Asymmetry-Aware Link Quality Services in Wireless Sensor Networks

Junzhao Du[¶†], Weisong Shi[¶], and Kewei Sha[¶]

[¶]Department of Computer Science
Wayne State University, Detroit, MI 48202, USA
[†]Software College
Xidian University, Xi'an, Shaanxi 710071, P. R. China

Abstract. Recent studies in wireless sensor networks (WSN) have observed that the irregular link quality is a common phenomenon, rather than an anomaly. The irregular link quality, especially link asymmetry, has significant impacts on the design of WSN protocols. In this paper, we propose two asymmetry-aware link quality services: *the neighborhood link quality service (NLQS)* and *the link relay service (LRS)*. The novelty of the NLQS service is taking the link asymmetry into consideration to provide timeliness link quality and distinguishing the inbound and outbound neighbors with the support of LRS, which builds a relay framework to alleviate the effects of link asymmetry. To demonstrate the proposed link quality services, we design and implement two example applications, *the shortest hops routing tree (SHRT)* and *the best path reliability routing tree (BRRT)*, on the TinyOS platform. We found that the performance of two example applications is improved substantially. More than 40% of nodes identify more outbound neighbors and the percentage of increased outbound neighbors is between 14% and 100%. In SHRT, more than 15% of nodes reduce hops of the routing tree and the percentage of reduced hops is between 14% and 100%. In BRRT, more than 16% of nodes improve the path reliability of the routing tree and the percentage of the improved path reliability is between 2% to 50%.

1 Introduction

Recent empirical studies [1–6] on Berkeley motes platform show that there are highly irregular links in real deployments of wireless sensor networks (WSN). The packet delivery performance varies significantly with spatial and temporal factors and it is difficult to get the timeliness link quality information between neighbors. Furthermore, approximately 5% to 15% of all links are asymmetric links and asymmetric links vary significantly in different directions and distances. Asymmetric links, especially unidirectional links, bring the problem to the design of WSN protocols, such as MAC protocols, neighborhood and topology discovery protocols, and routing protocols.

Motivated by the recent studies on link irregularity, we intend to alleviate the problem caused by asymmetry on link quality services. On the other hand, we observed an interesting phenomenon that one or two steps relay is enough to forward packets across unidirectional links. This inspires us to design a relay service to utilize those otherwise useless asymmetric unidirectional links in traditional wisdom. To this end,

we have designed and implemented asymmetry-aware link quality services, including *the neighborhood link quality service (NLQS)* and *the link relay service (LRS)*, to provide the timeliness link quality information of neighbors and build a relay framework to alleviate effects of the link asymmetry on the TinyOS platform [7]. We leverage the Window Mean Exponentially Weighted Moving Average Estimator (WMEWMA) [3, 8] to estimate the packet reception rate (PRR) between neighbors. Our proposed link quality services are able to identify more outbound neighbors and provide timeliness link quality information of inbound and outbound neighbors. Both of these services are crucial for the design of link quality aware routing protocols. Our services also provide a relay framework to forward packets across unidirectional links, which have more positive contributions to localized coordination between neighbors. To demonstrate our proposed link quality services, we design and implement two example applications, building *the shortest hops routing tree (SHRT)* and *the best path reliability routing tree (BRRT)*. To evaluate our proposed link quality services, we have conducted both static analysis and simulation using the TOSSIM simulator [9]. We found that the performance of two example applications was improved substantially.

The rest of the paper is organized as follows. The design and implementation of the link quality services are presented in Section 2. SHRT and BRRT are described in Section 3. In Section 4, performance evaluation results are discussed. Related work and concluding remarks are listed in Section 5 and 6 respectively.

