Wireless Sensor Network, **2010, 2, 793-800**
doi:10.4236/wsn.2010.210095 Published Online October 2010 (http://www.SciRP.org/journal/wsn/).

BEAR: A Balanced Energy-Aware Routing Protocol for Wireless Sensor Networks

Ehsan Ahvar[1], **Mahmood Fathy**[2]
[1]*Department of Information Technology & Communication, Payame Noor University, Iran*
[2]*Department of Computer Engineering, Iran University of Science & Technology, Tehran, Iran*
E-mail: *Ehssana2000@yahoo.com, Mahfathy@iust.ac.ir*
Received May 28, 2010; *revised July* 4, 2010; *accepted August* 9, 2010

Abstract

Energy aware routing protocols can be classified into energy saver and energy manager. Energy saver protocols decrease energy consumption totally. Most of them try to find the shortest path between source and destination to reduce energy consumption. But energy manager protocols balance energy consumption in network to avoid network partitioning. Finding best route only based on energy balancing consideration may lead to long path with high delay and decreases network lifetime. On the other hand, finding best route only with the shortest distance consideration may lead to network partitioning. This paper improves SEER [1] routing protocol. Traditional SEER is only energy saver and has poor idea about energy balancing. Our proposed protocol, named BEAR, considers energy balancing and optimal distance both. It finds a fair tradeoff between energy balancing and optimal distance by *learning automata* concept. We simulate and evaluate routing protocols by Glomosim [2] simulator.

Keywords: Sensor Network, Energy Aware, Routing Protocol

1. Introduction

A Wireless Sensor Network (WSN) contains hundreds or thousands of sensor nodes. Basically, each sensor node comprises sensing, processing, transmission, mobility, position finding system, and power units. However, some of these components are optional like the mobility.

Sensor nodes are usually scattered in a sensor field, which is an area where the sensor nodes are deployed. Sensor nodes coordinate among themselves to produce high-quality information about the physical environment. These sensors have the ability to communicate either among each other or directly to an external base-station (BS). A base-station may be a fixed node or a mobile node capable of connecting the sensor network to an existing communications infrastructure or to the Internet where a user can have access to the reported data.

Many routing algorithms have been developed for sensor and ad hoc networks [3-26]. All of these routing protocols can be classified according to the network structure as flat, hierarchical, or location-based. In flat networks, all nodes play the same role while hierarchical protocols aim at clustering the nodes so that cluster heads can do some aggregation and reduction of data in order to save energy. Location-based protocols utilize the position information to relay the data to the desired regions rather than the whole network.

On the other hand, energy consumption in sensor networks is a very important factor. Because batteries carried by each mobile node have limited power supply, processing power is limited, which in turn limits services and applications that can be supported by each node. This is a big issue in sensor networks because, as each node is acting as both an end system and a router at the same time, additional energy is required to forward packets from other nodes.

Most of energy aware routing protocols are designed to save total energy consumption. They usually find the shortest path between Source and Sink to reduce energy consumption. In our opinion, an energy saver protocol that balances energy consumption is better than a poor energy server protocol. Finding best route only with the shortest distance consideration may lead to network partitioning. On the other hand, finding best route only based on energy balancing consideration may lead to long path with high delay and decreases network lifetime.

SEER is a simple energy efficient routing protocol [1]. It tries to reduce number of transmissions. But it has poor idea about energy management and energy balancing. On the other hand, LABER routing protocol [20] tries to balance energy consumption. But it has some visible problems such as low accuracy in updating of energy and high control overhead. In LABER, the Acknowledgement packet is forwarded to previous data sender and the other neighbors cannot update energy level of sender of Acknowledgement; low accuracy. Moreover, the Acknowledgment packet is an extra control overhead. We try to update energy without Acknowledgment packets to increase accuracy and decrease control overhead.

This paper addresses to design an energy-aware routing protocol for flat structure wireless sensor networks. The proposed protocol, BEAR, tries to save and balance energy consumption in network. It finds optimal route in energy level and hop count both. Routing decisions in BEAR are based on the distance to the Base Station as well as on remaining battery energy levels of nodes on the path towards the base station.

