LJMU Research Online

Ahvar, E, Ahvar, S, Lee, GM and Crespi, N

 An energy-aware routing protocol for query-based applications in wireless sensor networks.

http://researchonline.ljmu.ac.uk/id/eprint/13006/

Article

For more information please contact researchonline@ljmu.ac.uk

An Energy-Aware Routing Protocol for Query-Based Applications in Wireless Sensor Networks

Ehsan Ahvar,[1] Shohreh Ahvar,[2] Gyu Myoung Lee,[1] and Noel Crespi[1]

[1] *Wireless Networks and Multimedia Services Department, Institut Mines-Telecom, Telecom SudParis,*
 9 rue Charles Fourier, 91011 Evry Cedex, France
[2] *Electrical and Computer Engineering Department, Isfahan University of Technology, Isfahan, Iran*

Correspondence should be addressed to Gyu Myoung Lee; g.m.lee@ljmu.ac.uk

Wireless sensor network (WSN) typically has energy consumption restriction. Designing energy-aware routing protocol can significantly reduce energy consumption in WSNs. Energy-aware routing protocols can be classified into two categories, energy savers and energy balancers. Energy saving protocols are used to minimize the overall energy consumed by a WSN, while energy balancing protocols attempt to efficiently distribute the consumption of energy throughout the network. In general terms, energy saving protocols are not necessarily good at balancing energy consumption and energy balancing protocols are not always good at reducing energy consumption. In this paper, we propose an energy-aware routing protocol (ERP) for query-based applications in WSNs, which offers a good trade-off between traditional energy balancing and energy saving objectives and supports a soft real time packet delivery. This is achieved by means of fuzzy sets and learning automata techniques along with zonal broadcasting to decrease total energy consumption.

1. Introduction

A wireless sensor network (WSN) is composed of a large number of sensor nodes, which are densely deployed either inside the phenomenon or very close to it. Since the sensor nodes are often inaccessible, the lifetime of a sensor network depends on the lifetime of the power resources of the nodes. Power is also a scarce resource due to the size limitations [1].

Many routing algorithms and protocols have been proposed for different types of WSNs (i.e., [2–4]) among which we have identified a category known as query-based routing. In query-based routing protocols, a station S sends queries to find specific events among the WSNs. The strategies used for routing these queries and their corresponding replies can be classified into two major groups, energy savers and energy balancers. The former tries to decrease the energy consumption of the network as a whole and thereby increase the operation lifetime which also usually leads to the utilization of the shortest paths. The latter, on the other hand, tries to balance the energy consumption of the nodes to prevent partitioning of the network.

Rumor is an energy saving protocol that provides an efficient mechanism combining push and pull strategies to obtain the desired information from the network [5]. In Rumor, the nodes generating events send notifications that leave a sticky trail along the network. When query agents visit a node where an event notification agent has already passed through, they can find pointers (i.e., the trail) towards the location of the corresponding source. In general terms, when a node receives a query, two things can happen: (i) the node already has a route toward the target event, so it only needs to forward the query along the route; or (ii) the node does not have a route, and therefore, it forwards the query to a random neighbor. The random selection of the neighbor in this case is relatively constrained, since each node keeps a list of recently visited neighbors to avoid repeatedly visiting them. Clearly, the forwarding strategy in Rumor could end up producing spiral paths, so an intuitive improvement would be to reduce its level of routing indirection. To this end, Chou et al. proposed the straight line routing (SLR) protocol [6], which aims at making the routing path grow as straight as possible. More recently, Shokrzadeh et al. have made significant efforts

to improve Rumor in different aspects with their directional Rumor (DRumor) [7]. Shokrzadeh et al. later improved their DRumor protocol by means of what they called the second layer routing (SecondLR) [8]. SecondLR uses geographical routing immediately after locating the source of an event, and Shokrzadeh et al. have shown that this approach considerably improves the performance of DRumor. Despite these efforts, current query-based routing protocols are mainly energy savers and have shown relatively poor performance when it comes to balancing energy consumption. Much more recently, Ahvar et al. have proposed the energy-aware query-based routing protocol (EQR), an energy saving and balancer routing protocol [9]. EQR uses zonal broadcasting to reduce energy consumption.

This paper presents a routing strategy applicable to various forms of query-based applications that offers a reasonable trade-off between the energy saving and balancing. More precisely, we propose an energy-aware routing protocol for query-based applications in WSNs called the ERP, which is supported by learning automata and fuzzy sets and which uses zonal broadcasting to decrease the total energy consumed. Our initial results demonstrate the potential and the effectiveness of ERP, making it a promising candidate for a number of WSN applications.

The rest of the paper is organized as follows. In Section 2, we introduce our design goals. Section 3 presents the main contribution of this paper which is basically the ERP routing strategy. The assessment of ERP is covered in Section 4, and Section 5 concludes the paper.

2. Design Goals

More specifically, ERP satisfies the following design objectives.

