Receiver-oriented Load-balancing and Reliable Routing in Wireless Sensor Networks

Min Chen 1  Victor C.M. Leung 1  Shiwen Mao 2  Taekyoung Kwon *  †

1 Dept. of Electrical and Computer Engineering, The Univ. of British Columbia, Vancouver, Canada
2 Dept. of Electrical and Computer Engineering, Auburn University, Auburn, USA

Summary

Routing protocols in wireless sensor networks commonly employ a transmitter-oriented approach whereby the next hop node is selected based on previously collected neighbor information. This approach incurs a large overhead when accurate neighbor information is needed for efficient and reliable routing. In this paper, a novel receiver-oriented load-balancing and reliable routing (RLRR) protocol is proposed. In RLRR, a forwarding node requests the next hop and eligible neighbors self-select using their temporal gradients evaluated locally without any central coordination. Therefore, each node needs not maintain any neighbor information except the identity of the next hop node self-selected, and the energy information used in the routing decision is always accurate and up-to-date. Furthermore, peer neighbors whose hop count is equal to the upstream node can participate in next-hop contention with a mechanism that guarantees loop-free operation. Comprehensive simulations are carried out to show that RLRR exhibits relatively longer lifetime and higher reliability than several existing schemes.

KEY WORDS: Load balancing, reliability, energy efficient, wireless sensor network.

1. Introduction

Advances in microelectronics enable implementations of inexpensive sensors that can be deployed in a large scale in harsh environments, where sensors need to operate unattended in an autonomous manner. As sensor nodes communicate over error-prone wireless channels with limited battery power, reliable and energy-efficient data delivery is crucial. These characteristics of wireless sensor networks (WSNs) make the design of routing protocols challenging.

To address such issues, a lot of research focuses on prolonging the network lifetime by exploiting energy-efficiency (e.g., with load balancing), supporting reliability, or achieving low-cost sensor design. However, these design goals are generally orthogonal to each other. Specifically, most load balancing schemes are not robust to high link failure rate. In this paper, we classify existing load balancing schemes into two categories: local load-balancing [1, 2, 3] and global load-balancing [4], which can also be referred as hop-by-hop balancing and end-to-end balancing, respectively. To evaluate the performance in terms of load balancing, we define the “lifetime” of a WSN as the time until the first node in the WSN drains its battery and dies.

Local load-balancing is based on a “node-centric” approach, where a hello message is broadcast by
each sensor node periodically during the network operation, in order to notify its neighbors of its energy changes. The interval of broadcasting such a hello message provides a tradeoff between control overhead and timeliness of local energy information. Local load-balancing may cause a data packet to enter an “energy bottleneck” region, where the energy levels of all the available sensor nodes are relatively low while the sensors outside the region may have plenty of battery capacities. Thus, some schemes aim to find globally load balanced paths to achieve a higher network lifetime.

In [4, 1], to reduce packet losses due to frequent link failures, a forwarding node also uses alternative (backup) nodes by setting up multiple backup next hop nodes in advance. If the primary next hop node fails, the neighbor table becomes inaccurate but the predetermined primary node will still be inappropriately selected. In this case, the medium access control (MAC) layer is not able to deliver the packet to this unreachable primary node. After several retransmission attempts, the MAC layer either simply drops the packet or notifies the network layer of the failed transmissions and passes the packet back. The routing protocol in turn selects a different backup next hop and hands the packet again down to the MAC layer. This process is repeated until the packet can be delivered eventually to some next hop node. In the case of a high node failure rate, this scheme based on data caching and trying multiple backup nodes severely increases the delay, reduces the effective available bandwidth, and wastes energy for unnecessary transmissions. Nevertheless, the above operation is widely used in traditional routing protocols for ad hoc and sensor networks [1, 4], which operate in the following two-step manner: 1) select the next hop node first based on neighbor information table; 2) forward packet to the selected node. We call this a “transmitter-oriented” approach in this paper. Before executing step 1, the current node must acquire information about its neighbors and/or available paths. In step 2, data caching, multiple backup nodes and retransmission can be exploited to provide reliability by trading off energy-efficiency and delay.

