

# A Survey of Transport Protocols for Wireless Sensor Networks

Chonggang Wang<sup>1</sup> Mahmoud Daneshmand<sup>2</sup> Bo Li<sup>3</sup> Kazem Sohraby<sup>1</sup>

<sup>1</sup>Department of Electrical Engineering, University of Arkansas, Fayetteville, AR 72701, USA

<sup>2</sup>AT&T Labs Research, Florham Park, NJ 07932, USA

<sup>3</sup>Department of Computer Science, The Hong Kong University of Science and Technology, Hong Kong, China

**Abstract**—In this article we present an overview for transport protocols for Wireless Sensor Networks (WSNs). We firstly highlight the unique aspects in WSNs, and describe the basic design criteria and challenges of transport protocols including energy-efficiency, quality of service, reliability, and congestion control. We then provide a summary and comparison of existing transport protocols for WSNs. Finally, we discuss several open problems.

**Index Terms**—transport protocols, congestion control, reliability, energy efficiency, wireless sensor networks

## I. INTRODUCTION

Wireless sensor networks (WSNs) are generally composed of one or more sinks (or base stations) and tens or thousands of sensor nodes scattered in a physical space. With integration of information sensing, computation, and wireless communication, the sensor nodes can sense physical information, process crude information, and report required information to the sink. The sink can query sensor nodes for such information. WSNs have several distinctive features: 1) *unique network topology*: sensor nodes are generally organized into a multi-hop star-tree topology, either flat or hierarchical. The sink at the root of the tree is responsible for data collection and relaying to exterior networks. This topology can be dynamic due to the time varying link condition and node variation; 2) *diverse applications*: WSNs can be used in different environment supporting diverse applications from habitat monitoring, target tracking, to security surveillance and so forth. These applications might be interested in different sensory data and therefore create different requirements in QoS (Quality of Service) and reliability; 3) *peculiar traffic characteristics*: In WSNs, the primary traffic is upstream flows from sensor nodes to the sink, although the sink may occasionally generate certain downstream traffic for the purpose of query and control. This upstream is a many-to-one communication. Dependent on specific applications, the delivery of upstream traffic can be event-driven, continuous delivery, query-driven delivery, and hybrid delivery; 4) *resource constraints*: Sensor nodes have limited resources including low computation capability, small memory, low wireless communication bandwidth, and limited and

non-rechargeable battery; 5) *small message size*: Messages in sensor networks usually have a small size compared to existing networks. As a result, there is usually no concept of segmenting a message in most applications under WSNs. These distinctive features pose considerably new challenges in the design of WSNs that can meet application requirements and operate for the longest possible time. Specifically, we need to carefully cope with such problems as energy conservation, reliability, and QoS.

Our focus in this article is on the design of transport protocols for WSNs. Transport protocols are used to eliminate or mitigate congestion and reduce packet loss, to provide fairness in bandwidth allocation, and to guarantee end-to-end reliability. However, the traditional transport protocols that are designed for the Internet, i.e., UDP and TCP, can not be directly applied to WSNs [1]. It is well documented that UDP itself does not provide any reliability often needed for many sensor applications, nor does it offer any flow and control congestion that can lead to packet loss and unnecessary energy consumption. On the other hand, TCP suffers several drawbacks: 1) The overhead associated with TCP connection establishment might not justify its usage for short data collections in most event-driven applications; 2) The flow and congestion control mechanism in TCP can discriminate sensor node(s) far away from the sink, and cause unfair bandwidth allocation and unfair data collections; 3) It is well-known that TCP has a degraded throughput under wireless systems especially with a high packet loss rate because TCP assumes all packet loss is due to congestion and triggers rate reduction whenever packet loss is detected; 4) In contrast to hop-by-hop control, end-to-end congestion control in TCP has a tardy response, which needs longer time to mitigate congestion and in turn leads to more packet loss when congestion occurs; 5) TCP still relies on end-to-end retransmission to provide reliable data transport, which basically consumes more energy and bandwidth than hop-by-hop retransmission; 6) TCP guarantees successful transmission of each packet, which is not proper for event-driven applications.

This article summarizes the existing transport protocols for WSNs and outlines open problems. The remainder of the paper is organized as follows. Section II introduces basic principles and design criteria in transport protocols for WSNs. Section III presents a summary and comparison of existing transport

protocols designed for WSNs. Section IV outlines several and future directions.

## II. THE GUIDELINES FOR THE DESIGN OF TRANSPORT PROTOCOLS IN A WSN

The transport protocol runs over the network layer protocol. It enables end-to-end message transmission, where messages are fragmented to chains of segments at senders and reassembled at receivers. The transport protocol usually provides the following functions: orderly transmission, flow control and congestion control, loss recovery, and possibly QoS guarantee such as timing requirement and fairness. In WSNs many new factors such as the convergent nature of upstream traffic and limited wireless bandwidth can cause congestion. The congestion influences normal data transmission and leads to packet loss. In addition, wireless channel introduces packet loss due to high bit-error rate, which not only affects reliability, but also wastes energy. As a result, two major problems that WSN transport protocols need to cope with are congestion and packet loss. We next discuss the performance metrics, the required functions of transport protocols for WSNs, and design options.