(1) Energy-distance awareness: energy-awareness means both energy saving and energy balancing. Energy saver protocols try to decrease the energy consumption of a network to increase network lifespan. They usually try to find the minimum path length to reduce energy consumption. Energy balancer protocols try to balance the energy consumption of nodes to prevent network partitioning. Finding the best route only based on energy balancing may lead to longer paths with greater delay, and finding the best route based only on energy savings and optimal distance may lead to network partitioning. The ERP is an energy saver and an energy balancer at the same time. It achieves a trade-off between distance and energy by using learning automata and fuzzy sets techniques.

(2) Accuracy: finding the best node as a next hop in aspects of energy saving and balancing is a big challenge for routing protocols. Most of energy-aware routing protocols find the next hop based on only one measurement factor, such as energy level. The ERP, however, considers hop count and distance as well as energy level, simultaneously, utilizing more than one decision-making technique to achieve more exact results.

(3) Localized behavior and scalability: the ability to maintain performance characteristics irrespective of the size of the network is referred to as scalability. Pure localized algorithms are those in which any action invoked by a node should not affect the system as a whole. In these protocols, a node usually uses flooding to discover new paths. In WSNs, where thousands of nodes communicate with each other, broadcast storms may result in significant power consumption and even in a network meltdown. To avoid that situation, most of the distributed operations in ERP are localized to achieve high scalability.

(4) Minimal state architecture: the physical limitations of WSNs, such as large scale, high failure rate, and constrained memory capacity, demand a minimal state approach. The ERP only maintains the immediate neighbor's information and so it does not need a large routing table. Thus, its memory requirements are minimal.

(5) Link failure detection: the ERP has the ability to find a broken link. Unlike most protocols, it does not use the acknowledge packet to check link stability. The ERP verifies links by means of the overhearing technique described herewith.

(6) Minimal control packet: in many routing protocols, the nodes' energy levels are forwarded to neighbors by acknowledge packets. The proposed routing protocol uses the overhearing technique for updating energy levels. In most routing protocols, nodes with very low energy levels send a packet to warn their neighbors. The ERP instead has a threshold, and when each neighbor forwards a packet, all the neighbors receive it and compare the attached energy level of the sender node to the threshold level. If the sender energy is under the threshold, the sender is considered to be a dead node and will not be selected again as a next hop. Therefore, the proposed protocol does not need an extra packet to announce a dead node. The threshold level is the energy required to send a packet.

3. Energy-Aware Routing Protocol (ERP)

ERP is a query-based routing protocol designed to consider both energy and distance while routing packets across a network. It balances the load among the different sensors with a twofold goal: preventing the sensors from running out of battery while keeping the routes to reach the destinations relatively short. ERP can be divided into two phases: query broadcasting and data forwarding (Figure 1).

3.1. Query Broadcasting Phase. When a WSN starts its operation, station S that will issue the query packet to find destination D will have no zonal information. Therefore, the query mode used by station S at the beginning of the operations will typically be a simple broadcasting, which means that the entire region is assumed to be the query zone. The query packet contains the following fields: sender Id (SI), destination Id (DI), sender position (SP1), station position (SP2), destination position (DP), destination radius (DR) and energy level (EN). The SI saves the Id of the query-sender node; the destination Id is carried by the DI field. SP1 and SP2 forward the positions of the sender and the station, respectively, and, if available, DP includes the position of
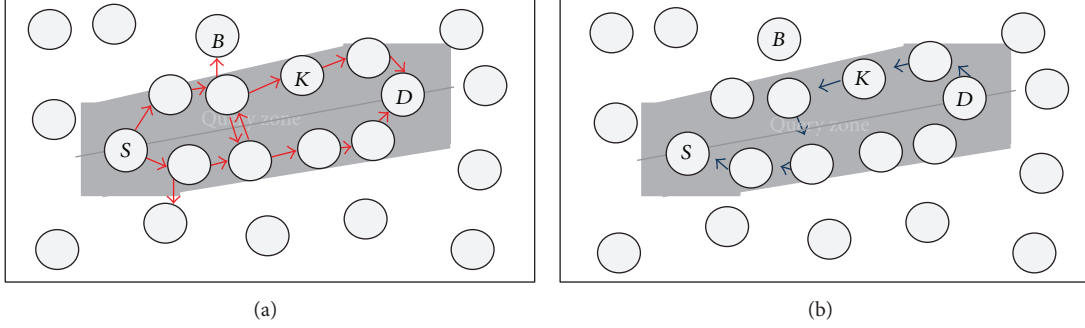
(a)

(b)

FIGURE 1: EPR routing mechanism: (a) query broadcasting and (b) data forwarding.

the destination or the center of requested zone. EN indicates the energy level of the sender node. In the first time of query broadcasting for finding destination D, the station S sets SI, SP2, and EN as its ID, position, and energy level. In this situation, SP1 and SP2 are similar because query sender is the station S itself. Also DP and DR fields can be null because in the first time there is no information about location of the destination D.