In this paper, a novel “receiver-oriented” load-balancing reliable routing protocol (RLRR) is proposed. RLRR aims to achieve both design objectives of load balancing and reliability simultaneously for large-scale WSNs. Instead of the aforementioned transmitter-oriented approach, RLRR adopts a receiver-oriented approach to achieve load balancing and high reliability. In the receiver-oriented approach, route discovery message is relayed by means of broadcasting, such that all the live neighbors with a good signal-noise-ratio (SNR) receive the message, and the responsibility of next-hop-selection is shifted to these candidates. Each neighbors eligibility as a next hop is evaluated by the temporal gradient mapped to the neighbors energy information and hop count. The neighbor with the lowest temporal gradient will self-select as the next hop by returning a route response message before other neighbors. To shorten the selection latency, the gaps between temporal gradients should be set as small as possible while large enough to effectively differentiate multiple next hop candidates at the MAC layer. Without central coordination, the proposed mechanism self-select the candidate with the least temporal gradient to deliver data packets toward the sink and suppress the other candidates.

The receiver-oriented approach employed by RLRR has the following advantages: (1) The protocol is stateless. No hello message beaconing is needed to update neighbor energy information periodically. Though position-based routing protocols [8] are also stateless and only require local rerouting in case of link failure, RLRR does not even require that intermediate nodes maintain neighbor information except for the identifier of the next hop node; (2) Energy information is used locally in each candidate next hop node in load-balanced routing decision, and hence is always accurate and up-to-date; (3) Peer neighbors whose hop count is equal to that of the upstream node can be exploited to increase reliability and facilitate load balancing while guaranteeing loop-free routing. Though the receiver-oriented idea is close to GeRaF [6] and ExOR [7] where efficient methods of using multi-receiver diversity for packet forwarding are explored, RLRR does not rely on geographical information provided by expensive GPS devices.

We present extensive simulations to show that RLRR normally yields a higher reliability than EDDD [4], DD [5] and GEAR [1]. More importantly, RLRR also achieves a longer network lifetime. The overall performance gain of RLRR, taking into account of reliability, lifetime, and data delivery latency, increases as link failure rate increases.

The rest of this paper is organized as follows. We describe RLRR design issues and the algorithm in Sections 2 and 3, respectively. Simulation model and experimental results are presented in Sections 4 and 5, respectively. Finally, Section 6 concludes the paper.
2. RLRR Design Issues

2.1. Accurate and Up-to-date Energy Information

In local load-balancing protocols, beaconing is required periodically for setting up energy information tables. During the interval between two beacons, the energy information stored in the table does not reflect the actual energy information, since sensor nodes likely consume energy continuously over time. Thus, the interval of broadcasting such a hello message provides a tradeoff between control overhead and timeliness of local energy information. In contrast, with the receiver-oriented approach in RLRR, a neighbor node uses its own energy information, which is always accurate and up-to-date, to evaluate its eligibility to be selected as a next hop node.

2.2. Load Balancing

We assume that every node starts with the same energy level corresponding to full battery capacity. In RLRR, the current energy levels (remaining battery capacities) of the sensor nodes are discretized into integer-valued quantized-energy-levels (QELs). An example is given in Fig. 1. Assuming the full energy level \( E_{\text{max}} \) of a battery is equal to 10000, and the “unit energy” (the unit of the quantization, \( E_{\text{unit}} \)) is equal to 2000. Then, the maximum value of QEL is \( QEL_{\text{max}} = \lceil \frac{E_{\text{max}}}{E_{\text{unit}}} \rceil = 5 \). In this paper, we do not differentiate the energy levels of sensor nodes with the same QEL. For example, both energy levels 6500 and 6750 have the same QEL of 4. With effective load balancing in a WSN, the sensor nodes close to one another (e.g., within one hop distance) will have similar QELs after an extended period of network operation, because neighbors with higher QELs will be selected to forward data until their QELs are decreased to levels no higher than those of other neighbors. The larger the range of QELs, i.e., the smaller the unit energy used in the quantization of the energy levels, the better the load balancing performance should be. In this case, a longer expected lifetime is likely to be achieved, but at the expense of a higher control overhead to carry out more frequent route oscillations. Thus, \( QEL_{\text{max}} \) should be optimized to achieve the best tradeoff between load balancing and control overhead.

2.3. Reliability

With the receiver-oriented approach, the property of broadcasting is exploited to attain high reliability. In RLRR, the source node and any intermediate sensor node broadcast a route selection message. Neighbors that receive the route selection message successfully have the responsibility of choosing the next hop among themselves. In the case that no such available neighbor is found, the node will mark itself a deadend node and inform the upstream node to discover a new route that bypasses the dead end. Especially, RLRR exploits neighbor nodes whose hop count is equal to that of the upstream node to increase reliability. However, RLRR faces the challenge of maintaining loop freedom when peer neighbors are exploited.