### A. Performance Metrics

Generally speaking, transport protocols for WSNs should provide end-to-end reliability and end-to-end QoS in an energy-efficient way. The performance of a transport protocol for WSNs can be evaluated using metrics including: energy-efficiency, reliability, QoS metrics such as packet loss ratio, packet delivery latency, and fairness.

**Energy-Efficiency:** Sensor nodes usually have limited energy. As a result, it is important for transport protocols to maintain high energy-efficiency in order to maximize system lifetime. Packet loss in WSNs can be common due to bit error and/or congestion. For loss-sensitive applications, the packet loss leads to retransmission and inevitably consumes additional energy. Therefore, there are several factors that have to be carefully evaluated including the number of retransmissions, the distance of each retransmission, and the overhead associated with control messages.

**Reliability:** different kinds of reliabilities can be needed in a WSN dependent on applications. The reliability can be classified into: 1) *packet reliability*: applications are loss-sensitive and require the successful transmission of all packets or a certain percent; 2) *event reliability* [2]: applications require only successful event detection, not the successful transmission of each packet. In addition, paper [3] defines destination-related reliabilities.

**QoS Metrics:** Traditional QoS metrics include bandwidth, latency or delay, and packet loss rate. Dependent on application requirements, these metrics or their variants could be still applicable to WSNs. For example, sensor nodes can be used to transmit continuous images for target tracking. These sensor nodes generate high-speed data stream and require more bandwidth than event-based applications. For a delay-sensitive

application, it can also require the timely delivery of sensory data.

**Fairness:** Sensor nodes are usually scattered in a geographical area. Due to the many-to-one convergent nature of upstream traffic, it is more difficult for sensor nodes far away from the sink to transmit the data. Transport protocols need to provide fair bandwidth allocation among all sensor nodes so that the sink can obtain a fair amount of sensory data from all sensor nodes.

### B. Congestion Control

There are mainly two causes for congestion in WSNs. The first is due to packet arrival rate exceeding packet service rate. This more likely occurs at sensor nodes close to the sink as they usually have more combined upstream relay traffic. The second is influenced by the link level performance such as contention, interference and bit error. This type of congestion occurs on the link.

Congestion in WSNs has a direct impact on energy-efficiency and application's QoS. First, the congestion can cause buffer overflow and furthermore lead to longer queuing time and more packet loss. Not only can the packet loss degrade reliability and application's QoS, but also wastes limited energy and lowers energy-efficiency. Second, the congestion can still degrade link utilization. Third, the link-level congestion usually results in transmission collisions if contention-based link protocols, for example CSMA (Carrier Sense Multiple Access), are used to share radio resources. The transmission collision in turn will increase packet service time and waste additional energy. Therefore congestion in WSNs must be efficiently controlled, either to avoid it or mitigate it. Usually there are three mechanisms that can deal with this problem: *congestion detection*, *congestion notification*, and *rate adjustment*.

**Congestion detection:** In TCP, the congestion is observed or inferred at end nodes based on either timeout or redundant acknowledgements. In WSNs, more proactive methods can be used. A common mechanism is to use the queue length [4][5], packet service time [6], or the ratio between packet service time and packet inter-arrival time at an intermediate node [7]. For WSNs using CSMA-like MAC protocols, channel loading also can be measured and used as an indication of congestion, and measurement is also a means for determining congestion as that in [5].

**Congestion notification:** After detecting congestion, transport protocols need to propagate congestion information from the congested node to upstream sensor nodes or even the source nodes who contribute the detected congestion. The congestion information could be a single binary bit (or called congestion notification (CN) bit in [2][4][5]) or more rich information such as allowable data rate in [6] or the congestion degree in [7].

The approaches to notify congestion usually can be categorized into *explicit congestion notification*, and *implicit congestion notification*. The *explicit congestion notification* uses special control messages to notify the involved sensor

nodes of congestion such as *suppression messages* used in [5]. On the other hand, the *implicit congestion notification piggybacks* congestion information in normal data packets. Through receiving or overhearing such packets, sensor nodes can obtain the piggybacked information. For example, in [2], sensor nodes can set a CN bit in the header of data packets. After receiving packets with CN bit set, the sink can know if congestion will happen in next reporting interval.

**Rate adjustment:** A sensor node upon receiving a congestion indication can adjust its sending rate. If a single CN bit is received, AIMD (Additive Increase Multiplicative Decrease) schemes or its variants are usually applied. On the other hand, if more congestion information is available, exact and accurate rate adjustment can be implemented such as in [6] and [7].

