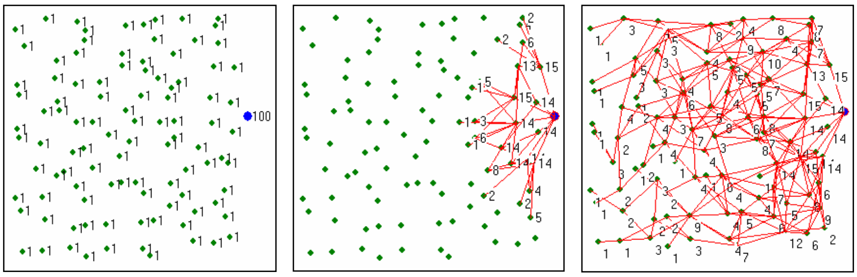
Although using deals to balance energy consumption as discussed above is applicable, exchanging deals information costs energy. To save energy, it is appropriate for the nodes to form fixed bargaining relations. We use a balanced bargaining network to meet this end. A bargaining tree constructed by our approach has following features:

1. The loads of nodes at the same level are nearly identical.

2. Loads of critical nodes are biggest, which means that the nodes that can communicate with the base station directly have the heaviest load and will deplete their energy earlier than other nodes.

A load-balancing network is constructed from the base station of a WSN. As described by Fig. 1 (a), each node is assigned one product and the base station is assigned the money that amounts to the number of the nodes firstly. To do this, the base station needs to know the number of the sensor nodes in the WSN. Then as Fig. 1 (b) describes, the base station sends every critical node a money message containing a number equal to $\lfloor (n / c_n) \rfloor$ or $\lceil (n / c_n) \rceil$, where n is the number of sensor nodes in the WSN and c_n is the number of neighbors of the base station. $\lfloor x \rfloor$ stands for the biggest integer of the integers that are smaller than or equal to the real number x, and $\lceil x \rceil$ stands for the smallest integer of the integers that are bigger than or equal to x. The number contained in the message represents the money that the base station used to buy data from it children. After a neighbor of the base station receives the message, if its product number is

1, then it deletes 1 from the number and resends the money to its children using certain strategy. In the end, all the nodes are organized into a load-balancing network which is depicted by Fig. 1 (c).



(a) (b) (c)

Fig. 1. Constructing Load-balancing Network from the Base station

3 Algorithm

Two steps are involved in constructing a flat WSN into a load-balancing network. The first step is initializing the WSN so as to organize the nodes into a layered network. Based on the layered network constructed in the first step, the second step uses a distributed algorithm to organize the nodes into a load-balancing network. Then the nodes begin to work and send the data packets to the base station using the load-balancing network constructed by the algorithm.

3.1 Initialization

A WSN can be viewed as a graph $G = (V, E, g)$ in which each member of set $V \setminus \{g\}$ stands for a sensor node, an edge (u, v) is included in E if sensor nodes u and v can communicate with each other directly, and node $g \in V$ represents the base station. First, we need to organize the WSN into a layered WSN according to the least hop counts of sensor nodes to the base station. Each node maintains several lists: the list of its parent nodes *PL*, the list of its children nodes *CL*, the list of its sibling nodes *SL*, and the list of its neighbor nodes *NL*.

For a node u, its four lists can be defined as:

$v \in PL$, if $v \in V$ and $(u, v) \in E$ and $h_u = h_v + 1$;

$v \in CL$, if $v \in V$ and $(u, v) \in E$ and $h_u = h_v - 1$;

$v \in SL$, if $v \in V$ and $(u, v) \in E$ and $h_u = h_v$;