2 Asymmetry-Aware Link Quality Services

We consider the topology of a WSN as a weighted directed graph (G). All motes in the topology are vertexes of the graph and all vertexes belong to the vertex set, called $V(G)$. In this paper, we also call the vertex of a graph as node. All links in the topology are edges of the graph and all edges belong to the edge set, called $E(G)$. We use $e(u, v)$ to represent the edge from Node u to Node v, and assign the estimated PRR of the link as $weight$ of the edge. Figure 1(a) is an example topology. In this topology, every node has some inbound neighbors and outbound neighbors. In a WSN, a node can identify inbound neighbors only when it receives some packets from them. To identify outbound neighbors, the node should get acknowledgement from them. For example, Node 8 and Node 1 can receive packets from each other, then one of them will identify its counterpart as inbound and outbound neighbor. However, the link quality, in term of PRR, from Node 8 to Node 1 is different from that from Node 1 to Node 8, and the PRR will vary with time. Due to the asymmetric links, node can not identify all of its outbound neighbors. For example, Node 22 can identify Node 8 as its inbound neighbor. But Node 8 can not directly identify Node 22 as its outbound neighbor. Nevertheless, if Node 9 can relay the acknowledgement for Node 22, so that Node 8 will identify Node 22 as its outbound neighbors. In some situations, one step relay is not enough to identify all outbound neighbors. For example, if Node 24 will identify Node 9 as its outbound neighbor, both Node 23 and Node 52 should relay the acknowledgement from Node 9 to Node 24. The more steps the acknowledgement is relayed, the more the outbound neighbors will be identified. Next, we give a formal definition of *theoretical outbound neighbors*, *No Relay outbound neighbors*, *One-Step Relay outbound neighbors*, and *Two-Step Relay outbound neighbors*.

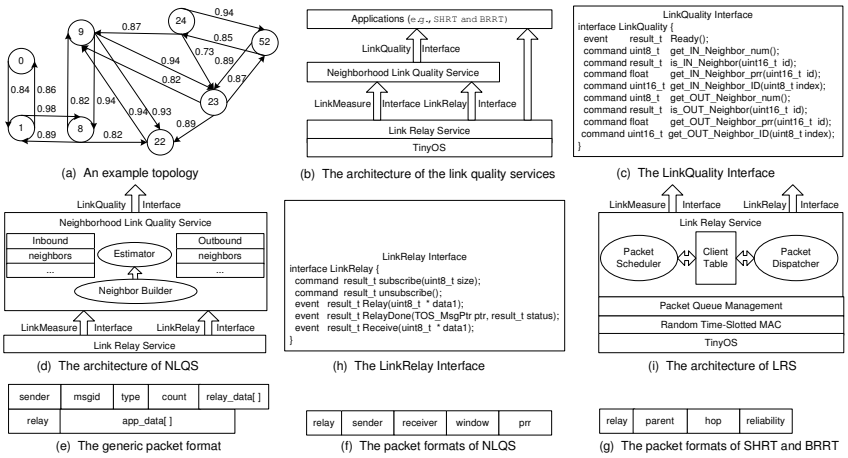


Fig. 1. The figures of the link quality services.

Definition 1 *Theoretical outbound neighbors of node v (N(v)):* $\{u|\forall u \in V(G) \wedge e(v,u) \in E(G)\}$

Definition 2 *No Relay outbound neighbors of node v (N0(v)):* $\{u|\forall u \in V(G) \wedge e(v,u) \in E(G) \wedge e(u,v) \in E(G)\}$

Definition 3 *One-Step Relay outbound neighbors of node v (N1(v)):* $\{u|(u \in N0(v)) \vee (\forall u \in V(G) \wedge e(v,u) \in E(G) \wedge \exists m \in V(G) \wedge e(u,m), e(m,v) \in E(G))\}$

Definition 4 *Two-Step Relay outbound neighbors of node v (N2(v)):* $\{u|(u \in N1(v)) \vee (\forall u \in V(G) \wedge e(v,u) \in E(G) \wedge \exists m,n \in V(G) \wedge e(u,m), e(m,n), e(n,v) \in E(G))\}$

Continuing with the above example, $N(24)$ includes Node 9, Node 23 and Node 52; $N0(24)$ only includes Node 52; $N1(24)$ includes Node 23 and Node 52; $N2(24)$ includes Node 9, Node 23, and Node 52. From these definitions and the above example, we know that we might find a relay path to relay packets across unidirectional links. If there is a path, there will be opportunity to relay packets across unidirectional links. If there is only one node in the middle of the path, we call this path *One-Relay path*. If there are two nodes in the middle of the path, we call this path *Two-Relay path* and so on. In next section, we will describe the link quality services, which can build inbound and outbound neighbors with timelessness link quality information and relay packets across unidirectional links.