2.4. Loop Freedom

In order to guarantee loop freedom, many routing schemes based on neighbor information only adopt the set of “minimum hop count” nodes as backup next hop nodes to counteract frequent route failures. A “minimum hop count” node has a hop count to the sink that is 1 less than the hop count of the current node. Therefore these schemes exclude the neighbors whose hop counts are the same as that of the current node (i.e., peer neighbors) as potential next hop nodes. In RLRR, the number of backup nodes is exploited to a maximum extent possible by also involving peer neighbors to route data packets in order to achieve better load balancing and reliability. With the receiver-oriented approach, loop freedom is guaranteed with no additional control overhead, as will be explained in detail in Section 3.4.

2.5. Low-Cost Sensor Design

Traditional sensor routing protocols usually require a sensor node to maintain the information of multiple neighbors (e.g., backup routes and energy levels). In very large scale and dense WSNs, the amount of such information may pose an additional challenge for the sensor nodes with low storage capacity. However, with RLRR, sensor nodes do not need to store any additional routing and energy-related information except for the identifier of its next hop node and its upstream node for each flow. Though stateless geographical routing schemes also do not
need to set up route tables, they need to obtain geographical information using GPS devices. By comparison, RLRR does not need any geographical information to achieve stateless routing.

3. The RLRR Mechanism

3.1. The RLRR Protocol

In RLRR, each node has a “flow-entry” which indicates the identifier of its next hop node for forwarding data to the sink. Initially, a sink floods interest packets to the network. Each sensor sets up its hop count gradient to the sink. Sensor(s) matching the interest will become the source node(s) [5]. Unlike minimum hop count-based routing schemes, the flow-entry is not set up during interest flooding in RLRR, since load balancing cannot be attained simply by considering hop count. Instead, the flow-entries of all the sensor nodes are still empty after interest flooding.

We denote a forwarding node (the source or an intermediate node) by “h”. The arrival of a sensory data packet (from the application layer of the source node or from the upstream node) triggers h to check its flow-entry. Since the flow-entry does not exist initially, h stores the data, starts a “route selection” process immediately to set up the flow-entry, and then transmits the stored data to the selected next hop node. As illustrated in Fig. 2(a), suppose node i with QEL of 5 is selected as the next hop node of node h. After the flow-entry was set up, data packets will be unicast directly to the next hop node recorded in the flow-entry.

As time goes on, node i will consume its energy faster than its neighbors. To achieve load balancing, node i should keep track of its own QEL in order to prevent excessive energy consumption for packet forwarding. When its QEL is decreased by 1, node i asks its upstream node h to re-select a new next hop node. In the example shown in Fig. 2(b), when QEL of node i changes from 5 to 4, it unicasts a next-hop-reselection message (RESEL) to its upstream node h. Upon receiving RESEL, h deletes its current flow entry and initiates route reselection. Assuming node j is selected due to its higher energy level, node i will be replaced by node j as the new next hop node of h.

In addition to balancing the energy consumption, route reselection is also triggered to recover a link failure. In the example shown in Fig. 2(c), node h fails to deliver a data packet to node i according to the existing flow-entry, and receives feedback information from its MAC layer that indicates a transmission failure. Then node h deletes its current flow entry and initiates route reselection. Assuming the wireless link to node j is in a good condition and other factors (such as remaining energy and hop count) are favourable, hence this node is selected as the next hop and recorded in the flow-entry. The original next hop node i is now replaced by node j.

In addition, route selection/reselection (denoted by Sel/Resel, respectively) itself may fail. For example, if all the eligible neighbors (whose hop count to the sink is less than or equal to that of h) of node h have either depleted their energies or failed, node h becomes a dead end node. In this case, node h transmits a RESEL message to its upstream node (e.g., node g in Fig. 2(d)), which triggers a new route Reselection by node g, and so forth. In this paper, we employ the route selection/reselection mechanism proposed in [13].

