EQR: A New Energy-Aware Query-Based Routing Protocol for Wireless Sensor Networks*

Ehsan Ahvar[1], René Serral-Gracià[2], Eva Marín-Tordera[2], Xavier Masip-Bruin[2], and Marcelo Yannuzzi[2]

[1] Department of Information Technology and Communication
Payame Noor University, Iran
ahvar@pnu.ac.ir
[2] Advanced Network Architectures Lab (CRAAX)
Technical University of Catalunya (UPC), Spain
{rserral,eva,xmasip,yannuzzi}@ac.upc.edu

Abstract. Over the last years, a number of query-based routing protocols have been proposed for Wireless Sensor Networks (WSNs). In this context, routing protocols can be classified into two categories, energy savers and energy balancers. In a nutshell, energy saving protocols aim at decreasing the overall energy consumed by a WSN, whereas 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 and vice versa. In this paper, we introduce an Energy-aware Query-based Routing protocol for WSNs (EQR), which offers a good trade-off between the traditional energy balancing and energy saving objectives. This is achieved by means of learning automata along with zonal broadcasting so as to decrease the total energy consumption. We consider that, in the near future, EQR could be positioned as the routing protocol of choice for a number of query-based WSN applications, especially for deployments where the sensors show moderate mobility.

Keywords: WSN, query-based routing, energy.

1 Introduction

Wireless Sensor Networks (WSNs) are bringing unprecedented abilities to monitor different types of physical phenomena, and thereby control specific operations or processes without human intervention. In the near future, a large number of low-power and inexpensive sensor devices will become part of the landscape in cities, highways, farms, airports, etc. Since the deployment and operation of these networks is application dependent, the possibility of having a unified routing strategy capable of meeting the requirements of all foreseeable applications is simply unfeasible—e.g., consider the difference in terms of routing requirements between a fixed WSN for an urban monitoring application and those of a mobile WSN for finding survivors trapped after a natural disaster.

* This work was supported in part by the Spanish Ministry of Science and Innovation under contract TEC2009-07041, and the Catalan Research Council (CIRIT) under contract 2009 SGR1508.

Y. Koucheryavy et al. (Eds.): WWIC 2012, LNCS 7277, pp. 102–113, 2012.

Indeed, the design and implementation of routing schemes able to efficiently support the exchange of information—basically by saving processing overhead and thus energy—is particularly challenging in WSNs. A number of theoretical and practical limitations must be thoroughly examined and considered during the design and the implementation phases, including the methods for route discovery and maintenance, the methods for handling losses and failures as well as potential radio interferences during network operation. Until now, many routing algorithms and protocols have been proposed for WSNs (see, for instance, [1], [2], [3]), among which we found a category known as *query-based routing*. In this category, a station S sends queries to find specific events among the sensor network (e.g., a query may encode a question such as: is the measured temperature higher than 40C?). In this context, the strategies used for routing the queries and their corresponding replies can be classified into two major groups, namely, energy savers and energy balancers. The former try to decrease the energy consumption of the network as a whole and thereby increase the operation lifetime—usually leading to the utilization of shortest paths. The latter, on the other hand, try to balance the energy consumption of the nodes in order to prevent the potential partitioning of the network. Overall, finding the best route based only on energy balancing objectives may lead to rather long paths and increased delays, whereas finding the best route based only on energy saving and optimal distance objectives may lead to network partitioning.

In light of this, we propose a routing strategy applicable to various forms of query-based applications that offers a reasonable trade-off between the energy saving and energy balancing objectives. More precisely, we propose an Energy-aware Query-based Routing protocol for WSNs called EQR, which is supported by learning automata and uses zonal broadcasting to decrease the total energy consumed. Our initial results demonstrate the potential and effectiveness of EQR, making it a promising candidate for a number of WSN applications, especially those where the sensors have moderate mobility.

The rest of the paper is organized as follows. In Section 2, we overview related work. Section 3 presents the main contribution of this paper which is basically the EQR routing strategy. The assessment of EQR is covered in Section 4, and finally, Section 5 concludes the paper.