Section 2 discusses our design motivation and Section 3 presents SEER routing protocol. Section 4 discusses an introduction about learning automata. In Section 5 we present our proposed method. In Section 6 we present simulation details and Section 7 presents the results. Section 8 concludes on simulation results.

2. Motivation

Most of energy-aware routing protocols are only energy saver. They do not have an acceptable idea about energy balancing. This paper addresses to propose an energy saver protocol that considers energy balancing too. We found SEER routing protocol suitable to improve. SEER routing protocol is a simple energy aware routing protocol that considers energy saving and energy balancing both. But it has poor idea about energy balancing. Our proposed routing protocol, named BEAR, improves SEER in energy balancing and network lifetime.

3. SEER Routing Protocol

The different steps involved in the routing of packets in a SEER network are discussed next. It is important to note that each node is required to keep a neighbor table, which contains an entry for each node within transmission distance.

STEP 1: Network setup and neighbor discovery

Once the network has been deployed in the area where it is to operate, the sink transmits a broadcast packet. The broadcast packet contains the header fields shown in **Table 1.**

The source and destination addresses are 16 bit addresses enabling 65536 (216) unique addresses. Each node in the network is assumed to have a unique address within the network. The 8-bit sequence number is used to identify new broadcast messages. The sink increments the sequence number every time it sends a new broadcast message.

Table 1. Fields contained in the network layer header of broadcast messages.

Field Size (bits)
Source address 16
Destination address 16
Sequence number 8
Hop count 8
Energy level 16
Total 64

Nodes store the sequence number locally and forward broadcast messages only if the sequence number of the message is different from the stored one. The sequence number uses 8 bits in order to ensure that latency in the network does not cause nodes to mistakenly forward old broadcast messages. An 8-bit hop count ensures that nodes can be up to 255 hops from the sink.

When a node receives this initial broadcast message, it checks whether it has an entry in its neighbor table for the node that transmitted the message. If not, the receiver node adds an entry that consists of the neighbor address, hop count and energy level. The node then increments the hop count stored in the message and stores this hop count as its own hop count. It then retransmits the broadcast, but changes the source address field to its address and the energy level field to its remaining energy level. Every node in the network retransmits the broadcast message once, to all of its neighbors.

If a node receives a broadcast message with a lower hop count than the hop count it currently has, it updates its hop count. When this initial broadcast has been flooded through the network, each node knows its hop count and has the address, hop count and energy level of each of its neighbors.

STEP 2: Transmitting new data

When a node observes new data, as defined earlier, it initiates the process of routing. Two types of data packets can be sent: normal data and critical data. If a message is considered critical, for example when the sensed temperature changes from 25℃ to 100℃ within a very short time, a flag is set in the message indicating that it is critical. A node that originates a critical message transmits it to two neighbors instead of only one. The fields contained in the network layer header of data messages are shown in **Table 2.**

The creator address field is used to inform the sink of which node in the network originated the data message, since the source address is changed at every hop of the routing path. It is assumed that the sink knows where each

Table 2. Fields contained in the network layer header of data messages.

Field Size (bits)
Source address 16
Destination address 16
Creator address 16
Critical flag 1
Hop count 8
Energy level 16
Total 73

node is in the network. If the sink does not know which node originated the data and where the node is located, the data is useless.

A node bases its routing decision on two metrics, namely hop count and remaining energy. A node searches its neighbor table for all its neighbors with smaller hop counts than itself. If there is only one such neighbor, that neighbor is selected as the destination for the message. If there is more than one neighbor with a smaller hop count, the node selects the neighbor who has the highest remaining energy entry in the neighbor table. If a node does not have a neighbor with a smaller hop count, it searches for a neighbor with a hop count that is the same as its own. If there is only one such neighbor, that neighbor is selected. If more than one neighbor has the same hop count, the neighbor with the highest remaining energy is selected. If a node does not have any neighbors with hop counts smaller or equal to its own hop count, the message is discarded.

Before the message is sent, the remaining energy entry for the selected neighbor is decreased in the neighbor table. If the message is a critical message, the process of selecting a neighbor is repeated and the message is sent to a second neighbor. Using hop count as the routing metric ensures that the message is always sent in the direction of the sink.