3.2. Time Gradient Calculation

In RLRR, the energy and hop count information of each “live candidate (LC) node is converted to a TG that is used to evaluate the eligibility of the node as a next-hop node. The set of LCs of a forwarding node h includes all neighbors of h, which hop counts to the sink are less than or equal to that of h. To perform route Sel/Resel [13], h broadcasts a probe message (PROB) that is received by its LC’s. Each LC sets its “TG-Timer” to the calculated TG value and sends a “reply” message (REP) back to h when its “TG-Timer”
expires. The LC that originated the first REP received by \( h \) is selected as the next hop node. The best LC will have the least \( TG \); therefore, its TG-Timer will expire first among all the LCs and it will be selected as the next hop node.

In Fig. 3, the LCs of node \( h \) are divided into two groups: (1) less-hop-count group (L-Group), consisting of LCs which are 1-hop closer to the sink than node \( h \); and (2) equal-hop-count group (E-Group), consisting of the LCs having the same hop count as node \( h \). Obviously, the LCs in L-Group should have higher priority than those in E-Group. In Fig. 3, the L-Group includes nodes LC1, LC2, LC3, LC4, LC5; and the E-Group includes nodes LC6, LC7, LC8, LC9, LC10.

Recall that node \( h \) broadcasts a PROB to initiate route Sel/Resel. The PROB contains the QEL and hop count of node \( h \). In Fig. 3, \( QEL_{\text{max}} \) is equal to 10, and the QEL of each LC is indicated by the number in the respective circle. Upon receiving the PROB, an LC first decides which group it belongs. Then, it calculates the gap between its own QEL and the upstream node \( h \)'s QEL, which is denoted by \( \Delta E \). Since \( TG \) determines the delay of sending a REP back to \( h \), its value has a large impact on the data latency. In order to make \( TG \) as small as possible while achieving sufficient differentiation among all the LCs, we should avoid using large \( TG \) values to differentiate the LCs. Thus, we adopt \( \Delta E \) instead of QEL to differentiate the LCs in the same group, since \( \Delta E \) can be much smaller than QEL in a load balanced WSN.

Let \( TG_i \) denote the \( TG \) of node \( i \). \( TG_i \) is calculated by Eqn. (1), where \( x \) is a parameter reflects both \( \Delta E \)s and the type of group which an LC belongs to.

\[
\Delta E_i = \begin{cases} 
QEL_h - QEL_i, & \text{if } QEL_h = QEL_{\text{max}} \\
QEL_h - QEL_i + 1, & \text{if } QEL_h > QEL_i \text{ and } QEL_i \leq QEL_{\text{max}} \\
0, & \text{if } QEL_h \leq QEL_i
\end{cases}
\]

\[
x_i = \begin{cases} 
\Delta E_i, & \text{if } i \in \text{L-Group} \\
\Delta E_i + \alpha, & \text{if } i \in \text{E-Group}
\end{cases}
\]

\[
TG_i = f(x_i) = x_i \times \Delta TG + \text{rand}(\Delta TG)
\]

(1)

In Eqn. (1), \( \alpha \) is a positive constant used to differentiate between LCs in different groups by favoring the L-Group over the E-Group. \( \Delta TG \) is a constant, and rand(\( \Delta TG \)) is a random value between 0 and \( \Delta TG \).

\( \Delta TG \) should be set as small as possible to decrease the Sel/Resel delay, but if it is set too low, collisions of REP messages will occur frequently, because many LCs will likely try to send REPs within the small time period of \( \Delta TG \). Thus, \( \Delta TG \) should be set according to the node density. Let \( N \) be the total number of sensor nodes in a WSN that has an area \( A \). The node density of the WSN is equal to: \( \delta = \frac{N}{A} \). Let \( r \) be the transmission range of a sensor node. Roughly, LCs are located within approximately one third of the whole transmission range in Fig. 3. Then, the number of LCs can be approximated by:

\[
L = \frac{1}{3} \cdot \pi \cdot r^2 \cdot \delta
\]

(2)

Among the LCs in the same group, on the average, half of them will have the same QEL in a load balanced WSN. Let S-Group denote the set of the LCs with the same QEL in the same group (i.e. L-Group or E-Group). Our goal is to make a contention time long enough to differentiate the LCs in the same S-Group. Let \( T_{REP} \) be the average time to successfully deliver a REP message. In order to minimize collisions with other LCs in the same S-Group, at least \( T_{REP} \) should be reserved for each LC. Thus, \( \Delta TG \) is approximately equal to:

\[
\Delta TG = \frac{L}{2} \cdot T_{REP}
\]

(3)