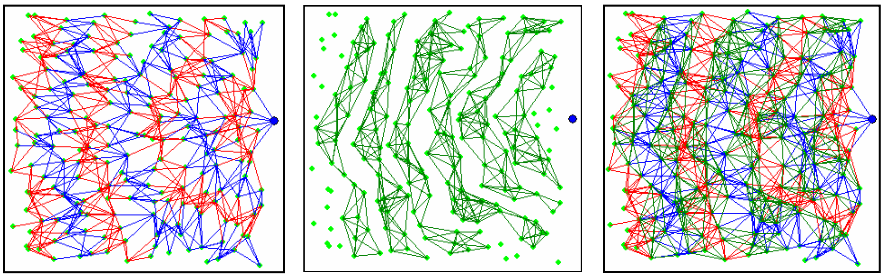
$v \in NL$, if $v \in V$ and $(u, v) \in E$;

where v is a neighbor node of node u; h_u and h_v are the hop count of node u and v respectively. A member of these lists is a structure defined as: <*nodeID*, *moneyTo*, *moneyFrom*, *productTo*, *productFrom*, *isFull*>, where *nodeID* is the ID of the member node in the list; *moneyTo* denotes the amount of money that node u sends to node v; *moneyFrom* denotes the amount of money that node u receives from node v; *productTo* denotes the amount of product that node u sends to node v; *productFrom* denotes the amount of product that node *u* receives from node *v*; *isFull* denotes if the money of node *v* exceeds the upper bound. If node *u* receives a money message that the *isReturn* field of the message is set to 1 from node *v*, then *isFull* is set to 1. If the field *isFull* of node *v* is 1, no money will be sent to node *v* as node u distributes money to its siblings and children.

The layered WSN can be constructed as follows. After being deployed, all nodes broadcast to notify its neighbors its existence. Using such a way, a node finds its neighbors and stores its neighbors into list *NL*. Then the base station sets its hop count to 0 and every other node sets its hop count to infinity. After that, the base station broadcasts a *hello* message containing its hop count value to all its neighbors. When a node *u* receives a *hello* message from node *v*, it extracts the hop count value h_v from the message. Then node *u* does following comparisons and takes corresponding action based on the comparison results:

1. If $h_v = h_u$, node *u* adds *v* into *SL*.
2. If $h_v = h_u + 1$, node *u* adds *v* into *CL*.
3. If $h_v > h_u - 1$, node *u* does nothing.
4. If $h_v = h_u - 1$, node *u* adds node *v* into *PL*.
5. If $h_v < h_u - 1$, node *u* deletes all the nodes from *PL* and adds node *v* into *PL*. Then, node *u* sets $h_u = h_v + 1$, and re-broadcasts the hello message with hop count value h_u to its neighbors.

By broadcasting *hello* messages and comparing h with hu step by step, the layered network can be constructed. An example of such a layered network is shown in Fig. 2. Parent-children relations are displayed in Fig. 2 (a); Sibling relations are exhibited by Fig. 2 (a); and Fig. 2 (c) is the combination of Fig. 2 (a) and Fig. 2 (b). It can easily be seen that *PL* list of the base station is empty. There is only one member, the base station, in the *PL* list of a critical node. And the *CL* list of a leaf node is empty.



(a) (b) (c)

Fig. 2. The Use of Siblings

3.2 Constructing an LBN

Five types of message, *hello*, *count*, *register*, *money* and *deal*, are used to build a load-balancing network. A message is defined as: <msgType, senderID, value, upperBound, isReturn>, where msgType can be *mtHello*, *mtCount*, *mtRegister*, *mtMoney*, or *mtDeal*; *senderID* is the source node of the message; *value* refers to the hop count if *msgType* is *mtHello*, the mount of money if *msgType* is *mtMoney* and the amount of product if *msgType* is *mtDeal*; *upperBound* is used to avoid the load of a non-critical node exceeding the load of any critical nodes. *isReturn* indicates whether or not a money message is returned back.

After the construction of the layered WSN, then the base station sends a *money* message containing the amount of money to each of the critical nodes. Before doing this, the base station needs to know how many nodes are there in the WSN. To do this, the base station floods a *count* message to all the nodes. A node sets one variable *product* to 1 and two variable *moneyAccept* and *moneyTotal* to be 0, and then it replies with a *register* message to the base station to report its existence. After counting the number of the *register* it receives, the base station can know the total number of sensor nodes in the WSN.