STEP 3: Forwarding data

When nodes receive a data message they update the remaining energy value in the neighbor table for the neighbor that sent the message. Nodes that forward data messages follow the same process, except for minor differences, that the originating node uses to select the next neighbor in the routing path. The most important difference is that forwarding nodes take the creator address and source address into consideration when selecting the next hop neighbor. When searching the neighbor table for nodes with hop counts smaller or equal to its own, forwarding nodes also make sure that they do not select either the creator of the message, or the node from whom the message was received as the next destination. This ensures that there are no routing loops in the network.

STEP 4: Energy updates

Nodes may be used by more than one neighbor for routing and therefore the energy value stored in the neighbor tables of both of the node's neighbors will not be completely accurate. When a node's remaining energy falls below a certain threshold, it transmits an energy message to all of its neighbors to inform them of its energy level. The fields contained in the header of an energy message are shown in **Table 3**. Energy messages do not contain any data.

STEP 5: Network maintenance

The sink node periodically sends a broadcast message through the network so that nodes can add new neighbors that joined the network to neighbor tables and remove neighbors that have failed from the neighbor tables.

Nodes also update remaining energy values stored in the neighbor tables. It is important to note that broadcast messages do not contain any data.

4. Introduction to Learning Automata

Learning automata is an abstract model that randomly selects one action out of its finite set of actions and performs it on a random environment. Environment then evaluates the selected action and responses to the automata with a reinforcement signal. Based on selected action, and received signal, the automata updates its internal state and selects its next action. **Figure 1** depicts the relationship between an automata and its environment.

Environment can be defined by the triple $E = \{\alpha, \beta, c\}$ where $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_r\}$ represents a finite input set, $\beta = \{\beta_1, \beta_2, ..., \beta_r\}$ represents the output set, and $c = \{c_1, c_2, ..., c_r\}$ is a set of penalty probabilities, where each element c_i of c corresponds to one input action α_i

Environments, in which β, can take only binary values

Table 3. Fields contained in the network layer header of energy messages.

Field Size (bits)
Source address 16
Destination address 16
Hop count 8
Energy level 16
Total 56

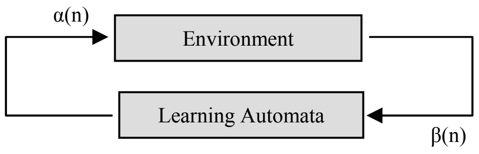

Figure 1. Relationship between learning automata and its environment.

0 or 1 are referred to as P-models. A further generalization of the environment allows finite output sets with more than two elements that take values in the interval [0, 1]. Such an environment is referred to as Q-model. Finally, when the output of the environment is a continuous random variable that assumes values in the interval [0,1], it is referred to as an S-model. Learning automata are classified into fixed-structure stochastic and variable structure stochastic. In the following, we consider only variable-structure automata.

A variable-structure automaton is defined by the quadruple $\{\alpha, \beta, p, T\}$ in which $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_r\}$ represents the action set of the automata, $\beta = \{\beta_1, \beta_2, ..., \beta_r\}$ represents the input set, $p = \{p_1, p_2, ..., p_r\}$ represents the action probability set, and finally $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ represents the learning algorithm. This automaton operates as follows. Based on the action probability set p, automaton randomly selects an action α_i, and performs it on the environment. After receiving the environment's reinforcement signal, automaton updates its action probability set based on Rel. (1) for favorable responses, and Rel. (2) for unfavorable ones.

$$\begin{aligned} p_i(n+1) &= p_i(n) + a[1-p_i(n)] \\ p_j(n+1) &= (1-a)p_j(n) \qquad \forall j,\ j \neq i \end{aligned} \tag{1}$$

$$\begin{aligned} p_i(n+1) &= (1-b)p_i(n) \\ p_j(n+1) &= \frac{b}{r-1} + (1-b)p_j(n) \qquad \forall j \quad j \neq i \end{aligned} \tag{2}$$

In these two equations, a and b are reward and penalty parameters respectively. For $a = b$, learning algorithm is called L_{R-P}, for $a << b$, it is called $L_{R\varepsilon P}$, and for $b = 0$, it is called L_{R-I}. For more information about learning automat the reader may refer to [27].