2 A Brief Outline of Related Work

One of the references in the subject of energy-aware query-based routing protocols is RUMOR [4]. RUMOR is an energy saving protocol that provides an efficient mechanism combining push and pull strategies to obtain the desired information from the network. In RUMOR, the nodes generating events send notifications that leave a “sticky” trail along the network. Then, when query agents visit a node where an event notification agent has already passed, they can find pointers (i.e., the trail) toward the location of the corresponding source. In general terms, when a node receives a query two things can happen: i) the node has already 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 visiting them again.

Clearly, the forwarding strategy in RUMOR might end up producing spiral paths, so the intuition for improving RUMOR is to reduce its level of routing indirection. To this end, Cheng-Fu Chou et al. proposed in [5] the Straight Line Routing (SLR) protocol, which aims at making the routing path grow as straight as possible. In recent years, Shokrzadeh et al. have made significant efforts to improve the basis of RUMOR in different aspects, and proposed Directional RUMOR (DRUMOR) [6]. Later on, Shokrzadeh et al. improved their DRUMOR protocol by means of what they called Second Layer Routing (SecondLR) [7]. The latter uses geographical routing right after locating the source of an event, and the authors have shown that this approach considerably improves the performance of DRUMOR.

Despite these efforts, current query-based routing protocols are mainly energy savers, and show relatively poor performance when it comes to balance the energy consumption. Although several papers can be found in literature addressing the issue of energy balancing in the context of routing protocols for WSNs, to the best of our knowledge, our work is the first attempt toward a better balance of the energy consumed in destination initiated query-based routing protocols.

3 Energy-Aware Query-Based Routing (EQR)

3.1 Overview

EQR is a routing scheme especially designed to consider both energy and distance while routing packets across a network. To this end, EQR balances whenever possible the load among the different sensors with a twofold goal: avoid that the sensors run out of battery while still keeping the routes to reach the destinations relatively short. As shown in Fig. 1, the architecture that supports EQR is basically composed of an API, a Zone Estimation Unit (ZEU), a Probability Computation Unit (PCU), a Data Path Selection Unit (DPSU), and an Error Detection Unit (EDU). In this architecture, the

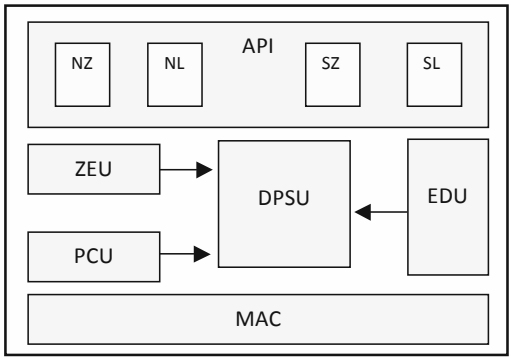


Fig. 1. The architecture supporting the EQR routing protocol

EQR routing protocol provides four application-level API modes called Non-specific Zone (NZ), Non-specific Location (NL), Specific Zone (SZ), and Specific Location (SL). The SZ and NZ modes are for event monitoring: the former for applications that monitor events in a specific zone of the network, while the latter is used when there is no prior knowledge about the area where such events occur. In this second mode, the query packets must be broadcasted throughout the entire network to find potential events. The two remaining modes, i.e., SL and NL, are designed for querying a given node and getting information directly from it: the first when there is prior knowledge about the expected location of the node, and the second when its location is unknown.

The ZEU unit is responsible for estimating the query zone, and similarly to [8], it is based on a low-power GPS module which is actively powered on and off in order to save energy. As its name indicates, the EDU is designed for detecting errors, while the DPSU is the core routing module and hence the one responsible for choosing the next-hop during the packet forwarding process. In EQR, the neighbor with the highest probability is designated as the "primary" neighbor, and it is the one that will be chosen by the DPSU as the next-hop. The probability associated to each neighbor is computed by the PCU unit, and the process for assigning and updating these probabilities shall be described in detail along this section. As shown in Fig. 1, the constellation of modules is mainly designed to assist the DPSU unit. Due to space limitations, we cannot develop in detail all the modules in the architecture, so the main focus of this work is centered on the DPSU and PCU units, leaving the description of the rest of the architecture and its building blocks for an extended version of this paper.