2.1 System Overview

Next, we describe the design and implementation the link quality services, including NLQS and LRS, on the TinyOS platform. In NLQS, we measure and estimate the link quality using link quality estimator, build the inbound and outbound neighbors tables,

and provide the *LinkQuality* interface for applications to query the timeliness link quality information of neighbors. In LRS, we propose a link relay protocol, implement a link relay framework, and provide the *LinkRelay* interface for applications to relay packets across unidirectional links. Furthermore, NLQS uses the *LinkRelay* interface to relay the estimated PRR of the inbound neighbors back to them. Figure 1(b) shows the architecture of the link quality services. Due to space limitation, we provide the high level view only, interested readers please refer the technical report version of this paper [10].

2.2 Neighborhood Link Quality Service

NLQS provides the *LinkQuality* interface. Figure 1(c) lists the interface, which consists of one event and eight commands, complying with the TinyOS and nesC language [7]. When the service is ready, it will signal the $Ready$ event to notify the availability of the service. Even the service is ready for use, NLQS will continue measure and estimate the timeliness link quality between neighbors and update inbound and outbound neighbors tables. Applications can call those commands to query the link quality information. The link quality information includes that how many inbound and outbound neighbors are identified, what are their node ID, what are the estimated PRR from inbound neighbors and to outbound neighbors.

To provides this service, we need to measure and estimate the link quality between neighbors and build the inbound and outbound neighbors tables with the estimated PRR. To do this, every node will periodically broadcast some packets, which contain node ID and packet ID. Other nodes overhear the channel for the broadcast packets to measure and estimate the link quality from their inbound neighbors. Every node only measures and estimates PRR of the link from its inbound neighbors. We leverage WMEWMA to estimate PRR based on current and history measured results. WMEWMA uses a time window to observe the received packets and it adjusts the estimation result using latest average value of the measured PRR. When nodes have estimated PRR for their inbound neighbors, they will send the estimated results back to those inbound neighbors. When their inbound neighbors get the estimated results from their outbound neighbors, they can identify those outbound neighbors and build the outbound neighbors table with the estimated results. As we have discussed before, due to the link asymmetry, nodes may not directly receive packets from their outbound neighbors. They will get the estimated PRR to their outbound neighbors from LRS (Section 2.3).

Figure 1(d) shows the architecture of NLQS, which consists of four components: the inbound neighbor table, the outbound neighbor table, the neighbor builder, and the link quality estimator. The neighbor builder builds the inbound neighbor table using the PRR estimated by the link quality estimator and builds the outbound neighbor table using the PRR getting from LRS.

2.3 Link Relay Service

LRS is a general service to relay packets across unidirectional links and it is an additional layer between TinyOS and applications. Every node provided LRS will collaborate with each other to forward relay packets. In LRS, we propose a link relay protocol for inter-node communication, and design a relay framework and provide an interface for applications to relay packets.

Figure 1(e) illustrates a generic packet format used by LRS for inter-node communication. The *type* field indicates the type of the relay data. In LRS, every *type* of the relay data is associated with one application and LRS uses the value of the *type* field to dispatch the relay data to the associated application. The *relay_data* field contains the relay data, which will be forwarded to the application. Usually the first field of the relay data is the *relay* field and the value of this field indicates how many relay steps this relay data can be relayed. The value of the *relay* field will be decreased by one for every relay step until it reaches zero. If every node initializes the *relay* field as one, then the relay packet will be relayed once. We call this *One-Step Relay* algorithm. If every node initializes it as two, then the relay packet will be relayed twice. And we call this *Two-Step Relay* algorithm and so on. The more the packets are relayed, the more neighbors will receive the packets and the more opportunities exist to bypass unidirectional links. Applications will store its data in the *app_data* field. Figure 1(f) is the packet format used in NLQS to relay the estimated PRR to inbound neighbors (Section 2.2). Figure 1(g) is the packet format used in SHRT and BRRT (Section 3).

Figure 1(h) lists the *LinkRelay* interface. This interface is a parameterized interface, which can support 256 applications concurrently. As we have discussed that every application will be associated with an unique ID. This ID will be store in the $type$ field of the relay packet. This mechanism is similar to the active message on TinyOS platform. When an application will use LRS to relay packet, it first call the *subscribe* command. When a relay packet arrives this node, the associated application will get the *Receive* event from LRS. At this time, the application can make decision about how to process this incoming relay data. When the radio channel is available to send data, LRS will signal a *Relay* event to get the relay data from applications. At this time, the application can make decision about which relay data will be relayed and how many steps the relay data will be relayed. If the application is not interested in the LRS events any more, it can call the *unsubscribe* command to unsubscribe these events. In LRS, we follow the basic idea in system design to separate mechanism and policy . We design and implement the link relay mechanism and provide this interface for applications to make policy decisions.