Here \( \frac{L}{2} \) is the average number of LCs in an S-Group. Let \( \alpha \) be 2. In the example shown in Fig. 3, we can get four S-Groups: LC2, LC3, and LC5 with the \( x = 0 \); LC1 and LC4 with \( x = 1 \); LC6, LC7, and LC9 with \( x = 2 \); LC8 and LC10 with \( x = 3 \). The TGs of the LCs in each S-Group are randomly distributed over a range of \( \Delta TG \). In the example in Fig. 3, the increasing order of the TGs of all the LCs is: TG3, TG2, TG5, TG1, TG4, TG6, TG9, TG7, TG10, TG8. It corresponds to the decreasing order of LC’s eligibility level as \( h \)’s next hop node: LC3, LC2, LC5, LC1, LC4, LC6, LC9, LC7, LC10, LC8. As time goes on, the TG-Timer of LC3 will expire first, which causes LC3 to be selected as the next hop node.

3.3. Solving the Dead End Problem

The so-called dead end problem [9] arises when a packet is forwarded to a local optimum, i.e., a node with no neighbor of closer hop distance to the destination. The problem can be solved as follows: (1) If node \( h \) does not receive any REP until its NoREP-Timer expires, it will mark itself an unavailable node...
and unicast a RESEL to its upstream node $u$. An unavailable node will not participate in route Sel/Resel until the sink floods a new control message. The frequency of the sink flooding a control message should be traded off between control overhead and the timeliness of mitigating the deadend problem; (2) On receiving the RESEL, node $u$ initiates route reselection and finds a new next hop to replace node $h$.

### 3.4. Loop Freedom in RLRR

Exploiting multiple backup nodes or multipath for data delivery can increase reliability. In general, the nodes in equal-hop-count group ($E$-Group) are not used as backup nodes to guarantee loop freedom in conventional routing schemes in ad hoc and sensor networks. By comparison, in RLRR, the LCs in $E$-Group are exploited to achieve better performance in terms of both load balancing and reliability. While using the LCs in $E$-Group, it is critical to ensure that an LC in the $E$-Group not be selected as a next hop again by another LC in the same $E$-Group. The receiver-oriented approach of RLRR makes this goal easily achieved with no additional control overhead, as illustrated in Fig. 4.

In Fig. 4(a), assume that node $a$ is the only minimum-hop-count neighbor of node $h$, and it fails. Then, node $h$ initiates route Resel by broadcasting a PROB message and starting a Drop-PROB-Timer. The route Resel results in node $h$ selecting its peer neighbor node $b$ (in $E$-Group) as the next hop node. Assuming that the flow-entry of node $b$ does not exist, in Fig. 4(b), node $b$ initiates route selection. Note that both $h$ and $c$ are peer neighbors of node $b$. When node $b$ broadcasts a PROB and node $h$ replies first, node $h$ is selected as $b$’s next hop node and a loop is formed.

To prevent node $h$ from being selected as a next hop node by its peer neighbors, it ignores the PROBs and never participates in the selection until its Drop-PROB-Timer expires. When the Drop-PROB-Timer of node $h$ expires, it can participate in the next hop selection process again. In general, the time value of Drop-PROB-Timer ($T_{drop-prob-timer}$) should be long enough, such as:

$$T_{drop-prob-timer} = N_{E-group} \cdot t_{sel}. \quad (4)$$

In Eqn. (4), $N_{E-group}$ denotes the maximum number of LCs in $E$-Group. $t_{sel}$ denotes the time for one-hop route selection. Considering the worst case where all the LCs in the $E$-Group have no minimum-hop-count neighbors, each of them will initiate route selection once and finds a peer neighbor in the same $E$-Group as its next hop node. Then the accumulated
time for all of the route selection attempts is equal to Eqn. (4), which guarantees that a node (e.g., node $h$) initiating route $Sel/Resel$ will never be selected as a next hop node of any other $LC$'s in the same $E$-Group. Thus, loop freedom is guaranteed.

4. Simulation Methodology

4.1. Simulation Settings

In order to demonstrate the performance of RLRR, we compare it with several representative existing routing protocols for WSNs by extensive simulation studies.

We choose a global load-balancing scheme (i.e., EDDD [4]), a local load-balancing scheme (i.e. GEAR [1]), and a non-load-balancing scheme (i.e., DD [5]) to compare with RLRR. We implement the HGR protocol and perform simulations using OPNET Modeler [14, 15]. The sensor nodes are battery-operated except for the sink, which is assumed to have an infinite energy supply. The network with 800 nodes is uniformly deployed over a $500 \times 500$ field. As in [11], we let one sink stay at a corner of the field and one source node be located at the diagonal corner. Each source node generates sensed data packets at a constant bit rate with a 5 second interval between packets (1K Bytes each). As in [12], we use IEEE 802.11 DCF as the underlying MAC, and the radio transmission range ($R$) is set to 45m.