In EQR, the design of the DPSU and PCU are based on learning automata, where we also integrate other well-known strategies, such as piggybacking and overhearing techniques. More precisely, the next-hop in EQR is chosen using learning automata, which compute the probabilities and select the best possible neighbor considering two metrics: their energy level as well as the distance to reach the destination.

3.2 Initialization Process

When the WSN starts its operation, the ZEU unit in the station S that will issue the queries will lack any zonal information. Therefore, the query modes used by the station S at the beginning of the operations will typically be NL or NZ, which means that the entire region is assumed to be the "query zone". Once the ZEU starts collecting information, the subsequent queries issued by the station S can be made using the SZ or SL modes, thus exploiting the advantages of zonal broadcasting. In general terms, the station S will gather zonal information in its ZEU unit, and whenever required it will generate a query packet in which it will broadcast its ID, S_{ID}, the query mode (NZ, NL, SZ, or SL), its position, $(x_S(t), y_S(t))$, the zone information, and optionally, the destination ID, d_{ID}. The zone is defined in the query packet as a rectangle specified by four corners, so that the nodes receiving the query can forward or discard the same depending on their location. For instance, on the left-hand side of Fig. 2, when node j receives the query originally sent from the station S and forwarded by node k, it will process the packet but it will not forward it, since j is not within the "query zone" delimited by the four corners. Instead, nodes i and k will process and forward the query given that they are inside the zone determined by the ZEU unit.

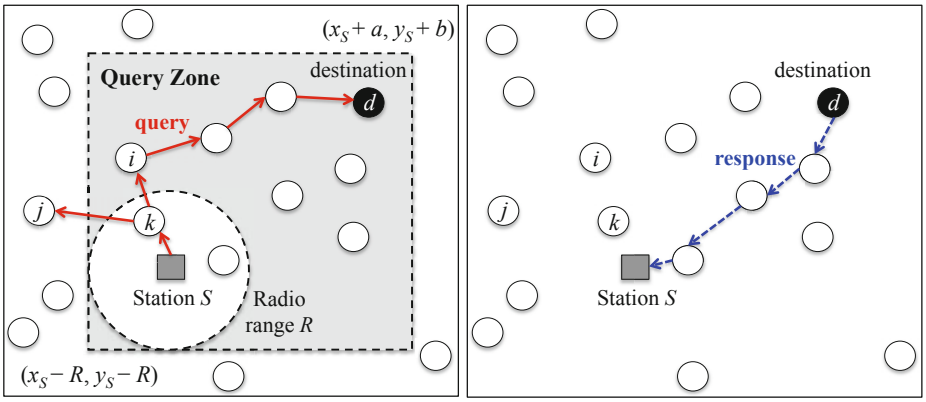


Fig. 2. (Left-hand side) Zonal broadcasting: nodes i and k will process and forward the query since they are inside the "query zone" determined by the ZEU, while node j will discard the query right after its processing. (Right-hand side) The response path is determined by the DPSU.

When a node i receives a query for the first time from a neighbor k, this produces a new entry in its "Neighbor List"— observe that the neighbor may also be the station S if node i is within the radio range of S. The Neighbor List is composed of four fields, and the data to be stored into these fields can be computed from the information contained in the query packet received from neighbor k. The fields in the Neighbor List are basically the following: 1) the neighbor's ID, k_{ID}; 2) its energy level at time t, $\mathcal{E}_k(t)$ (except for $k = S$, since the station's energy is assumed to be inexhaustible); 3) its position, $(x_k(t), y_k(t))$; and 4) the probability $P_k(t)$ associated with neighbor $k \mid k \neq S$, which is computed according to expression (1). As mentioned above, the probability $P_k(t)$ is computed using the PCU unit, where $\mathcal{E}_m(t)$ is the energy level advertised by neighbor m, N_i is the size of node i's Neighbor List (including now node k), $D_m(t)$ is the distance advertised by neighbor m to the station S, and the sums in the denominators represent the terms to normalize the probabilities and to make $\sum_{k=1}^{N_i} P_k(t) = 1$. The rationale of using (1) is that it produces a good balance between energy and distance, though at the cost of the potential recomputation of the probabilities right after receiving each query, since the sum of the probabilities for all neighbors must be equal to one.