Figure 1(i) shows the architecture of LRS. The packet dispatcher dispatches the relay data to applications. The packet scheduler schedules applications to forward relay data. The packet queue management manages packets received from the TinyOS platform and created by applications. The random time-slotted MAC is designed to avoid collisions of simultaneous packets transmissions between neighbors.

3 Example Applicatons

To demonstrate and evaluate our link quality services, we design and implement two example applications on the TinyOS platform: *the shortest hops routing tree* (SHRT) and *the best path reliability routing tree* (BRRT). In SHRT, every node has a routing path with shortest hops to the sink node. In BRRT, every node has a routing path with best reliability to the sink node. We define the path reliability for every node as follows: $Reliability_A = \prod_A^S PRR$, where the $Reliability_A$ is the path reliability for node A. The PRR is the packet reception rate of the links along the routing path from Node A

to the sink node S. For example, in Figure 1(a), Node 0 is the sink node and the path reliability of Node 9 is $0.72 = (0.94 * 0.89 * 0.86)$.

In SHRT and BRRT, every node initializes its hop count as the maximum value or its path reliability as zero, and its parent as empty. The sink node floods a packet, which contains its node ID and its hop count or its path reliability. The hop count of sink node is 0 and path reliability is 1.0. When receiving the flooding packet, node will update its parent and hop count or path reliability according to the received packet. In SHRT, if the hop count of the node in the flooding packet is smaller than that of the current parent, the receiver will choose the node as its parent and update its hop count based on the new parent. In BRRT, the receiver will choose the node as its parent, based on which the node can get better path reliability than based on current parent. After node updates its parent, it will broadcast its routing tree information again. This process continues until no node updates its parent.

However, due to the link asymmetry, nodes have more chances to build a broken routing tree when they choose the parent only considering the information embedded in the received packet. For example, in Figure 1(a), if Node 22 receives a flooding packet from Node 8, Node 22 will choose Node 8 as its parent. Unfortunately, Node 22 will build a broken routing tree. To avoid this, nodes should leverage NLQS to check whether the node in the flooding packet is one of their outbound neighbors before choosing it as its parent. The link asymmetry also has negative effects on building a better routing tree. For example, in Figure 1(a), Node 9 is the best parent candidate for Node 24. But due to the unidirectional link, Node 24 can not get flooding packet from Node 9. In this topology, Node 24 can get flooding packet from Node 52 and will choose Node 52 as its parent. In this case, Node 24 needs more hops to route packets to the sink node. However, if Node 9 is able to relay its flooding packets across the unidirectional link using LRS, Node 24 can get the flooding packet and will choose Node 9 as its parent. Our technical report [10] has the detail algorithms to build the SHRT and BRRT using NLQS and LRS.

4 Performance Evaluation

We are now in the position to evaluate the performance of our link quality services. We propose and define the following performance metrics: *the Number of Increased Outbound Neighbors, the Number of Reduced Hops in SHRT, the Path Reliability Improvement in BRRT,* and *Energy Consumption in SHRT and BRRT.* Based on these performance metrics, we conduct a static analysis to calculate the optimal results in an ideal network without packet loss. Then we simulate our link quality services using TOSSIM [9] in the lossy model. We totally simulate 156 motes. The bit error rate of links are randomly chosen from the real measurements using Berkeley MICA2 motes in a controlled environment.

As we know in Section 2 and Section 3 that the more steps the packets are relayed, the more outbound neighbors will be identified in NLQS and the better routing tree will be built in SHRT and BRRT. So we will simulate *No Relay*, *One-Step Relay*, and *Two-Step Relay* algorithms in our link quality services. We hope to compare the benefit and the overhead of the link relay mechanism and relay steps. In SHRT and BRRT, due to the lossy links and collisions, the flooding packets have a high probability to be lost.