After the base station gets the number of nodes in the WSN, it sends a message containing the amount of money to each of its children. The amount is calculated as follows: let n be the number of nodes in the WSN; let c_n be the number of critical nodes. If n can be divided by c_n exactly, then $m_n = n / c_n$; otherwise the base station send some nodes with $m_n = \lfloor (n / c_n) \rfloor$, and other nodes with $\lceil (n / c_n) \rceil$ randomly. $\lfloor x \rfloor$ stands for the biggest integer that is smaller than x, and $\lceil x \rceil$ stands for the smallest integer that is bigger than x.

A node u has two list, *moneyList* and *dealList*. List *moneyList* is used to manage the money node u receives. List *dealList* is used manage the deals node u makes or the deal message it receives. Members of these two lists have same structure as: <*value*, *nodeID*>, where *value* refers to the amount of *money* or *product*; nodeID refers to the node that node u receives money from or sends deals to.

```
if (v in PL) or ((v in SL) and (msg->isReturn != 1) {
      if msg->upperBound <= moneyTotal
          return the money back to v;
      if (msg->upperBound - moneyTotal - msg->value)<0 {
                return partial money back to v;
                add a money record into money list;
      }
else
                add a money record into money list;
      if product = 1 {
                make a deal and send deal message to v;
                add a money record into deal list;
      }
}
if (v in CL) or ((v in SL) and (msg->isReturn == 1)
      mark the link as full;
if (moneyAccept >0) and (ifFull field of some child or sibling = 0) then {
       Select a node whose isFull is 0 randomly and send pmsg to it;
}
else { // return money back to v
       return money to v;
       delete corresponding record from money list;
}
```

Fig. 3. Processing Algorithm for a Money Message

After a node *u* receives a money message from node *v*, node *u* does following actions as shown in Fig. 3. Compared to the handling of a *money* message, the handling of a *deal* message is very easy. The processing algorithm for a *deal* message is shown in Fig. 4.

```
if (node u is not the base station) {
niv->productFrom = niv->productFrom + 1;
Find the oldest money record;
nix->productTo = nix->productTo + 1;
}
else
niv->productFrom = niv->productFrom + 1;
```

Fig. 4. Processing Algorithm for a Deal Message

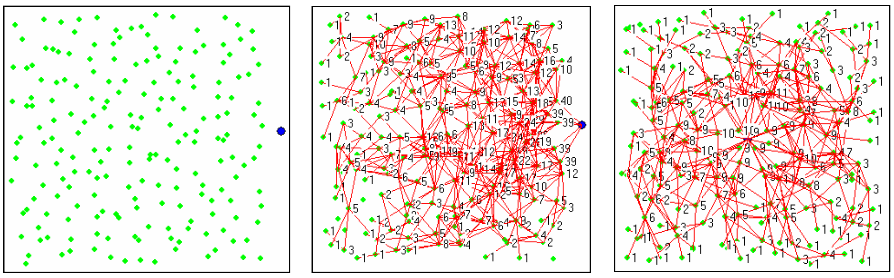
After the completion of the algorithm, each node has a deal list. The node collates the deal list and produces a list in which each member is a probability associated with a parent. When a node receives a data packet as it is working, it use this probability list to judge to which parent it should send the data packet.

4 Experimental Results

In this section we will show through simulations that LBN can prolong the lifetime of a WSN and avoid traffic congestion in the WSN. Our simulations do not consider the MAC or physical layer. The sensors are randomly deployed on regions with different size and shape. For each instance of deployment, we calculate the lifetime for the static network. The result is averaged over 50 instances for each set of network size and density. All simulations are implemented using a standalone VC++ program.