After receiving the query packet, each intermediate node will save information of the query sender (i.e., energy level and also sender and the station position) into its Neighbor List. The components of the Neighbor List for each neighbor are as follows:

> *Neighbor ID (NID)*: for holding the ID of a neighbor;
>
> *Energy Level (EN)*: for holding the energy level of the sender's neighbor;
>
> *Hop Count (HP)*: the number of hops from a neighbor to the station;
>
> *Sender Position (SP1)*: the position of the sender (neighbor) node;
>
> *Station Position (SP2)*: the position of the station;
>
> *Module1 (M1)*: the probability associated with a neighbor as computed by the learning automata (LA);
>
> *Module2 (M2)*: the membership degree associated with a neighbor as computed by the Fuzzy Set (FS) technique.

After inserting information of the query packet into the Neighbor List, the intermediate node checks the DP field. In first time, the DP field of the received query packet is null and the intermediate node only inserts its ID, position, and energy level and rebroadcasts the query packet. When destination D received the query packet, it will respond by sending a data packet (attached some information such as its position) to the station D; details of data packet and its routing will be described in Section 3.2.

In the next rounds of query sending, for finding destination D, the station S has knowledge about previous position of destination D. Using this knowledge can help us to limit query zone for reducing energy consumption. Although zone limitation in query broadcasting can reduce energy consumption, using small zone can lead to network

partitioning. Because in a small zone just a few nodes transfer all packets and their energy can be finished soon. Therefore, our goal is designing an effective zone that covers important parts of networks such as all neighbors of the station for effective managing of energy near the station.

If we consider that destination D at time t_0 was at location L, the station S at time t_1 computes a circular Expected Domain (ED). For instance, if node S knows that node D travels with average speed v or Velocity, then S may assume that the ED is the circular region of radius $v(t_1 - t_0)$, centered at location L [10]. It means that we expect the destination D located at one place inside this domain. Therefore, our query zone must cover this ED.

The radius is computed based on the following:

$$F = \text{Velocity} \times (t_1 - t_0) + \varepsilon. \tag{1}$$

In the equation, Velocity is the average speed. t_1 is the current time and t_0 is the time of the previous location of node D. The constant epsilon is used to keep radius nonzero in an immobility status.

Using the ED, the ERP protocol will compute a limited query zone to reduce energy consumption of broadcasting.

We propose a new query zone with an optimum size (see Figure 2).

The nodes receiving the query packet can forward or discard it depending on their location. For instance, in Figure 2, when node B receives the query originally sent from station S, it will process the packet but it will not forward it, since B is not within the query zone. Instead, node K will process and forward the query given that it is inside the zone. In brief, the nodes within the query zone distribute the queries complementing the information originally sent by the station S with their own ID, their energy level, the procedure start time, their position, hop count, distance to the station, and a list of hops to prevent forwarding loops. This process is repeated until a destination is found. To determine whether a node $B(x, y)$ is in, on, or outside our zone (the zone can be considered as a positively oriented polygon), the following equation is used:

$$A = (x_2 - x_1)(y - y_1) - (y_2 - y_1)(x - x_1). \tag{2}$$

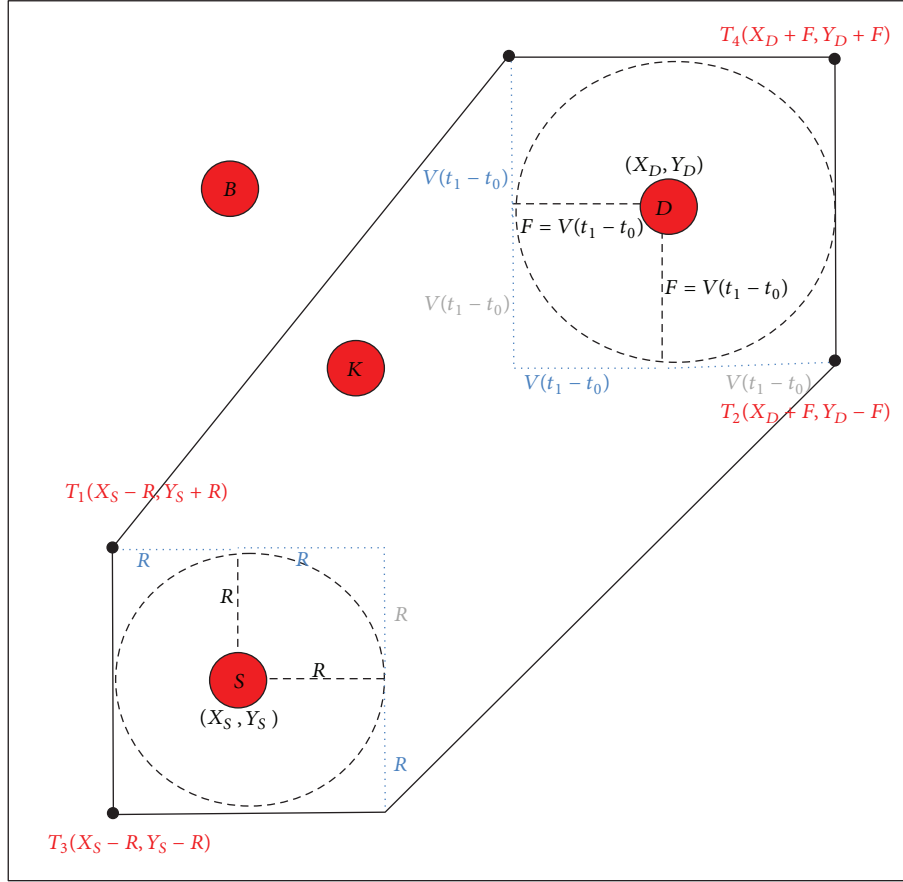If A is more than zero for all sides, node B is in the zone.