The data rate of the wireless channel is 2 Mbps. All messages are 128 bytes in length. We assume both the sink and sensor nodes are stationary. In DD, EDDD, and RLRR, the sink will initiate interest flooding to carry out a new task. Interest packets are propagated hop-by-hop throughout the network. Among the target sensor nodes, while several nodes may match the interest, only one of these nodes will become a source node for each instance of interest flooding. We assume that a mechanism exists to elect one source node among several nodes that matches the interest, e.g., based on the remaining energy. In addition to the initial interest flooding, the sink also floods the interest packet periodically to update stale information in terms of hop count and energy. Since RLRR does not rely on the periodical flooding for local repair, the sink only floods interest once until network lifetime is reached. We employ the energy model used in [11, 13] and link failure model used in [10]. For each set of results, we simulate the WSN sixty times with the specific set of parameters and different random seeds.

4.2. Performance Metrics

In this section, five performance metrics are defined:

- **Number of Successful Data Deliveries during Lifetime** - It is the number of data packets delivered to the sink before network lifetime is reached. It is denoted by $n_{data}$, which is also used as an indication of the lifetime in this paper.
- **Packet delivery ratio** - It is the ratio of the number of data packets delivered to the sink to the number of packets generated by the source nodes.
- **Average End-to-end Packet Delay** - It includes all possible delays during data dissemination, caused by queuing, retransmissions due to collisions at the MAC layer, and transmission time.
- **Energy Consumption per Successful Data Delivery** - It is denoted by $e$. It is the ratio of network energy consumption to the number of data packets delivered to the sink during the network lifetime. The network energy consumption includes all the energy consumption due to transmitting and receiving during the simulation. As in [10, 11, 12, 13], we do not account for energy consumption during the idle state, since this element is approximately the same for all the schemes considered.
- **Number of Control Messages per Successful Data Delivery** - It is denoted by $n_{ctrl}$ and is the ratio of the number of control messages transmitted to the number of data packets delivered to the sink during the network lifetime.

We use $n_{data}$ as an approximate indication of the network lifetime. If the packet delivery ratio is 100%, then $n_{data}$ is exactly proportional to the network lifetime, due to the CBR traffic model used in the simulations. We believe that $n_{data}$ is the most important metric for WSNs.

5. Performance Evaluation

5.1. Performance Evaluation of RLRR with Different Design Parameters

There are three main design parameters in RLRR, including $QEL_{max}$, $\alpha$ and $\Delta T G$. In the following, we examine the impact of these parameters on the RLRR performance.
be the total number of control messages used for route Sel/Resel. Then, $n_{ctrl}$ is equal to:

$$n_{ctrl} = \frac{N_f + N_s}{n_{data}} \tag{5}$$

If $QEL_{max}$ is very small (say, 2), $n_{data}$ is small, which makes $n_{ctrl}$ large in Fig. 6(b). When $QEL_{max}$ is up to 8, the quickly increased $n_{data}$ dominates Eqn. 5, and makes $n_{ctrl}$ decrease. Furthermore, when $QEL_{max}$ goes beyond 8, $n_{data}$ begins to decrease slightly (see Fig. 5(a)). However, $N_s$ increases in proportion to $QEL_{max}$, and dominates Eqn. 5. Thus, $n_{ctrl}$ increases again.

(b) Effects of $\alpha$: In these experiments, we change $\alpha$ from 0.5 to 4 by the step size of 0.25. Recall that $\alpha$ is an offset used to differentiate the $TG$s of $L$-$Group$ between the $L$-$Group$ and the $E$-$Group$. The lower is $\alpha$, the smaller is the difference in $TG$ values between the $L$-$Group$ and the $E$-$Group$. As an extreme example, if $\alpha = 0$, the $x$ values of the $LC$s in the $L$-$Group$ and the $E$-$Group$ will overlap (the case of $\alpha = 2$ is shown in Fig. 3). Thus, twice the number of $TG$-Timers may expire in the same $\Delta TG$ period, which is more likely to cause $REP$s to collide with each other. In Fig. 7(a), $n_{data}$ increases with $\alpha$, and reaches its maximum value when $\alpha = 1.5$. Then $n_{data}$ decreases when $\alpha$ is further increased, since the larger is $\alpha$, the less load balancing is achieved among all the $LC$s in Fig. 7(b).
It is unnecessary to increase $\Delta G$ minimum value when increasing $\Delta G$. In the opportunity to become a next-hop node. With $\Delta G$ change, it is likely will the REPs transmitted by LCs collide, which causes LCs in the E-Group may still have plenty of battery power.