$$P_k(t) = \frac{1}{2}\left(\frac{\mathcal{E}_k(t)}{\sum_{m=1}^{N_i}\mathcal{E}_m(t)} + \frac{\frac{1}{D_k(t)}}{\sum_{m=1}^{N_i}\frac{1}{D_m(t)}}\right) \qquad \forall k \neq S \mid k \leq N_i \tag{1}$$

In a nutshell, 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, their position and distance to the station, and a list of hops to prevent forwarding loops. This process is repeated until the event is found in a node d. At this point, every node in the zone knows the energy levels of their neighbors and the distance from them to the station S. As shown on the right-hand side of Fig. 2, the response to the station could use a different path, since this will depend on the primary neighbor chosen by node d. The response packets from d to S will follow the highest probability path, since every

node in the query zone has already chosen its primary neighbor to reach the station S. Overall, the queries are routed using zonal broadcasting while the responses are routed through the highest probability chain. This approach not only simplifies the design of the sensors but also saves a considerable amount of their energy.

The rest of this section shall be mainly devoted to describe the role of the learning automata and the process for updating the probabilities in the Neighbor List.

3.3 The Update Mechanism

The basics of the mechanism are illustrated in Fig. 3, and it works as follows. A node A selects the node with the highest probability as its primary neighbor (node B in this case), and then it sends the response packet to it. As shown in Fig. 2, this response corresponds to a query previously issued by the station S. An important feature in EQR is that every node along the highest probability chain piggybacks its energy level, its position, and its distance to the station S in the response packet, using a set of dedicated fields to this end. Thus, when node B receives the response packet, it updates the energy level and position of node A in its Neighbor List, and it may also update the probability $P_A(t)$ depending on the energy and distance reported by node A. As shown on the left-hand side of Fig. 3, all the other neighbors of A overhear the response and perform the same updates as B, though they discard the packets right after processing them. The routing process continues now with node B selecting node C as its highest probability neighbor. When B sends the response to C, its is now node A the one that overhears the packet sent by B, and thereby updates the position and energy level of the latter (cf. the right-hand side of Fig. 3).

Based on piggybacking and overhearing techniques, the nodes can compute and mutually update the probabilities in their Neighbor Lists according to the energy levels and distances obtained from their neighbors. In the example below, if the metrics received from node B are acceptable, then node B is rewarded by the learning automaton in A, and the probability associated to B is increased in node's A 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. First, the worst case occurs when:

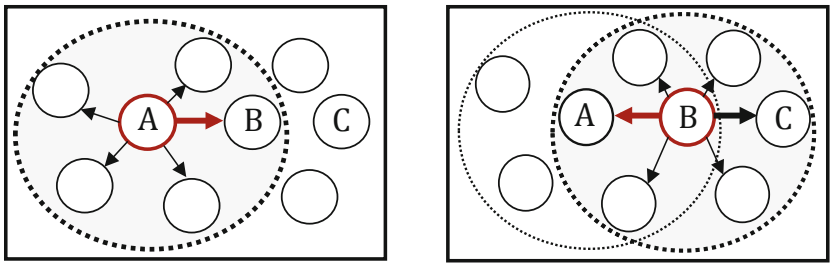


Fig. 3. Mutual updates of the energy levels, distances, and probabilities. (Left-hand side) The neighbors of node A perform the updates. (Right-hand side) Idem for the neighbors of node B.

$$\frac{\mathcal{E}_B(t)}{<\mathcal{E}_A(t)>} \times \frac{<D_A(t)>}{D_B(t)} < 1 \tag{2}$$

where $<\mathcal{E}_A(t)>= \sum_{m=1}^{N_A} \mathcal{E}_m(t)/N_A$ represents the average energy of all neighbors of node A, and $\mathcal{E}_B(t)$ stands for the energy level obtained from B. Likewise, $<D_A(t)>$ represents the average distance of all neighbors of A to the station S, while $D_B(t)$ represents the distance to S reported by node B. In this first case, the energy–distance relationship is below the average, and thus the learning automaton in A will penalize node B with a factor β—more details about this computation can be found later in Section 3.4. Second, a relatively bad situation occurs when:

$$\frac{\mathcal{E}_B(t)}{<\mathcal{E}_A(t)>} \times \frac{<D_A(t)>}{D_B(t)} = 1 \tag{3}$$