FIGURE 2: Proposed query zone.

3.2. Data Forwarding Phase.

When the destination D receives the query packet, it replies by sending a data packet. There is a single data packet format for the ERP protocol, which contains the following major fields: Station ID (SI) which is the destination of the data packet or the ID of the station, payload, an array with a size of 5 that saves the 5 last previous hops to prevent loops, energy level (EN) for forwarding the energy level of the data sender node, and sender position (SP) which gives the location of the data sender node as well as the time and destination position files that indicate the location of the destination node at a certain time so that the query zone in the Station can be estimated for a future query.

Procedure of selecting next hop in data forwarding phase is based on learning automata (LA) and fuzzy set (FS) techniques. LA and FS offer their idea about selecting next hop individually and finally one decision maker (DM) algorithm makes final decision.

As we mentioned before, Neighbor List includes M1 and M2 fields. M1 field is filled by learning automata (LA) and M2 field by fuzzy set (FS). Each node that receives a data packet runs its DM algorithm. The DM looks at the Neighbor List and finds the neighbor with highest probability in the M1 field (called $\text{ID}^{(M1)}$) and the neighbor with highest membership degree in the M2 field (called $\text{ID}^{(M2)}$). Actually, the highest probability neighbor in the M1 field is a selected neighbor

when using the learning automata technique and the highest membership degree in the M2 field is a selected neighbor by means of the FS technique. Note that if the selections made by the LA and FS techniques are the same neighbor, then that neighbor is chosen as the next hop. Otherwise, the node runs a basic sequence of tie-breaking rules until the next hop is selected.

The processing of the DM module is summarized in Algorithm 1. In this algorithm, $\mathscr{E}_{\text{ID}}^{(M1)}$ and $H_{\text{ID}}^{(M1)}$ are the energy level and hop count, respectively, of selected neighbor $\text{ID}^{(M1)}$ by means of learning automata and $\mathscr{E}_{\text{ID}}^{(M2)}$ and $H_{\text{ID}}^{(M2)}$ are the energy level and hop count, respectively, of selected neighbor $\text{ID}^{(M1)}$ by utilizing the FS technique. Method of filling M1 and M2 fields by LA and FS is described in the following.

3.2.1. Membership Degree Computation.

The theory of fuzzy sets was introduced by Zadeh in 1965 [11]. Since the pioneering work of Zadeh, there has been a great effort to obtain fuzzy analogues of classical theories. Fuzzy set theory is a powerful tool for modeling uncertainty and for processing vague or subjective information in mathematical models. Their main directions of development have been quite diverse and they have been applied to a great variety of real problems. The notion central to fuzzy systems is that truth values (in fuzzy

```
Input: {ID^(M1), 𝓔_ID^(M1), H_ID^(M1)}
Input: {ID^(M2), 𝓔_ID^(M2), H_ID^(M2)}
Output: next-hop node
(1)  foreach packet to be forwarded do
(2)      If (ID^(M1) == ID^(M2))
(3)          send the packet to the selected neighbor;
(4)          // If IDs do not match then run tie-breaking rules;
(5)      else if (𝓔_ID^(M1) > 𝓔_ID^(M2)) && (H_ID^(M1) < H_ID^(M2)) then
(6)          choose ID^(M1) as the next-hop;
(7)      else if (𝓔_ID^(M1) < 𝓔_ID^(M2)) && (H_ID^(M1) > H_ID^(M2)) then
(8)          choose ID^(M2) as the next-hop;
(9)      else choose the one with the highest energy;
```

ALGORITHM 1: The decision maker (DM) algorithm.

logic) or membership values (in fuzzy sets) are indicated by a value on the range of $[0, 1]$, with 0 and 1 representing absolute falseness and absolute truth, respectively.

The ERP algorithm considers a fuzzy set A. Fuzzy set A includes all possible candidates or neighbors. The set also has a membership function. The membership function maps each value (neighbor) to a membership value on the range of $[0, 1]$.