5. BEAR Routing Protocol

BEAR is an extended version of traditional SEER routing protocol with some visible difference specially in forwarding data procedure. The different steps involved in the routing of packets in a BEAR network are discussed next.

STEP 1: Network setup and neighbor discovery

The sink initializes the network by flooding the network with a broadcast message. Each node that receives the initiate packet adds an entry that consists of neighbor id, energy level and hop count. Then it computes and inserts the *selection probability* of neighbor nodes into the Neighbor List based on Rel. (3).

The node then increments the hop count stored in the message and then retransmits the broadcast, but changes the source address field to its address and the energy level field to its remaining energy level. Every node in the network retransmits the broadcast message only once, to all of its neighbors.

When this initial broadcast has been flooded through the network, each node knows their hop count, energy level and probability of each of its neighbors.

$$\Pr ob_s = 1/2 \times \left(\frac{1/HopCount_s}{\sum_{i=1}^{m} 1/HopCount_i} + \frac{EnergyLevel_s}{\sum_{i=1}^{m} EnergyLevel_i} \right) \qquad \forall s \leq m \tag{3}$$

STEP 2: Forwarding data

When a node observes new data it initiates the process of routing. In traditional SEER, the neighbor with a hop count that is smaller than the sending node's hop count is selected as the next hop. If multiple neighbors have smaller hop counts, the neighbor with the highest remaining energy is selected as the next hop. If a node does not have a neighbor with a smaller hop count, it selects the neighbor with the highest remaining energy from neighbors with an equal hop count to it. If the node does not have a neighbor with an equal hop count to it, the message is discarded.

But selecting next hop in our proposed protocol, BEAR, is based on learning automata. A node searches into its neighbor table for the neighbors with highest probability. The energy level of data sender node is attached to original data packet, piggybacking. Then the data packet is forwarded to neighbor with highest probability.

STEP 3: Updating energy and probabilities

All neighbors of data sender node receive the forwarded data packet, by overhearing technique. They only update the remaining energy value in the neighbor table for the neighbor that sent the data packet, by piggybacking technique, **Figure 2(a)**. Also, the previous data sender node receives its data packet again and updates all probabilities in neighbor List, **Figure 2(b)**.

In **Figure 2(a)**, green node selects gray node as its highest probability neighbor and forwards the data packet to it. The gray node receives data packet and updates energy level of green node in its Neighbor List. Another neighbor only updates energy level of green node in their Neighbor List and then discards the data packet. In **Figure**

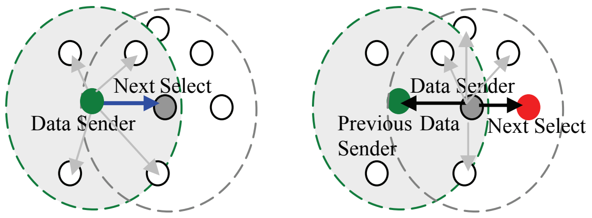

Figure 2. Updating (a) energy level (b) energy level and probabilities.

2(b), the gray node is data sender that selects the red node as highest probability neighbor. Therefore, gray node sends data to red. The red node receives the data packet from gray and update energy level of gray node. Green node receives the data packet again by overhearing and updates energy level of gray node and then updates probability of its neighbors.

There are 4 behavioral scenarios when previous data sender node receives the data packet by overhearing.

- If [($Energy_s/Avgenergy$) + ($HopCount/AvgHopCount$) < 2] then the route penalizes with β. β is computed based on Rel. (6). (Very bad)
- If [($Energy_s/Avgenergy$) + ($HopCount/AvgHopCount$) = 2] then the route penalizes with $\beta/2$. (Bad)
- If [2 < ($Energy_s/Avgenergy$) + ($HopCount/AvgHopCount$) < 2.2] then the route rewards with $\alpha/2$. α is computed based on Rel. (5). (Acceptable)
- If [($Energy_s/Avgenergy$) + ($HopCount/AvgHopCount$) ≥ 2] then the route rewards with α. α is computed based on Rel. (5). (Good)

That $Energy_s$ is the received energy of data sender node, next selected node, and *HopCount* is the distance of data sender node from Station.