In this case, the penalization chosen is $\beta/2$. Third, if the relationship is such that:

$$1 < \frac{\mathcal{E}_B(t)}{<\mathcal{E}_A(t)>} \times \frac{<D_A(t)>}{D_B(t)} < 1.5 \tag{4}$$

then node A will reward node B with $\alpha/2$ — the details about this computation are described in Section 3.4. And fourth, we consider that the best case occurs when:

$$\frac{\mathcal{E}_B(t)}{<\mathcal{E}_A(t)>} \times \frac{<D_A(t)>}{D_B(t)} \geq 1.5 \tag{5}$$

where the reward chosen in this case is α. We proceed now to describe in more detail the incentive mechanism outlined above.

3.4 The Incentive Mechanism

As previously discussed, to incentivize the selection of next-hops that show a good energy–distance relationship, we designed a simple yet effective mechanism for rewarding or penalizing neighbors.

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

$$\alpha = \lambda_\alpha + \delta_\alpha \left(\frac{\mathcal{E}_B(t)}{<\mathcal{E}_A(t)>} \times \frac{<D_A(t)>}{D_B(t)} \right) \tag{6}$$

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:

$$\beta = \lambda_\beta + \delta_\beta \left(\frac{\mathcal{E}_B(t)}{<\mathcal{E}_A(t)>} \times \frac{<D_A(t)>}{D_B(t)} \right)^{-1} \tag{7}$$

where analogously to the reward mechanism, λ_β is the minimum penalty, and δ_β is the limiting factor. Note that in (6) and (7), the better (worse) the energy–distance relationship the greater the reward (penalization) assigned to node B.

Upon obtaining the energy and distance metrics from node B, the learning automaton in node A will update the probabilities of its N_A neighbors based on equations (8) and (9). The former applies for the rewarding cases, i.e., 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.

$$\begin{cases} P_B(t_{n+1}) = P_B(t_n) + x_\alpha[1 - P_B(t_n)] \\ P_k(t_{n+1}) = (1 - x_\alpha)P_k(t_n) \qquad\qquad \forall k \mid k \neq B \wedge k \leq N_A \end{cases} \tag{8}$$

$$\begin{cases} P_B(t_{n+1}) = (1 - x_\beta)P_B(t_n) \\ P_k(t_{n+1}) = \dfrac{x_\beta}{N_A - 1} + (1 - x_\beta)P_k(t_n) \qquad \forall k \mid k \neq B \wedge k \leq N_A \end{cases} \tag{9}$$

A summary of the collective processing performed by EQR inside the query zone is shown in Fig. 4.

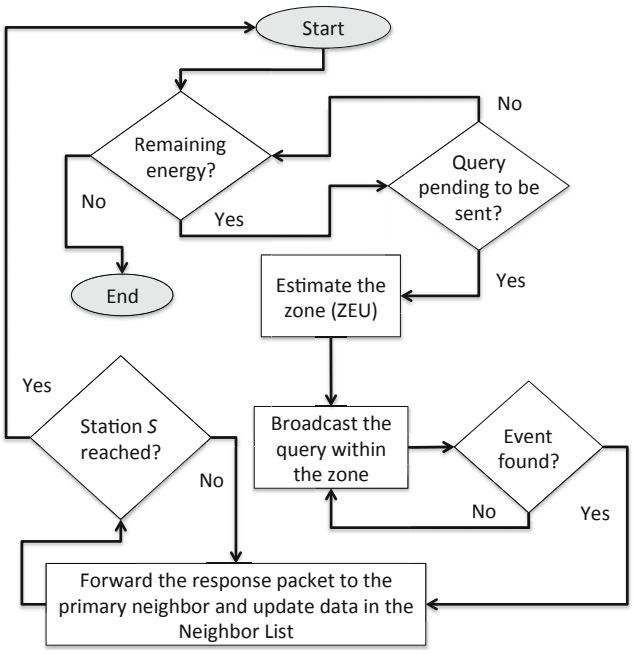


Fig. 4. Collective processing performed by the nodes inside the query zone

4 Performance Evaluation

In this section, we evaluate the performance of EQR by comparing against the following set of routing protocols: RUMOR [4], Straight Line Routing (SLR) [5], Directional Rumor (DRUMOR) [6], and Second Layer Routing (SecondLR) [7]. To this end, we used the Glomosim simulator developed by UCLA [9]. We proceed now to describe the simulation model used and the results obtained.