Membership function computation: the membership function consists of three factors, $K1$, $K2$, and $K3$, on a range of $[0, 1]$. A fairly efficient way to compute the membership degree of neighbors can be achieved by multiplying these factors together. All three factors of each neighbor will thus be multiplied together to get neighbor's membership degree. Assume there are n neighbors. Computing of the factors for the membership function of the ith neighbor is described below in more detail:

$$K_1 = 1 - \left(\frac{\text{Hop}_i}{\text{MaxHop}} \right),$$

$$K_2 = \left(\frac{\text{Energy}_i}{\text{MaxEnergy}} \right), \qquad (3)$$

$$K_3 = 1 - \left(\frac{\text{Distance}_i}{\text{MaxDistance}} \right).$$

The membership degree of the ith neighbor can then be computed based on the following function and inserted into the M_2 field of the ith neighbor:

$$M_2 = \text{Membership Function} = K_1 \times K_2 \times K_3. \qquad (4)$$

3.2.2. Probability Computation. When a node (i.e., node i) receives a query packet from a neighbor for the first time (i.e., from neighbor k), this produces a new entry in its Neighbor List. The Neighbor List is composed of fields, and each part of the data has to be stored in its related field. The LA can then compute the probability of neighbor k from the information contained in the Neighbor List received from neighbor k and store it in M1 field of the Neighbor List. The probability

$P_k(t)$ associated with neighbor k is computed according to the expression

$$P_k(t) = \frac{1}{3} \left(\frac{\mathscr{E}_k(t)}{\sum_{m=1}^{N_i} \mathscr{E}_m(t)} + \frac{1/D_k(t)}{\sum_{m=1}^{N_i} (1/D_m(t))} + \frac{1/H_k(t)}{\sum_{m=1}^{N_i} (1/H_m(t))} \right) \quad k \le N_i, \qquad (5)$$

where En is the energy level advertised by neighbor m, N is the size of node i's Neighbor List (including now node K), and D is the distance advertised by neighbor m to the station S, computed based on expression (6) where (x_1, y_1) and (x_2, y_2) are the positions of the station and the current neighbor already saved in Neighbor List. The sums in the denominators represent the terms to normalize the probabilities and to make $\sum_{k=1}^{N_i} P_k(t) = 1$,

$$D = \left((y_2 - y_1)^2 + (x_2 - x_1)^2 \right)^{1/2}. \qquad (6)$$

The rationale of using expression (5) is that it produces a good balance between energy and distance, though at the cost of the potential recomputation of the probabilities immediately after each query packet is received, since the sum of the probabilities for all neighbors must be equal to one.

3.2.3. Updating Probabilities. The basics of the mechanism are illustrated in Figure 3, and it works as follows. The M1:LA module in node A offers the neighbor with the highest probability, from the M1 field of Neighbor List, as its offered next hop (neighbor B in this case), and then it waits for final decision by the DM algorithm. If the DM selects the same node (node B in this case) as a next hop, it informs the LA (learning automata) whose offered neighbor was selected as a next hop.

Thus, in fact, the probability updating will be enabled if the LA and the DM select the same neighbor as the next hop. When node B receives the data packet, it updates the piggybacked energy level of node A in its Neighbor List and all the other neighbors of A overhear the data packet and perform the same updates as B, although they discard
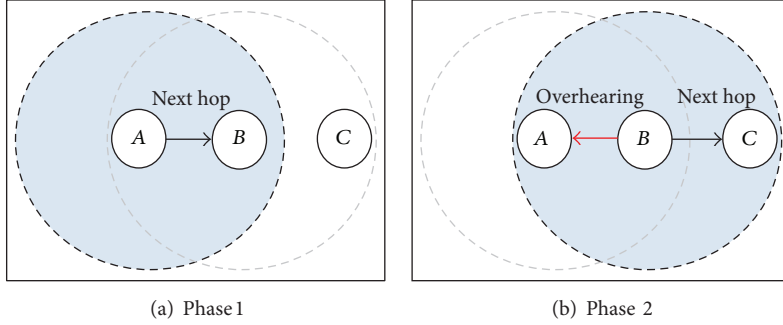
(a) Phase 1

(b) Phase 2

FIGURE 3: Updated probability scheme.

the packets immediately after processing them. The routing process continues now with node B selecting node C as its next hop. When B sends the data packet to C, node A is the one that overhears the packet sent by B; it thereby updates the energy level of the latter and then it will update probability of node B. The updating functions are based on piggybacking and overhearing techniques; they can compute and mutually update the probabilities in the Neighbor Lists according to the energy levels, hop count, and distances obtained from the neighbors. In the example, if the metrics received from node B are acceptable, then node B is rewarded by the learning automaton in A, and the probability associated with B is increased in node A's Neighbor List. Otherwise, B is penalized and its probability is decreased.

In our model, we considered four behavioral cases for rewarding or penalizing a neighbor B.

In the first case, the energy-distance-hop relationship is below the average, and thus the learning automata in A will penalize node B with a factor β, where $\langle \mathscr{E}_A(t) \rangle = \sum_{m=1}^{N_i} \mathscr{E}_m(t)/N_i$ represents the average energy of the neighbors of node A, and $\mathscr{E}_B(t)$ stands for the energy level obtained from B. Likewise, $\langle D_A(t) \rangle$ represents the average distance of the neighbors of A to the station S, while $D_B(t)$ represents the distance to S reported by node B. $\langle H_A(t) \rangle$ represents the average hop count of the neighbors of A to the station S, while $H_B(t)$ represents the hop count to S reported by node B. In the second case, we consider a lower penalization. The penalization selected is $\beta/2$. In the third case, node A will reward node B with $\alpha/2$. We consider that the best case occurs in the fourth case.