4.1 WSN Model

As Fig. 5 (a), one of the simulation scenes, shows, a WSN used in the experiments consists of 200 sensor nodes scattered on a 300m×300m square area. All nodes have same transmission ranges of 50 meters. The initial energy of a sensor node is 50 joules, and the energy of the base station is infinite. The WSN is deployed to monitor the environment in the sensing field. The sensor nodes are required to send their sensing data to the base station every 20 seconds. The size of a data packet is 10 bytes. In our simulations, the energy needed for a sensor node to sense, receive and transmit a packet is 2×10-5 joule, 2×10-5 joule and 1×10-4 joule respectively. These values of power consumption are calculated based on information presented in [12]. As Fig. 5 (b) and (c) show, the base station is located at the border of the simulation area or in the middle of the monitoring field. These two figures give two load-balancing networks constructed by LBN. In the experiments, we use these two kinds of WSNs that have different base station positions to compare the performances of different load-balancing approaches.



(a) (b) (c)

Fig. 5. Load-balancing Network for Hierarchichal WSNs

4.2 Approaches Compared and Performance Metrics

Four approaches are compared in the experiment: 1) the most energy saving approach which is denoted as MESA (stands for most energy saving approach). In MESA every node sends it data packets to the base station using a most energy saving route; 2) the approach using static load-balancing trees, and this approach is denoted as SLBT (stands for static load-balancing tree); 3) the approach using dynamic load-balancing tree to balance loads, and this approach is denoted as DLBT (stands for dynamic load-balancing tree); 4) the forth approach is our approach which is denoted as LBN.

Two metrics, the lifetime of WSNs and the average traffic of the node that has the heaviest load, are used to evaluate the performances of different approaches. The lifetime of a WSN is defined as the time span from the moment the WSN begins to work to the moment that the first sensor node runs out of its energy. The average traffic of the node that has the maximum load is used to reflect the congestion degree of the traffic flows in the network. The higher the value is, the severer is the congestion degree.

4.3 Performance Results

In this section, the comparison results of the performances of different approaches are presented. Two kinds of WSNs, WSNs that the base stations locate at the border of the monitoring area and WSNs that the base stations locate at the center of the monitoring area, are used in the experiments. 20 WSNs are used for each kind of WSNs. Four data sensing rates, 10 seconds, 20 seconds, 30 seconds and 40 seconds are used, that is to say, a node may generate and send a data packet to the base station every 10, 20, 30 or 40 seconds. For each data rata, the network lifetime and the average traffic of the node that has the maximum load is figured out. For the WSNs under same data rate, we average the network lifetimes and average traffics of the node that has the maximum load, compare these averages and present comparison results in the following figures.

Based on the WSNs that the base station locates at the border of the monitoring area, Fig. 6 compares the network lifetime of the four approaches. From the figure, with the data rate decreases, the lifetime of the WSNs grows. And the performance of LBN outscores other approaches under all data rates. This figure shows LBN can prolong the lifetime of a WSN compared to other approaches. Fig. 7 gives the comparison of the average traffics of the nodes that have the heaviest load of the four approaches. The traffic of the node that has the heaviest load decreases with the data rate decreases. Under all data rates the traffics of LBN are all lowest. Compared with other approaches, LBN is more effective in balancing the traffic load and thus can avoid congestion more effectively.

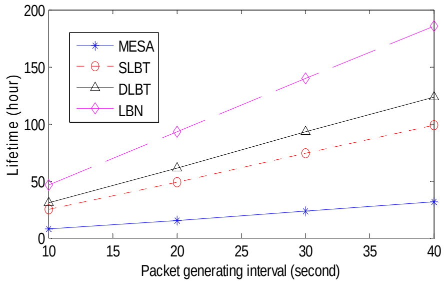


Fig. 6. Lifetime Comparison of All Approaches

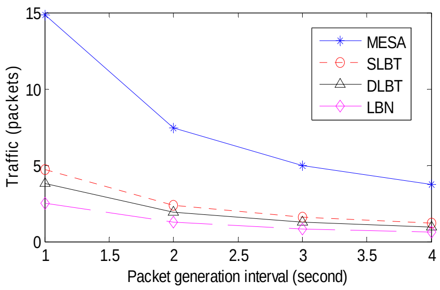


Fig. 7. Traffic Comparison of All Approaches

Base on the WSNs that the base station lies at the center of the monitoring area, Fig. 11 compares the network lifetime of the four approaches. From the figure, with the data rate decreases, the lifetime of the WSNs grows. And the performance of LBN outscores other approaches under all data rates. This figure shows LBN can prolong the lifetime of a WSN compared to other approaches. Fig. 12 gives the comparison of the average traffics of the nodes that have the heaviest load of the four approaches. The traffic of the node that has the heaviest load decreases with the data rate decreases. Under all data rates the traffics of LBN are all lowest. Compared with other approaches, LBN is more effective in balancing the traffic load and thus can avoid congestion more effectively.