To make sure most nodes can receive the flooding packets, we define three policies to forward it. First, the node only forwards its routing tree information twice *(P2 Policy)*. Second, the node will forward its routing tree information twice and forward the relay information once *(P2+R1 Policy)*. Third, the node will forward its routing tree information and the relay information twice *(P2+R2 Policy)*. In all, we will simulate *No Relay*, *One-Step Relay*, and *Two-Step Relay* algorithms with *P2 Policy*, *P2+R1 Policy*, and *P2+R2 Policy*.

Increased Outbound Neighbors Using our link quality services, node will identify more outbound neighbors. Figure 2 (a) shows the cumulative distribution function for the percentage of increased outbound neighbors using link quality services. In this figure, the x-axis stands for the percentage of increased outbound neighbors for every node and the y-axis stands for the percentage of these kinds of nodes. Results of the *One-Step Relay* and *Two-Step Relay* algorithms in the analysis and simulation are shown together to facilitate comparison. From this figure, we can see that more than 40% of nodes identify more outbound neighbors in the *One-Step Relay* algorithm and the percentage of increased outbound neighbors is from 17% to 100%. Also we can find that more than 45% of nodes identify more outbound neighbors in the *Two-Step Relay* algorithm and the percentage of increased outbound neighbors is from 14% to 100%. Note that the analysis results is a little better than those in the simulation. The reason for this is that some acknowledgement packets are lost due to loss links in the simulation. Therefore some nodes fail to identify those outbound neighbors. The *Two-Step Relay* algorithm is better than *One-Step Relay* as we expected.

Reduced Hops in SHRT With the help of our link quality services, we expect to build better SHRT, in which every node has shorter hops to the sink node than without using our services. Figure 2 (b) reports the CDF of the percentage of reduced hops for every node. In this figure, the x-axis is the percentage of reduced hops for every node and the y-axis shows the percentage of those kinds of nodes. From this figure, we can see that the results in the simulation do not match well with those in the analysis. In the analysis, there are more than 30% of nodes reduce their hops to the sink node and the percentage of reduced hops is mainly between 20% to 40%. In the simulation results, there are only more than 15% of nodes that reduce their hops to the sink node and the percentage of reduced hops is mainly between 15% and 25%. The *P2+R2 Policy* of *Two-Step Relay* is the best algorithm and more than 20% of nodes reduce their hops to the sink node. While the *P2+R1 Policy* of *One-Step Relay* is the worst one of all algorithms and only more than 13% of nodes reduce their hops to the sink node.

Improved Path Reliability in BRRT With the help of our link quality services, we expect to build better BRRT, in which every node has better path reliability to the sink node than without using our services. Figure 2 (c) reports the CDF of the percentage of the improved path reliability for every node in the simulated topology. Figure 2 (d) is the zooming of the results. In these figures, the x-axis stands for the percentage of the improved path reliability for every node and the y-axis stands for the percentage of those kinds of nodes. In the simulation, more than 15% of nodes improve the path reliability and the percentage of the improved path reliability is mainly between 2% and 40% and the percentage of some nodes even reach on 50%. The combination of the *P2+R2 Policy* with the *Two-Step Relay* algorithm is the best one, where more than