$$\text{Case 1: } \frac{\mathscr{E}_B(t)}{\langle \mathscr{E}_A(t) \rangle} + \frac{\langle D_A(t) \rangle}{D_B(t)} + \frac{\langle H_A(t) \rangle}{H_B(t)} < 3,$$

$$\text{Case 2: } \frac{\mathscr{E}_B(t)}{\langle \mathscr{E}_A(t) \rangle} + \frac{\langle D_A(t) \rangle}{D_B(t)} + \frac{\langle H_A(t) \rangle}{H_B(t)} = 3.$$

$$\text{Case 3: } 3 < \frac{\mathscr{E}_B(t)}{\langle \mathscr{E}_A(t) \rangle} + \frac{\langle D_A(t) \rangle}{D_B(t)} + \frac{\langle H_A(t) \rangle}{H_B(t)} \leq 3.5.$$

$$\text{Case 4: } 3.5 < \frac{\mathscr{E}_B(t)}{\langle \mathscr{E}_A(t) \rangle} + \frac{\langle D_A(t) \rangle}{D_B(t)} + \frac{\langle H_A(t) \rangle}{H_B(t)}.$$

(7)

Reward Computation. The reward parameter α is used during the update mechanism in order to grant more priority to the nodes giving them more possibilities to forward the response packets to the station. The value of α is computed using

$$\alpha = \lambda_\alpha + \delta_\alpha \left(\frac{\mathscr{E}_B(t)}{\langle \mathscr{E}_A(t) \rangle} \times \frac{\langle D_A(t) \rangle}{D_B(t)} \times \frac{\langle H_A(t) \rangle}{H_B(t)} \right), \quad (8)$$

where λ_α is the minimum reward granted to a well-positioned node, and δ_α is the limiting factor for the reward.

Penalty Computation. Similarly, we use

$$\beta = \lambda_\beta + \delta_\beta \left(\frac{\mathscr{E}_B(t)}{\langle \mathscr{E}_A(t) \rangle} \times \frac{\langle D_A(t) \rangle}{D_B(t)} \times \frac{\langle H_A(t) \rangle}{H_B(t)} \right)^{-1}, \quad (9)$$

where, analogously to the reward mechanism, λ_β is the minimum penalty, and δ_β is the limiting factor. Note that, in expression (8) and (9), the better (or worse) the energy-distance relationship, the greater the reward (or penalization) assigned to node B. Upon obtaining the energy and distance metrics from node B, the learning automata in node A will update the probabilities of its N_A neighbors based on (3) and (11). The former applies to the rewarding cases, that is, the third and fourth cases described above, with $x_\alpha = \alpha/2$ and $x_\alpha = \alpha$, respectively. The latter corresponds to the penalization cases, that is, the first and second cases, with $x_\beta = \beta$ and $x_\beta = \beta/2$, respectively,

$$P_B(t_{n+1}) = P_B(t_n) + x_\alpha \left[1 - P_B(t_n) \right],$$

$$P_k(t_{n+1}) = (1 - x_\alpha) P_k(t_n) \quad \forall k \mid k \neq B \wedge k \leq N_A,$$

(10)

$$P_B(t_{n+1}) = \left(1 - x_\beta \right) P_B(t_n),$$

$$P_k(t_{n+1}) = \frac{x_\beta}{N_A - 1} + \left(1 - x_\beta \right) P_k(t_n) \quad (11)$$

$$\forall k \mid k \neq B \wedge k \leq N_A.$$

4. Performance Evaluation

In this section, we evaluate the ERP's performance by comparing it to the following routing protocols: Rumor [5], as a basic query-based routing protocol, and EQR [9], as a new query-based routing protocol. To this end, we used

the GloMoSim simulator developed by UCLA [12]. The simulation model used and the results we obtained with it are described below.

4.1. Simulation Model. We used a surface that was 1000 m × 1000 m. The radio range was set to 177 m, with an available bandwidth of 2 Mbps and a radio transmission (TX) power of 4 dBm. Each simulation had a 4-hour duration, and the tests were run under various conditions, such as with different amount of sensors, namely, 1000, 1200, and 1400 nodes, and also with 10 different amounts of seeds. Moreover, the placement of the sensors in the terrain and their initial energy levels were selected randomly. It is worth highlighting that even though the placement and initial energy of the nodes were set randomly, once set, those factors remained fixed for rest of the trials to obtain comparable results across experiments. In the simulations presented here, the traffic in the network is always initiated by a source station *S*, which periodically acquires information from a particular sensor *d*. The sensor *d* moves at a speed of 40 metre/hour. Once the query is received at *d*, the sensor will immediately send back the response to *S* with the requested information.

4.1.1. Scenario I. In this scenario, we assume a critical situation, where the energy levels for transmission mode are very low. Under these conditions, we evaluate the different routing schemes considering three different tests.

Test 1: Time until the First Node Runs Out of Battery Power. This test is one of the indicators of the effectiveness of the routing schemes in terms of energy management. In general, those with the capacity to balance the energy consumed should last longer without node failure.