4.1 Simulation Model

During the simulations we made the following assumptions. We used a surface of terrain of 1000 m $\times$ 1000 m, where the sensors have a maximum energy level of $7 \times 10^{-4} mW$. The radio range was set to 177 m, with an available bandwidth of 2 Mbps and a radio TX power of 4.0 dBm using IP over 802.11. The duration of each simulation was of 4 hours, and the tests were run under various conditions, such as with different amount of sensors, namely, 800, 1000, and 1200 nodes. Moreover, the placement of the sensors in the terrain was randomly chosen. It is worth highlighting that, even though the placement of the nodes was random, once it was set it remained fixed for rest of the trials to obtain comparable results across experiments. In the simulations presented here, the traffic in the network was always initiated by a source station S, which periodically acquires information from a particular sensor d. Once the query is received at d, the sensor will immediately send back the response to S with the requested information.

4.2 Simulation Results

To evaluate the performance of our protocol we have carried out five different tests:

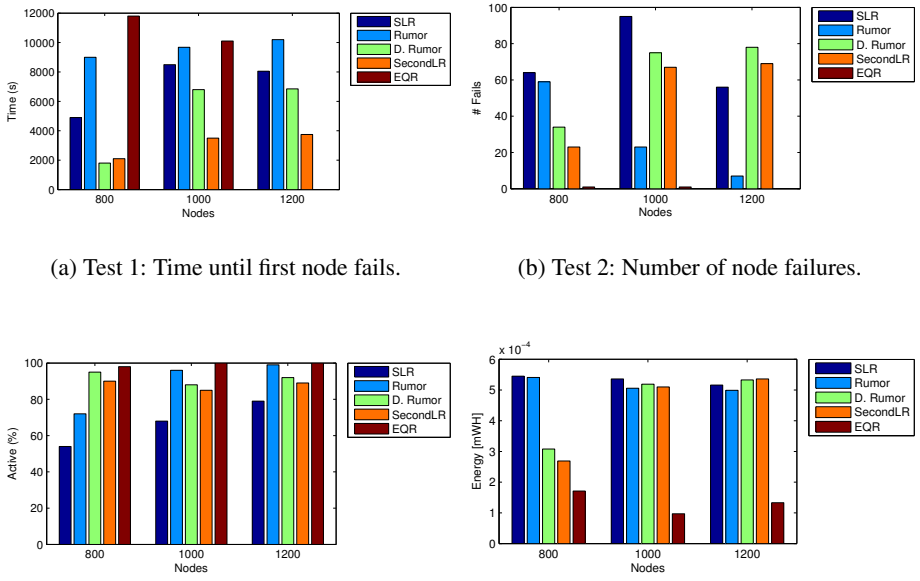
- *Test 1: Time until the first node fails*. This test is one of the indicators of the effectiveness in terms of energy management. In general terms, energy balancing protocols should last longer without failing nodes than others.
- *Test 2: Number of nodes that fail*. This test computes the number of sensors running out of power for each protocol, and therefore provides and indicator of the capacity of the protocols for saving energy.
- *Test 3: Percentage of active neighbors of the station S at the end of the simulation*. This test shows the ability of the routing protocols to keep the station S connected.
- *Test 4: Average energy consumption*. This test provides another indicator of which protocol is better at managing energy.
- *Test 5: Remaining energy level of the neighbors of the source station S*. This test provides a third indicator highlighting which protocol is able to perform better energy balancing among the nodes close to the station.