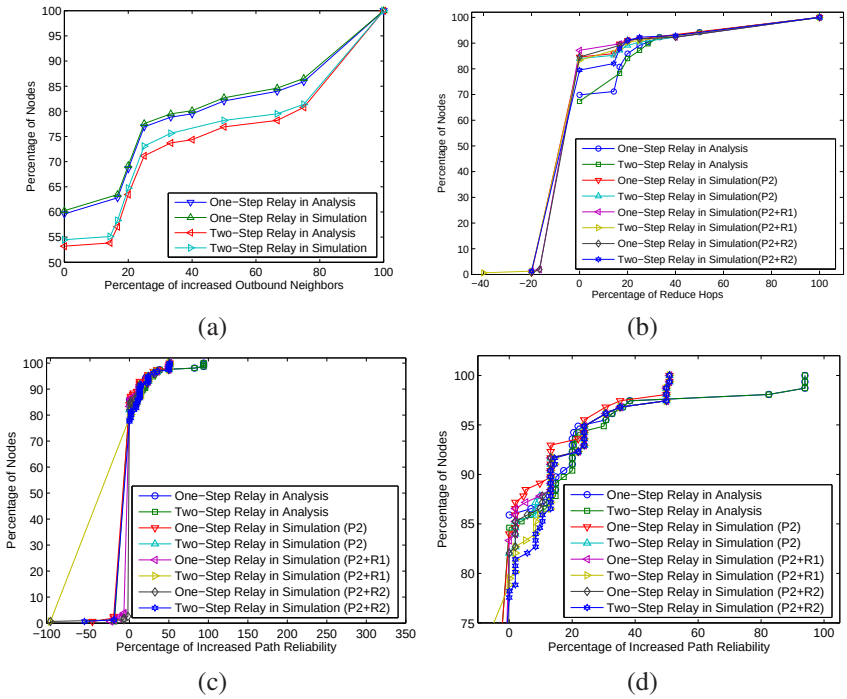
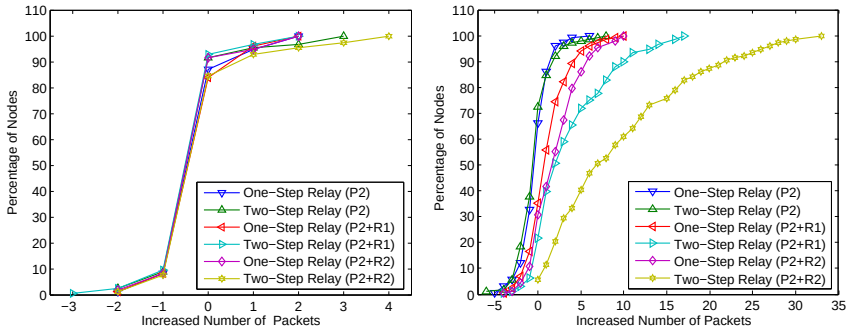


Fig. 2. Evaluation results: (a) The CDF of the percentage of increased outbound neighbors using link quality services, (b) The CDF of the percentage of reduced hops in SHRT, (c) The CDF of the improved path reliability in BRRT, and (d) Zooming of the CDF of the improved path reliability in BRRT.

27% of nodes improve the path reliability to the sink node and the percentage of the improved path reliability focus on 15% to 22% and 50%. The *P2 Policy* with the *One-Step Relay* algorithm is the worst one of all algorithms. Also we note that there are a lot of nodes improve the path reliability than the number in SHRT. Nearly every neighbor of a node in BRRT has different path reliability while most neighbors of a node in SHRT have same hops to the sink nodes. Therefore, nodes in BRRT have more opportunities to change their parents than they do in SHRT.

Energy Consumption in SHRT and BRRT Figure 3 (a) shows the CDF of more packets sent by the *One-Step Relay* and *Two-Step Relay* algorithms than *No Relay* in SHRT. In this figure, the x-axis stands for more packets send by every node and the y-axis reports the CDF. From this figure, we can see that all algorithms have little difference in terms of the number of extra packets sent. For the *One-Step Relay* and *Two-Step Relay* algorithms, about 90% of nodes do not send more packets and about 10% of nodes send less packets. Figure 3(b) shows the CDF of more packets received by the *One-Step Relay* and *Two-Step Relay* algorithms than *No Relay* one. In this figure, the x-axis stands for more packets received by every node and the y-axis reports the CDF. From this figure, we can see the following results. The *P2+R2 Policy* of *Two-Step Relay*



(a) (b)

Fig. 3. (a) The CDF of more packets sent in SHRT, (f) The CDF of more packets received in SHRT.

receives more packets than others. The *P2+R1 Policy* of *Two-Step Relay* is the second worst. The *P2 Policy* of *One-Step Relay* and the *P2 Policy* of *Two-Step Relay* follow the similar pattern and they receive less packets than other algorithms. Except the *P2+R2 Policy* of *Two-Step Relay*, most algorithms have nearly 20% of nodes which receive less packets than the *No Relay* algorithm. The CDF of more packets sent and received by the *One-Step Relay* and *Two-Step Relay* algorithms than *No Relay* one in BRRT have similar results with SHRT [10].

The results in this section show that there is a tradeoff between the energy consumption and the benefits brought by LRS. We argue that it is the application's decision to turn on/off LRS. For example, if the path reliability or path length is more important than the energy, e.g., event notification, then it should choose LRS. However, for monitoring-based applications it's better to turn off this service.

5 Related Work and Discussions

Our work is inspired by a variety of previous work, including *link layer characterization*, *link quality estimation*, and *existing solutions to link irregularity*.