Test 2: Number of Nodes That Run Out of Battery Power. This test computes the total number of sensors that fail in each routing scheme during a simulation period of 2 hours, providing an indicator of the capacity of the routing schemes for saving energy.

4.2. Scenario II. In this scenario, the energy levels of the sensors are set sufficiently high so as to avoid experiencing node failures during the simulation runtime. Our goal in this case is to compare the fairness in terms of energy consumption. In order to avoid bias in the comparison, we ensure that all the routing schemes transmit the same amount of data and that this occurs without node failures. We carry out three tests to examine how the routing schemes save and manage energy in regular operation mode.

Test 3: Variance in the Remaining Energy Levels for the Neighbors of Station S. This test allows us to examine which routing scheme is best at performing energy balancing among the nodes close to the station.

Test 4: Average Energy Consumption. This test provides another indicator of which routing scheme is more efficient in managing energy.
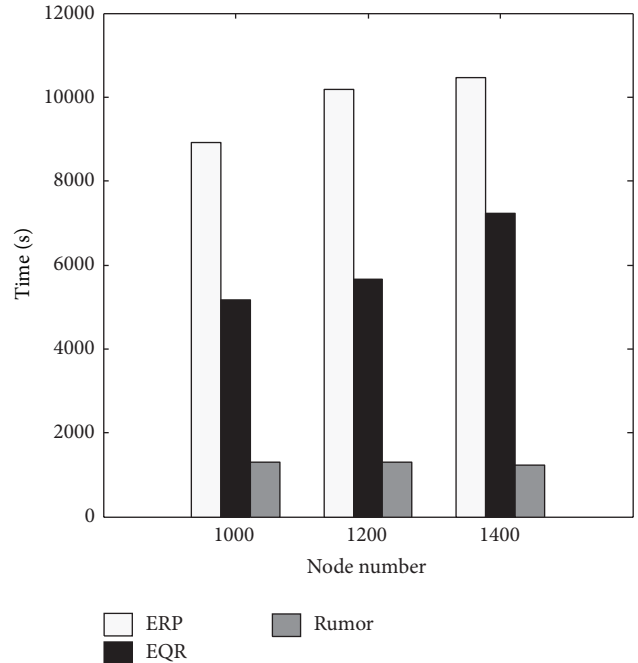

Figure 4: *Test 1:* time until the first node runs out of battery.

4.3. Simulation Results

4.3.1. Scenario I. The most commonly used measure of network lifetime is the time until the first node runs out of battery energy. In ERP, the first sensor fails after ~8900 to ~10500 seconds depending on the number of nodes present in the network (see Figure 4). The time in which the first node runs out of battery is relatively shorter for EQR and significantly shorter for Rumor.

Figure 5 shows that the ERP is much better than the more traditional Rumor and that it is relatively similar to the new EQR protocol. In brief, even though in low battery situations there is no big difference between EQR and ERP in terms of the number of node failures, it is clear that ERP can keep a network alive longer than EQR. Its main reason can be related to this point that decision-making system of ERP (LA and FS) is more accurate than EQR (only LA).

4.3.2. Scenario II. In this scenario we evaluate the quality of these protocols in normal energy situations. Figure 7 shows that Rumor is the worst routing protocol, mostly because Rumor selects its next hops randomly. There is no visible difference between ERP and EQR in energy consumption, because both protocols use zonal broadcasting. However Figure 6 shows that ERP is more successful and a better energy manager than EQR.

Actually, beyond merely comparing the particular values obtained in each figure, the most important conclusions that can be extracted from the tests as a whole are basically the following. The results show that the combination of a FS technique and LA can improve energy balancing and, more importantly, that the combined operation (ERP's use of
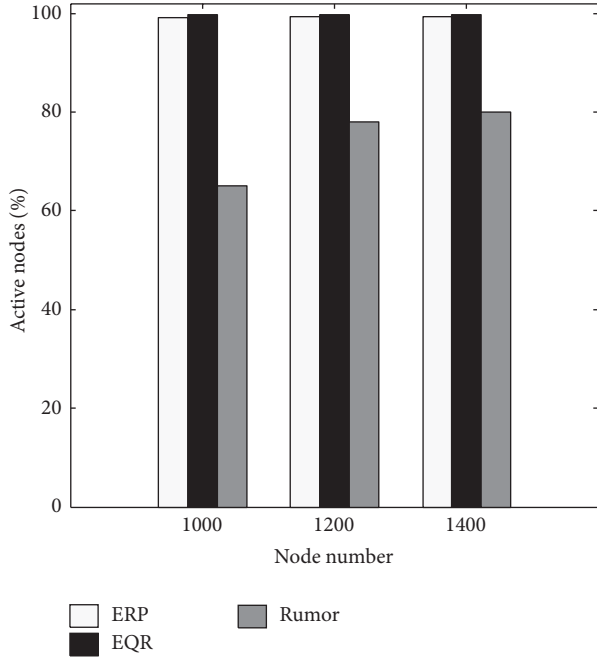
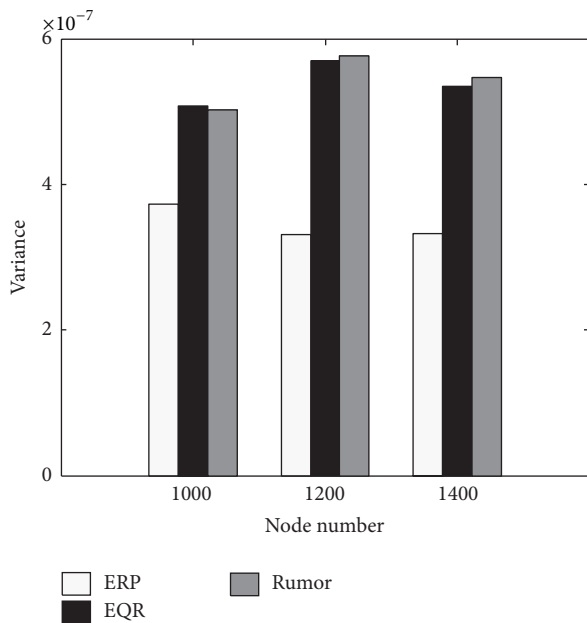Figure 5: *Test 2:* fraction of active nodes.