Avgenergy is the average energy level of neighbor nodes. *AvgHopCount* is average distance of all neighbor nodes from the Station.

$$Avgenergy = \left(\sum_{i=1}^{m} EnergyLevel_i\right)/m \tag{4}$$

In the Rel. (4), $EnergyLevel_i$ is the energy level of neighbor_i and m is Neighbor List size.

Avghopcount is the average hopcounts of neighbor nodes from the Station node.

$$Avg\text{HopCoun} = \left(\sum_{i=1}^{m} \text{HopCoun}_i\right)/m \tag{5}$$

In the Rel. (5), HopCount_i is the distance of neighbor_i from Station and m is Neighbor List size.

$$\alpha = \lambda + \delta_1 \frac{Energy_i + Max\text{Hop} - \text{HopCoun}_i}{InitEnergy + MaxHop} \tag{6}$$

$$\beta = \lambda + \delta_2 \frac{Avgenergy - Energy_i + \text{HopCoun}_i}{Avgenergy + MaxHop} \tag{7}$$

In Rel. (6) and Rel. (7), α and β are reward and penalty parameters respectively, HopCoun_i is distance between sender node and Station node, *InitEnergy* is the initial energy of nodes, *Maxhop* is maximum distance between nodes in network and λ is minimum value for reward and penalty parameters.

δ_1 and δ_2 are selected which cause to dose not exceed of a threshold for α and β parameters.

The operation of the BEAR protocol can be summarized as follows:

- The sink initializes the network by flooding the network with a broadcast message.
- Nodes add all their neighbors' information to their neighbor tables.
- The node with highest probability is selected and data packet is forwarded to it.
- The energy level of data sender node is updated by its neighbors.
- The probability of data sender node is updated into the Neighbor List of previous data sender node in each hop.

BEAR routing protocol does not need an energy message, Step.4 in traditional SEER, because the energy level of each sender node is updated into its Neighbor Table automatically, overhearing and piggybacking. Also the sink node sends a broadcast message through the network so that nodes can add new neighbors that joined the network to neighbor tables and remove neighbors that have failed from the neighbor tables. But sending rate of broadcast message through the network is related to mobility. The network with none mobility nodes dose no need to send broadcast message as well as the network with high mobility.

6. Simulation Model

This section simulates and compares our proposed routing protocol, BEAR, with traditional SEER routing protocol. To compare the routing protocols, a parallel discrete event-driven simulator, *GloMoSim*, is used. *GloMoSim* is a simulation tool for large wireless and wired networks [2].

Table 4 describes the detailed setup for our simulator.

7. Simulation Results

In this section we evaluate and compare the various routing schemes. The performance measures of interest in this

Table 4. Simulation setting.

SIMULATION-TIME	1200 SECOND
TERRAIN-DIMENSIONS	1000 m * 1000 m
NUMBER-OF-NODES	200, 300, 500, 1000, 2000
NODE-PLACEMENT	Uniform/Random
MOBILITY	NONE
NUMBER OF EVENTS (Sources)	100
TEMPARATURE	290.0 (in K)
RADIO-BANDWIDTH	2000000 (in bps)
RADIO-TX-POWER	5.0 (in dBm)
ENERGY- TRANSMIT-LEVEL	0.0002 (in mW)
MAC-PROTOCOL	802.11
NETWORK-PROTOCOL	IP
PROPAGA-TION-PATHLOSS	FREE-SPACE
RADIO-TYPE	RADIO-ACCNOISE

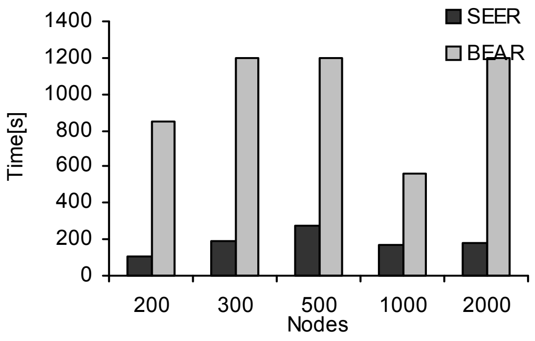

Figure 3. Time at which the first neighbor of Station fails due to depleting its energy source. The nodes are randomly distributed over the sensor area.