The results obtained for Tests 1 to 4 are shown in Fig. 5, while the ones for Test 5 are shown in Fig. 6. As it can be observed, in most of the tests EQR outperforms the rest of the routing protocols. In particular, for Tests 1 and 2 and 1200 nodes (Figs. 5(a)

and 5(b)), EQR does not experience any node failure during the whole simulation runtime, while showing better performance than the other protocols for lower number of nodes. In Figs. 5(a) and 5(b), the protocol with the closest performance to EQR is RUMOR. However, the main issue with RUMOR is that when a query is issued, rather than flooding it throughout the network, it is sent on a random walk until the event path is found. As soon as the query discovers the event path, it can be routed directly to the event. Opposed to that, if the path cannot be found, the application can resubmit the query, or, as a last resort, flood the network with it. Therefore, RUMOR could in the best case be seen as an energy saver protocol, but as shown in Fig. 5(d), compared to EQR it lacks functionality for energy balancing.

Figure 5(d) shows that DRUMOR can provide a better energy balance than RUMOR, though this comes at the cost of a relatively poor performance in terms energy saving (see Figs. 5(a) and 5(b) together). One of the main penalties in DRUMOR is that if the length of the query path is too short, it might not reach its destination, so the source could need to flood the network with the query packet again.

In the case of SLR, the protocol selects the next-hop from a special region. This implies that the nodes in this special region will have higher energy consumption. This is reflected in Fig. 5(b) by a higher amount of node failures, which is also reflected in the decrease suffered in the percentage of active neighbors of the source station in Fig. 5(c). Moreover, SLR selects the next-hop in two steps [5]; the first step targets the selection of a “candidate region”, while the second chooses one node from this region



(a) Test 1: Time until first node fails.

(b) Test 2: Number of node failures.

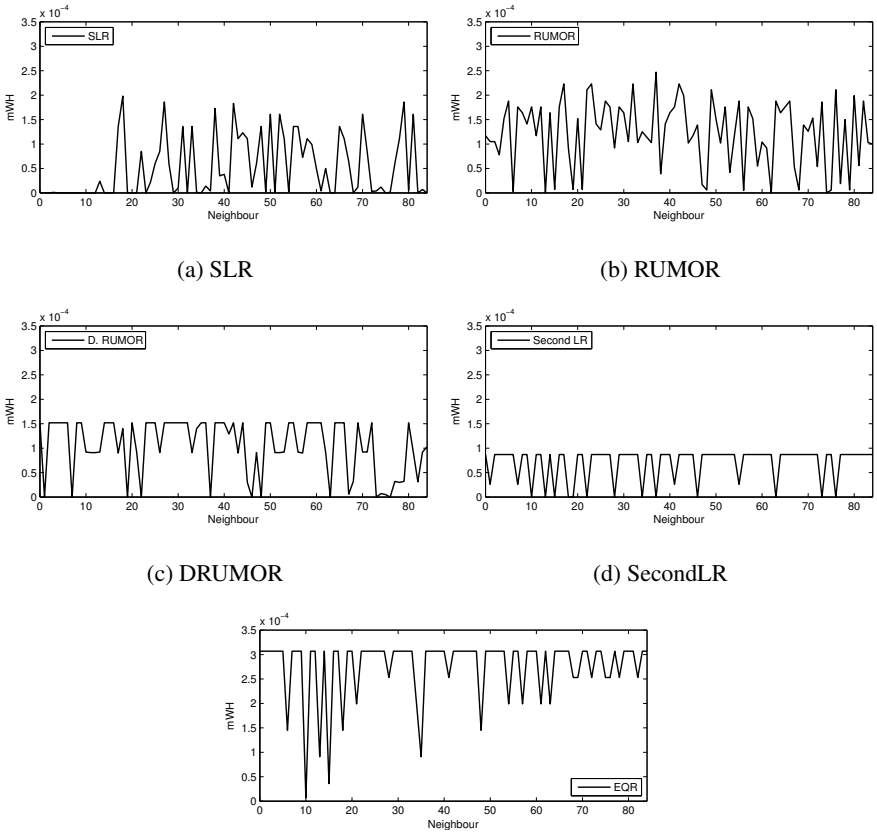
(c) Test 3: Percentage of active neighbors.

(d) Test 4: Average energy consumption.