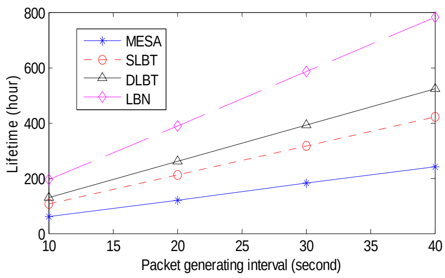


Fig. 8. Lifetime Comparison of All Approaches

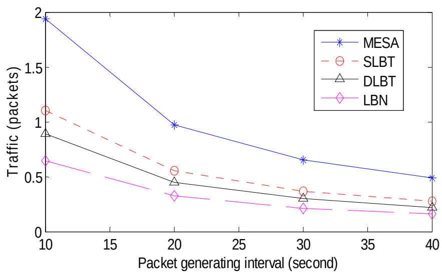


Fig. 9. Traffic Comparison of All Approaches

5 Conclusion and Future work

Hotspots of energy consumption and network congestions can be caused by load imbalance among sensor nodes in wireless sensor networks. This will lead to loss of data packets and premature death of sensor nodes which may cause the premature death of entire network. Load-balancing techniques can prolong the lifetime of sensor networks and avoid the occurrence of packet congestions. This paper proposes an approach that using load-balancing network to balancing the load in a static data gathering wireless sensor network. Experiments show the effectiveness of our approach. The approach can be adjusted to meet the need of WSNs in which the transmission range of nodes is adjustable.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci. "A survey on sensor networks. Computer Networks", Computer Networks, 38(4): 393-422, March 2002.

[2] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman. "A taxonomy of sensor network communication models". Mobile Computing and Communication Review, 6:28--36, 2002.

[3] R. C. Shah, J. M. Rabaey. "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks". IEEE WCNC 2002.

[4] M. Perillo, Zhao Cheng, and W. Heinzelman. "On the Problem of Unbalanced Load Distribution in Wireless Sensor Networks". Globecom 2004.

[5] P. H. Hsiao, A. Hwang, H. T. Kung, and D. Vlah. "Load-Balancing Routing for Wireless Access Networks". IEEE Infocom 2001.

[6] H. Dai and R. Han. "A Node-Centric Load Balancing Algorithm for Wireless Sensor Networks", IEEE GlobeCom 2003.

[7] H. Yang, F. Ye and B. Sikdar. "References A Dynamic Query-tree Energy Balancing Protocol for Sensor Networks", IEEE WCNC 2004.

[8] S. Lindsey, C. Raghavendra, and K. Sivalingam, "Data Gathering in Sensor Networks using the Energy*Delay Metric". IEEE Transactions on Parallel and Distributive Systems, special issue on Mobile Computing, pp. 924-935, April 2002.

[9] F. Zhao, L. Guibas. "Wireless Sensor Networks : An Information Processing Approach". Boston: Elsevier-Morgan Kaufmann; 2004.

[10] O. Younis and S. Fahmy, "HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad-hoc Sensor Networks". IEEE Transactions on Mobile Computing, volume 3, issue 4, pp. 366-379, Oct-Dec 2004.

[11] W.R. Heinzelman, A. Chandrakasan, H. Balakrishnan. "Energy-efficient communication protocol for wireless microsensor networks". IEEE HICSS. Jan, 2000.

[12] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, K. Pister. "System architecture directions for networked sensors". ACM ASPLOS-IX 2000.