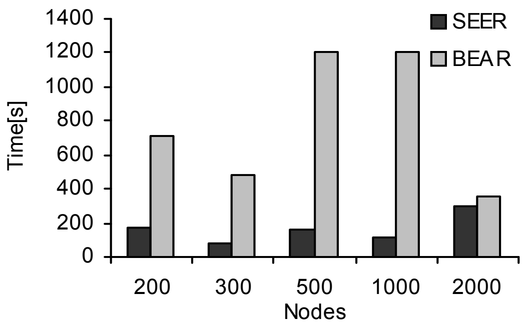

Figure 4. Time at which the first neighbor of Station fails due to depleting its energy source. The nodes are uniformly distributed over the sensor area.

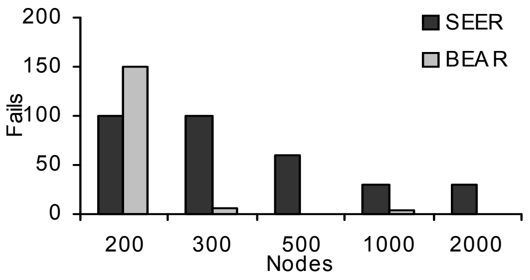

Figure 5. Number of fails at the end of simulation. The nodes are randomly distributed over the sensor area.

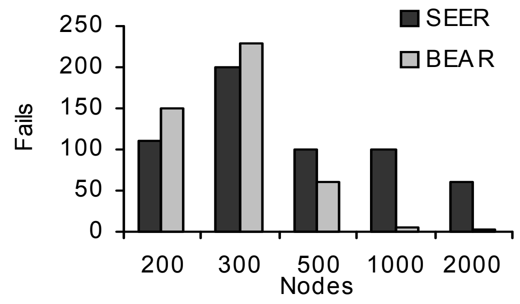

Figure 6. Number of fails at the end of simulation. The nodes are uniformly distributed over the sensor area.

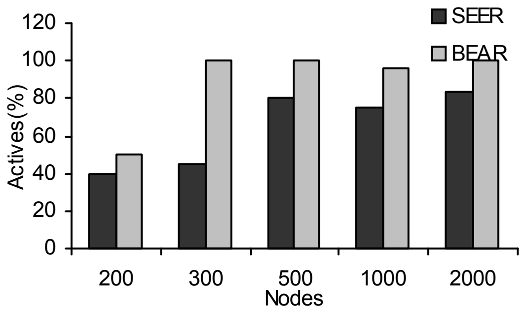

Figure 7. Percentage of active neighbors of Station at the end of simulation. The nodes are randomly distributed over the sensor area.

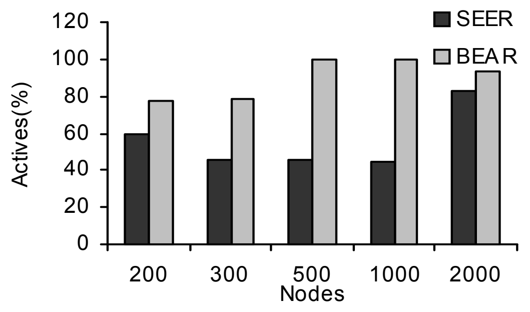

Figure 8. Percentage of active neighbors of Station at the end of simulation. The nodes are uniformly distributed over the sensor area.

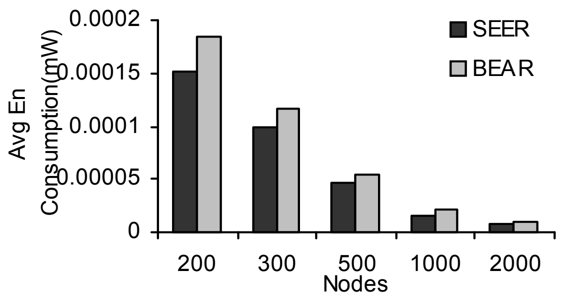

Figure 9. Average energy consumption (mW) in transmission mode. The nodes are randomly distributed over the sensor area.