Fig. 5. Simulation results for Tests 1-4

as the next-hop. To achieve this, the sender issues a route request and every node in the candidate region sets its own timer T_{wait}. This timer will determine which node could become the next-hop, since when T_{wait} expires the node will issue a message to notify its neighbors. The side effect of this behavior is that all these notification messages—which will be received by the rest of the nodes in the candidate region—are responsible for a higher energy consumption in SLR than in other protocols (see Fig. 5(d)). Indeed, the main idea behind SLR is to keep a straight routing path (supported by the definition of the candidate region), where the longest distance of every hop is less than a half of the transmission radius. As a consequence, if the length of the query path is short, this may lead to a lower hit rate, deriving in a larger number of broadcasts and thus more energy consumption as reflected in Fig. 5(d).

Concerning SecondLR, it can be observed from Fig. 5 that, except for Test1, SecondLR performs in general terms better than DRUMOR, but clearly its performance is much worse than EQR.



(a) SLR (b) RUMOR

(c) DRUMOR (d) SecondLR

(e) EQR

Fig. 6. Test 5: Remaining energy levels of the neighbors of the source station

Finally, the results for Test 5 are shown in Fig. 6. The latter reflects the remaining energy level of the neighbors of the source station for all the protocols assessed. As it can be observed, EQR is the most efficient protocol in terms of energy balancing. Even though other results are not reported here, it is worth mentioning that we have found that EQR also provides a good trade-off between latency and energy management. This is an important observation, since strict energy balancing protocols tend to use longer paths which usually lead to higher latencies.

5 Conclusion

Energy management is one of the central concerns in WSNs, and therefore it has been the subject of study of a large number of research works. From the routing perspective, we observe that current destination initiated query-based routing protocols can be considerably improved, especially, if we target a better balance between the energy saving and energy balancing objectives. In this line, we have presented EQR, a routing protocol which in our opinion is a promising candidate to achieve this goal. We have shown that EQR outperforms other (state of the art) destination initiated query-based routing protocols, such as RUMOR [4], SLR [5], DRUMOR [6], and SecondLR [5]. Indeed, five different types of tests were carried out and described in this paper, and in all of them the results obtained with EQR were significantly better. At this stage, the strengths of EQR are mainly limited to relatively static WSN deployments. We plan to extend our work to WSNs exhibiting more mobility among its sensors.

References

1. Akkaya, K., Younis, M.: A survey on routing protocols for wireless sensor networks. Ad Hoc Networks 3, 325–349 (2005)
2. Al-Karaki, J.N., Kamal, A.E.: Routing Techniques in Wireless Sensor Networks: A Survey. IEEE Wireless Communications Journal 11(6), 6–28 (2004)
3. Karl, H., Willig, A.: Protocols and Architectures for Wireless Sensor Networks. John Wiley & Sons, Ltd. (2005) ISBN: 0-470-09510-5
4. Braginsky, D., Estrin, D.: Rumor Routing Algorithm for Sensor Networks. In: Proceedings of the First ACM Workshop on Sensor Networks and Applications, Atlanta, GA, USA (2002)
5. Chou, C.F., Su, J.J., Chen, C.Y.: Straight Line Routing for Wireless Sensor Networks. In: 10th IEEE Symposium on Computers and Communications, Murcia, Spain (2005)
6. Shokrzadeh, H., Haghighat, A.T., Tashtarian, F., Nayebi, A.: Directional Rumor Routing in Wireless Sensor Networks. In: 3rd IEEE/IFIP International Conference in Central Asia on Internet, Tashkent, Uzbekistan (September 2007)
7. Shokrzadeh, H., Haghighat, A.T., Nayebi, A.: New Routing Framework Base on Rumor Routing in Wireless Sensor Networks. Computer Communications Journal 32 (January 2009)
8. Buchli, B., Sutton, F., Beutel, J.: GPS-Equipped Wireless Sensor Network Node for High-Accuracy Positioning Applications. In: Picco, G.P., Heinzelman, W. (eds.) EWSN 2012. LNCS, vol. 7158, pp. 179–195. Springer, Heidelberg (2012)
9. Glomosim Simulator: `http://pcl.cs.ucla.edu/projects/glomosim`