Link layer characterization Ganesan *et al.* find long links, backward links, asymmetric links, stragglers and clustering in large scale of WSN [1]. Zhao *et al.* measure the spatial and temporal characteristics of packet delivery and find the gray area in which there are significant variabilities in packet delivery performance and the no-determining relationship between signal strength and packet delivery [4]. Zhou *et al.* show that radio irregularity is a common phenomenon, which arises from multiple factors, such as variance in RF sending power, and different path losses depending on the direction of radio signal propagation [5]. All of these research work shows us the reality of the link quality in WSN, which make us have a deeper understanding of the wireless communication in WSN. We are the first to provide a set of APIs for link quality services.

Link quality estimation Woo *et al.* show that WMEWMA is a very effective estimator, in term of stability and agility [8]. Cerpa *et al.* present a statistical model of lossy links in WSN [11]. Zhou *et al.* establish the Radio Irregularity Model (RIM) for simulation [5].

Zuniga *et al.* intend to identify the causes of the transitional region, and a quantification of their influence [6]. Different from these previous work, we focus on the asymmetry-aware link quality services that alleviate the effect of asymmetric links in the link quality measurement and estimation process, which is neglected in previous research.

Existing solutions to link irregularity Woo *et al.* study reliable and cost-based routing protocol techniques [3]. Seada *et al.* present energy-efficient forwarding strategies for geographic routing in lossy WSN [2]. Ramasubramanian *et al.* provide a bidirectional abstraction of the unidirectional network to routing protocols in *ad hoc networks* [12]. They results also show that reception based forwarding strategies are more efficient than purely distance-based strategies in the lossy link. We step further to distinguish inbound and outbound neighbors to make good use of them for high level routing service, and to provide a set of link quality interfaces. We also provide LRS for relaying packets across unidirectional links.

6 Conclusions

In this paper, we design and implement two asymmetry-aware link quality services. The performance of the services is evaluated in the context of two example applications, SHRT and BRRT. The detail simulation results using TOSSIM show that the performance of the applications can be improved substantially using proposed services.

References

1. Ganesan, D., Krishnamachari, B., Woo, A., Culler, D., Estrin, D., Wicker, S.: Complex behavior at scale: An experimental study of low-power wireless sensor networks. Technical Report UCLA/CSD-TR-02-0013, University of California, Los Angeles (2002)
2. Seada, K., Zuniga, M., Helmy, A., Krishnamachari, B.: Energy-efficient forwarding strategies for geographic routing in lossy wireless sensor networks. In: Proceedings of SenSys '04. (2004)
3. Woo, A., Tong, T., Culler, D.: Taming the underlying challenges of reliable multihop routing in sensor networks. In: Proceedings of SenSys '03. (2003)
4. Zhao, J., Govindan, R.: Understanding packet delivery performance in dense wireless sensor networks. In: Proceedings of SenSys '03. (2003)
5. Zhou, G., He, T., Krishnamurthy, S., Stankovic, J.A.: Impact of radio irregularity on wireless sensor networks. In: Proceedings of MobiSys '04. (2004)
6. Zuniga, M., Krishnamachari, B.: Analyzing the transitional region in low power wireless links. In: Proceedings of SECON '04. (2004)
7. Levis, P., et al.: The emergence of networking abstractions and techniques in tinyos. In: Proceedings of NSDI '04. (2004)
8. Woo, A., Culler, D.: Evaluation of efficient link reliability estimators for low-power wireless networks. Technical Report UCB/CSD-03-1270, University of California, Berkeley (2003)
9. Levis, P., Lee, N., Welsh, M., Culler, D.: Tossim: Accurate and scalable simulation of entire tinyos applications. In: Proceedings of SenSys '03. (2003)
10. Du, J., Shi, W., Sha, K.: Asymmetry-aware link quality services in wireless sensor networks. Technical Report MIST-TR-2005-006, Wayne State University (2005)
11. Cerpa, A., Wong, J., Kuang, L., Potkonjak, M., Estrin, D.: Statistical model of lossy links in wireless sensor networks. Technical Report CENS Technical Report, University of California, Los Angeles (2004)
12. V. Ramasubramanian, R.C., Mosse, D.: Providing a bidirectional abstraction for unidirectional ad hoc networks. In: Proceedings of INFOCOM '02. (2002)