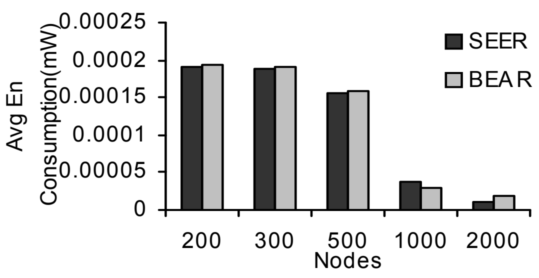

Figure 10. Average energy consumption (mW) in transmission mode. The nodes are uniformly distributed over the sensor area.

study are: a) Average energy consumption of transmission (in mW); b) Energy balancing. The variables are: number of nodes and node placement.

The simulation of the protocols started with a broadcast message at the start of the simulation. We select 100 Sources to send data packets to the Station during the simulation. The Sources are selected randomly in different times. Each Source generates a 512-bit data packet and forwards it through the network. Simulations are performed to evaluate the network lifetime achieved by each protocol. At the beginning of simulation, the transmission energy level of each node was 0.0002 mW. Four tests are achieved to evaluate the protocols:

Test 1: The time until the first neighbor of sink fails.

Test 2: Number of fails.

Test 3: Percentage of active neighbors of Station at the end of simulation.

Test 4: The average remaining energy of all the nodes in the network, at transmission mode.

Generally, the results from Test 1, Test 2, Test 3 and **Figures 3-8** show that BEAR is better than the SEER protocol in energy managing. This is due to the fact that BEAR sends data packet along a balanced path. Also at the end of simulation BEAR has very low fails. Therefore, performance of our protocol is very good especially in high-density networks. On the other hand, the results of Test 4, **Figures 9** and **10** show that there are not visible difference in energy consumption between SEER and BEAR.

Therefore, we could improve SEER routing protocol in energy balancing without visible increment in energy consumption. It means our BEAR routing protocol can increase network lifetime compare with traditional SEER.

8. Conclusions

In this paper we studied energy aware routing protocols. Then we proposed a new energy saver/balancer routing protocol. The bease of our study was SEER routing protocol. We simulated and compared our routing protocol with traditional SEER. Results showed that our protocol generally was better than SEER in energy. Our BEAR protocol had very low fails. It was more balancer than SEER routing protocol especially in high-density networks. In general we found that BEAR gives better performance, compared to SEER.

9. References

[1] G. P. Hancke and C. J. Leuschner, “SEER: A Simple Energy Efficient Routing Protocol for Wireless Sensor Networks,” *South African Computer Journal*, Vol. 39, 2007, pp. 17-24.

[2] X. Zeng, R. Bagrodia and M. Geria, “Glomosim: A Library for Parallel Simulation of Large Scale Wireless Networks,” *Proceedings of the* 12*th Workshop on Parallel and Distributed Simulations*, Banff, 1998, pp. 154-161.

[3] M. Youssef, M. Younis and K. Arisha, “A Constrained Shortest-Path Energy-Aware Routing Algorithm for Wireless Sensor Networks,” *Proceedings of IEEE Wireless Communications and Networking Conference*, Orlando, Vol. 2, 2002, pp. 794-799.

[4] D. Braginsky and D. Estrin, “Rumor Routing Algorithm for Sensor Networks,” *Proceeding of* 1*st ACM Workshop on Sensor Networks and Applications*, Atlanta, 2002, pp. 22- 31.

[5] T. Banka, G. Tandon and A. P. Jayasumana, “Zonal Rumor Routing for Wireless Sensor Networks,” *Proceedings of IEEE Conference on Information Technology*: *Coding and Computing*, Las Vegas, Vol. 2, 2005, pp. 562-567.

[6] T. Hou, Y. Shi and J. Pan, “A Lifetime-Aware Single-Session Flow Routing Algorithm for Energy-Constrained Wireless Sensor Networks,” *Proceedings of IEEE Military Communications Conference*, Boston, Vol. 1, 2003, pp. 603-608.

[7] F. Block and C. Baum, “An Energy-Efficient Routing Protocol for Wireless Sensor Networks with Battery Level Un- certainty,” *Proceedings of IEEE Military Communications Conference*, Anaheim, Vol. 1, 2002, pp. 489-494.

[8] A. Manjeshwar and D. P. Agrawal, “APTEEN: A Hybrid Protocol for Efficient Routing and Comprehensive Information Retrieval in Wireless Sensor Networks,” *Proceedings of the* 16*th International Parallel and Distributed Processing Symposium*, Fort Lauderdale, 2002, pp. 195-202.