Figure 6: *Test 3:* variance.

fuzzy set plus learning) can work better than using only one technique (EQR).

5. Conclusion

In this paper we studied energy-aware query-based routing protocols. From the routing perspective, we have observed that the current destination-initiated query-based routing protocols can be considerably improved, especially, if we aim for a better balance between the energy savings and
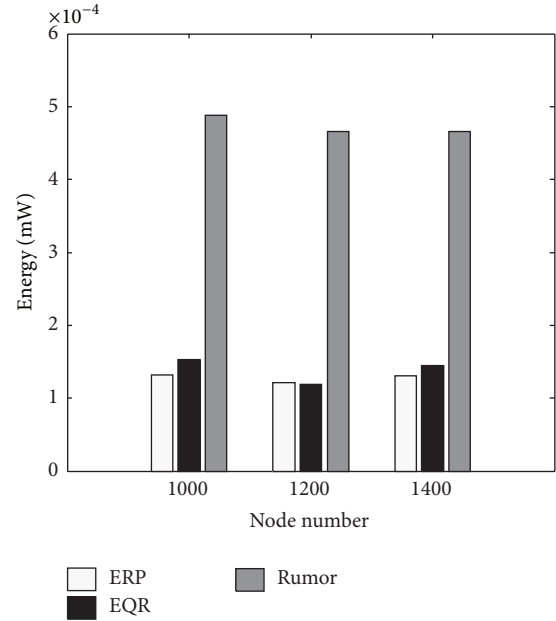

Figure 7: *Test 4:* average energy consumption.

energy balancing objectives. We have proposed a new energy saver/balancer routing protocol. We simulated and compared our routing protocol with traditional Rumor and newer EQR protocols. Indeed, four different types of tests were carried out and described, and in most of these tests it was indicated that the ERP obtained significantly better results.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] A. M. Ortiz, F. Royo, T. Olivares, J. C. Castillo, L. Orozco-Barbosa, and P. J. Marron, "Fuzzy-logic based routing for dense wireless sensor networks," *Telecommunication Systems Journal*, vol. 52, no. 1, pp. 2687–2697, 2013.

[3] Y. Chen and Y. Lin, "Mobicast routing protocol for underwater sensor networks," *IEEE Sensors Journal*, vol. 13, no. 2, pp. 737–749, 2013.

[4] X. Chen, Z. Dai, W. Li et al., "ProHet: a probabilistic routing protocol with assured delivery rate in wireless heterogeneous sensor networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 4, pp. 1524–1531, 2013.

[5] D. Braginsky and D. Estrin, "Rumor routing algorithm for sensor networks," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, pp. 22–31, Atlanta, Ga, USA, September 2002.

[6] C. Chou, J. Su, and C. Chen, "Straight line routing for wireless sensor networks," in *Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC '05)*, pp. 110–115, Murcia, Spain, June 2005.

[7] H. Shokrzadeh, A. T. Haghighat, F. Tashtarian, and A. Nayebi, "Directional rumor routing in wireless sensor networks," in *Proceedings of the 3rd IEEE/IFIP International Conference in Central Asia on Internet (ICI '07)*, Tashkent, Uzbekistan, September 2007.

[8] H. Shokrzadeh, A. T. Haghighat, and A. Nayebi, "New routing framework base on rumor routing in wireless sensor networks," *Computer Communications Journal*, vol. 32, no. 1, pp. 86–93, 2009.

[9] E. Ahvar, R. Serral-Gracia, E. Marin-Tordera, X. Masip-Bruin, and M. Yannuzzi, "EQR: a new energy-aware query-based routing protocol for wireless sensor networks," in *Proceedings of the IFIP 10th International Conference on Wired/Wireless Internet Communications (WWIC '12)*, pp. 102–113, Santorini, Greece, June 2012.

[10] Y. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," *Wireless Networks Journal*, vol. 6, no. 4, pp. 307–321, 2000.

[11] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[12] GloMoSim Simulator, http://pcl.cs.ucla.edu/projects/glomo-sim.