Fig. 9: The impact of $\Delta G$ on: (a) $n_{\text{data}}$ and (b) end-to-end delay.

Fig. 10: The impact of $\Delta G$ on: (a) $e$ and (b) $n_{\text{ctrl}}$.

both the L-Group and the E-Group. As an extreme example, if $\alpha$ is close to $\infty$, the LCs in the E-Group will never be selected as the next hop node. When the LCs in the L-Group deplete their energy, the LCs in the E-Group may still have plenty of battery power. This phenomenon also demonstrates the positive effect of adopting E-Group for alternate routing. Fig. 7(b) shows a similar trend as Fig. 8.

The larger is $\alpha$, the less load balancing is achieved, the less is the frequency of route reselection, and hence the less control overhead is needed. Thus, $e$ decreases with $\alpha$ as shown in Fig. 8.

(c) Effects of $\Delta G$: In these experiments, we change $\Delta G$ from 2ms to 20ms by the step size of 2ms. In Fig. 9(a), $n_{\text{data}}$ increases as $\Delta G$ is increased, since the larger is $\Delta G$, the less collisions will happen, and the more data packets will be delivered successfully to the sink.

In Fig. 9(b), when $\Delta G$ is small, end-to-end data delay of RLRR is high. The smaller is $\Delta G$, the more likely will the REPs transmitted by LCs with similar TGs collide, which causes LCs with lower TGs not to win the opportunity to become a next-hop node. With increasing $\Delta G$, the delay decreases, and reaches its minimum value when $\Delta G$ is equal to 8 milliseconds. It is unnecessary to increase $\Delta G$ more if the value is large enough to differentiate the LCs in the same S-Group, since a large $\Delta G$ also increases the time for route Sel/Resel. Thus, when $\Delta G$ goes beyond 8ms, the delay begins to increase again.

In Fig. 10(a) and Fig. 10(b), both $e$ and $n_{\text{ctrl}}$ decreases with $\Delta G$ increasing. The larger is $\Delta G$, the less likely collision happens. Thus, the control overhead decreases.

5.2. Comparison of RLRR, EDDD, DD and GEAR with Variable Link Failure Rates

In this section, we change the link failure rate from 0 to 0.5 by the step size of 0.05. Fig. 11 shows that the packet delivery ratios of EDDD and DD are more sensitive to link failures than those of GEAR and RLRR, and EDDD has the lowest reliability because the load balanced path is not robust to link failures, since the failure of any link along the path will cause data delivery failure. RLRR yields higher reliability than GEAR because it exploits E-Group for alternating routing. In most cases, the numbers of nodes in the L-Group and E-Group are larger than the number of backup nodes in GEAR. Thus, RLRR keeps achieving more than 90% packet delivery ratio until the link failure rate is larger than 0.35.

In Fig. 12, when the link failure rate is 0, $n_{\text{data}}$ of EDDD is larger than that of RLRR and GEAR, which illustrates the advantage of global load-balancing (EDDD) over local load-balancing (RLRR, GEAR) in reliable environments. Note that $n_{\text{data}}$ is closely related to the network lifetime. Since DD has no load balancing mechanism, its lifetime is the lowest. With increasing link failure rate, RLRR exhibits consistently higher reliability and $n_{\text{data}}$ than the other...
schemes, which shows that the proposed receiver-oriented scheme can achieve load-balancing with a lower control overhead and handle link failure better than conventional transmitter-oriented schemes.

In Fig. 13, the end-to-end delays of all the schemes increase as link failure rate is increased. RLRR exhibits a lower end-to-end delay than EDDD and DD in case of link failure. It is because RLRR recovers from link/node failure faster than EDDD or DD. When the link failure rate is smaller than 0.35, RLRR has higher delays than GEAR because next hop selection delay is introduced during route Sel/Resel. If the link failure rate goes beyond 0.35, the delay of RLRR becomes the lowest.