[9] R. Manohar and A. Scaglione, “Power Optimal Routing in Wireless Networks,” *Proceedings of IEEE International Conference on Communications*, Seattle, Vol. 4, 2003, pp. 2979-2984.

[10] Z. W. Zheng, Z. H. Wu and H. Z. Lin, “Clustering Routing Algorithm Using Game-Theoretic Techniques for WSNs,” *Proceedings of the International Symposium on Circuits and Systems*, Vancouver, Vol. 4, 2004, pp. IV-904-7.

[11] D. Petrovic, R. C. Shah, K. Ramchandran and J. Rabaey, “Data Funneling: Routing with Aggregation and Compression for Wireless Sensor Networks,” *Proceedings of the* 1*st IEEE International Workshop on Sensor Network Protocols and Applications*, Anchorage, 2003, pp. 156-162.

[12] C. Intanagonwiwat, R. Govindan and D. Estrin, “Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks,” *Mobile Networks and Applications*, Vol. 1, No. 4, 2000, pp. 56-67.

[13] C. Chen, “Energy Efficient Routing for Clustered Wireless Sensors Network,” *Proceedings of the* 29*th Annual Conference of the IEEE Industrial Electronics Society*, Roanoke, Vol. 2, 2003, pp. 1437-1440.

[14] C. Schurgers and M. Srivastava, “Energy Efficient Routing in Wireless Sensor Networks,” *Proceedings of IEEE Military Communications Conference*, Washington, Vol. 1, 2001, pp. 357-361.

[15] M. Senouci and G. Pujolle, "Energy Efficient Routing in Wireless Ad Hoc Networks," *Proceedings of IEEE International Conference on Communications*, Orlando, Vol. 7, 2004, pp. 4057-4061.

[16] D. Tian and N. Georganas, "Energy Efficient Routing with Guaranteed Delivery in Wireless Sensor Networks," *Proceedings of IEEE Wireless Communications and Networking Conference*, New Orleans, Vol. 3, 2003, pp. 1923-1929.

[17] J. Zhang and H. Shi, "Energy-Efficient Routing for 2D Grid Wireless Sensor Networks," *Proceedings of the International Conference on Information Technology*: *Research and Education*, Newark, 2003, pp. 311-315.

[18] A. Datta, "Fault-Tolerant and Energy-Efficient Permutation Routing Protocol for Wireless Networks," *Proceedings of the International Parallel and Distributed Processing Symposium*, Nice, 2003, pp. 22-26.

[19] M. gholipour and M. R. Meybodi, "LA-Mobicast: A Learning Automata Based Distributed Protocol for Mobicast in Sensor Networks," *Proceeding of* 12*th Annual International CSI Computer Conference*, Tehran, 2007, pp. 1154-1161.

[20] S. M. Abolhasani, M. R. Meybodi and M. Esnaashari, "LA-BER: A Learning Automata Based Energy-Aware Routing Protocol for Sensor Networks," *Proceedings of the* 3*rd Information and Knowledge Technology*, Ferdowsi University of Mashad, Mashad, 2007.

[21] Y. B. Ko and N. H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," *Proceedings of the IEEE International Conference on Mobile Computing and Networking*, Dallas, 1998, pp. 66-75.

[22] J. Broch, D. Johnson and D. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks," IETF Internet Draft, 1999, work in progress. http://www.ietf.org/internet-drafts/draft-ietfmanet-dsr-03.txt

[23] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing*, Kluwer Academic Publishers, Vol. 353, 1996, pp. 153-181.

[24] C. E. Perkins and E. M. Royer, "Ad Hoc On-Demand Distance Vector Routing," *Proceeding of the* 2*nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, 1999, pp. 90-100.

[25] T. He, J. Stankovic, C. Lu and T. Abdelzaher, "A Spatiotemporal Communication Protocol for Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 16, No. 10, 2005, pp. 995-1006.

[26] Z. J. Haas and M. R. Pearlman, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," IETF, MANET Internet Draft, 1998.

[27] K. S. Narendra and M. A. L. Thathachar, "Learning Automata: An Introduction," Prentice Hall, 1989.