Since EDDD and DD rely on message flooding to set up/update gradients, if all the backup nodes are broken, sensor nodes will wait for the sink node to initiate a new interest flooding to refresh the stale routes. In Fig. 14, the performace of RLRR and GEAR are better than those of EDDD and DD, since GEAR and RLRR never use control message flooding to repair a route. However, GEAR needs to exchange hello messages to update energy information. Thus, values of RLRR are less than those of GEAR. In addition, note that GEAR utilizes geographical information and requires that sensor nodes are equipped with GPS receivers. In contrast, RLRR does not need any geographical information. Furthermore, RLRR does not need to store the neighbor information in terms of energy and backup nodes as GEAR and EDDD do. Thus, RLRR can support low cost design for sensor nodes while achieving good performance as described above.

According to the simulation results, we observe the following: 1) lifetime is greatly prolonged if a load balancing mechanism is adopted (e.g., EDDD, RLRR, and GEAR vs. DD); 2) in a reliable environment, a global load-balancing scheme exhibits a longer lifetime than local load-balancing schemes (e.g. EDDD vs. RLRR and GEAR); 3) RLRR exhibits more consistent and relatively higher reliability and longer lifetime than EDDD, DD, and GEAR in unreliable environments.

6. Conclusion

We have proposed the novel RLRR protocol that employs a receiver-oriented approach to achieve both load balancing and reliability simultaneously in large scale WSNs. In unreliable communication environments, traditional routing protocols may fail to deliver data in a timely manner since global route

Fig. 12: Comparison of lifetime \( n_{\text{data}} \).

Fig. 13: Comparison of end-to-end delay.

Fig. 14: Comparison of energy consumption.
discovery may be needed to handle link failures. In RLRR, the upstream node of a broken link broadcast a route request message received by all the live neighbors with a good link. By taking this “local” approach, route repair is fast and reliability is enhanced even in highly unreliable environments. We have presented extensive simulation results to show that load balancing is achieved at the expense of energy for transmitting control messages. Thus, the related parameters should be selected carefully to achieve load balancing with energy-efficiency while minimizing the control overhead.

Acknowledgements

This work was supported in part by the Canadian Natural Sciences and Engineering Research Council under grant STGP 322208-05, the Basic Research Program of the Korea Science and Engineering Foundation under grant No. (R01-2004-000-10372-0), and the OPNET University Program.

References


Authors’ Biographies

Min Chen was born in Dec. 1980. He received the BS, MS and PhD degrees from the Dept. of Electronic Engineering, South China Univ. of Technology, in 1999, 2001 and 2004, respectively. Since 2006, he is a postdoctoral fellow in the Communications Group, Dept. of Electrical and Computer Engineering, Univ. of British Columbia. He was a post-doctoral researcher in School of Computer Science and Engineering, Seoul National Univ. for one and half years.

Victor C. M. Leung received the BASc (Hons.) and PhD degrees, both in electrical engineering, from the Univ. of British Columbia (UBC) in 1977 and 1981, respectively. He is a professor in the Dept. of Electrical and Computer Engineering, UBC. Dr. Leung is a Fellow of IEEE and a voting member of ACM. He is a member of the editorial boards of the IEEE Trans. on Wireless Communications, IEEE Trans. on Vehicularker Technology, International Journal of Sensor Networks, and International Journal of Communication Networks and Distributed Systems.

Shiwen Mao received the B.S. degree and the M.S. degree from Tsinghua Univ., Beijing, P.R. China in 1994 and 1997, respectively, both in Electrical Engineering, and the M.S. degree in System Engineering from Polytechnic Univ., Brooklyn, NY, in 2000. He received the Ph.D. degree in Electrical and Computer Engineering from Polytechnic Univ. in 2004. Currently, he is an Assistant Professor in the Dept. of Electrical and Computer Engineering at Auburn Univ., Auburn, AL. Before joining Auburn Univ., he had been a Research Scientist in the Bradley Dept. of Electrical and Computer Engineering at Virginia Tech. for over two years.

Taekyoung Kwon has been an assistant professor in the School of Computer Science and Engineering, Seoul National Univ. (SNU) since 2004. Before joining SNU, he was a post-doctoral research associate at UCLA and at City Univ. New York (CUNY). He obtained the B.S., M.S., and Ph.D. degrees from the Dept. of Computer Engineering, SNU, in 1993, 1995, 2000, respectively.