### C. Loss Recovery

In wireless environments, both congestion and link level bit error can cause packet loss, which deteriorate end-to-end reliability and QoS, and furthermore lower energy-efficiency. Other factors that can result in packet loss include node failure, wrong or outdated routing information, and energy depletion. In order to overcome this problem, we can increase source sending rate or introduce retransmission-based loss recovery. The first approach, for example used in ESRT (Event-to-Sink Reliable Transport) [2], works well for guaranteeing event reliability for event-driven applications that require no packet reliability; however, this method is not energy-efficient compared to loss recovery. The loss recovery is more active and energy-efficient, which can be implemented at both link layer and transport layer. The link layer loss recovery is hop-by-hop recovery while the transport layer loss recovery is usually done in an end-to-end way. Here we focus on loss recovery that consists of loss detection and notification and retransmission-based loss recovery.

**Loss detection and notification.** Since packet loss can be far more common in WSNs than that in a wired network, the loss detection has to be carefully designed. The common mechanism is to contain a *sequence number* in each packet header. The continuity of *sequence number* can be used to detect losses. Loss detection and notification can be either *end-to-end* or *hop-by-hop*. In the end-to-end approach such as that used in TCP protocol; the end-points (destination or source) are responsible for loss detection and notification. In the hop-by-hop method, intermediate nodes detect and notify packet loss.

The end-to-end approach is inappropriate for WSNs due to several reasons: 1) the control message (such as TCP ACK) used for end-to-end loss detection takes longer path and thus is not energy-efficient; 2) the control message travels through multiple hops, which might be lost with a high probability under WSNs due to either high link level error or congestion; 3) the end-to-end loss detection inevitably leads to end-to-end retransmission for loss recovery. However, the end-to-end retransmission will consume more energy than hop-by-hop retransmission.

In hop-by-hop loss detection and notification, a pair of

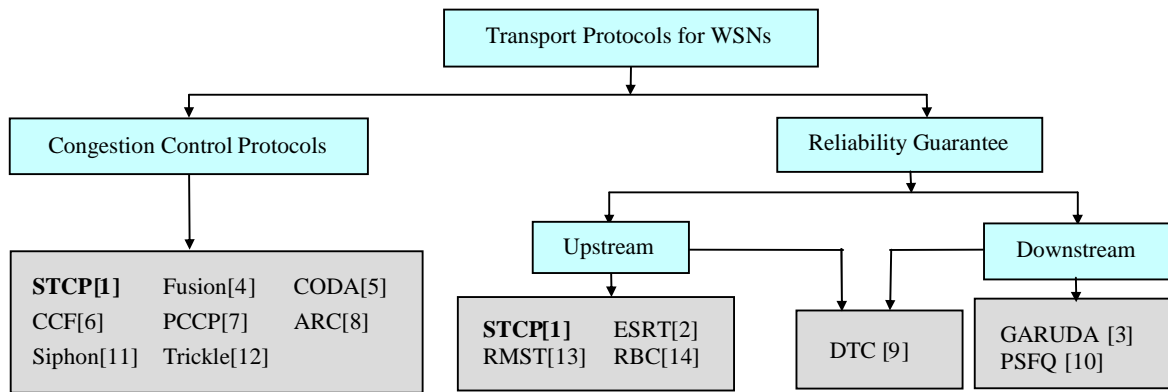
neighboring sender and receiver is responsible for loss detection, and in turn enables quick and energy-efficient retransmission within one hop. The hop-by-hop loss detection can be further categorized into *receiver-based* or *sender-based* according to the place where packet loss is detected. In the *sender-based* loss detection, the sender detects packet loss either timer-based or by overhearing. In the timer-based detection, a sender starts a timer each time it transmits a packet. If it does not receive any acknowledgment from the targeted receiver before the timer expires, it infers the packet lost. Taking the advantage of the broadcast nature of wireless channel, the sender can listen to the targeted receiver to see if the packet, which it sent to the receiver previously, has been successfully forwarded, and detect packet loss in a passively and indirect way.

In the *receiver-based* loss detection, a receiver infers packet loss when it observes out-of-sequence packet arrivals. There are three ways to notify the sender: ACK (Acknowledgement), NACK (Negative ACK), and IACK (Implicit ACK). Both ACK and NACK rely on special control messages, while Implicit ACK (IACK) [8] just piggybacks ACK information in the header of data packets. In IACK [8], if a packet is overheard being forwarded again, it implies that the packet has been successfully received and acknowledged simultaneously. The IACK avoids control message overhead and could be more energy-efficient. However, the application of IACK depends on if sensor nodes have the capability to correctly overhear the physical channel. In case that the transmission is corrupted before overhearing or the channel is not bidirectional or sensor nodes access the physical channel using TDMA-based (Time Division Multiple Access) protocols, IACK could be unreliable or impossible.

In addition, loss detection and notification can pinpoint the reason for packet loss, which can be further used to improve system performance. For example, if packet loss is caused by buffer overflow, source nodes need to reduce the sending rate. However, if they are resulted from channel error, it is unnecessary to reduce the sending rate in order to maintain high link utilization and throughput.

**Retransmission-based loss recovery.** The retransmission can be also either end-to-end or hop-by-hop. In end-to-end retransmission, the source performs retransmission. In hop-by-hop retransmission, an intermediate node that intercepts loss notification searches its local buffer. If it finds the lost packet, it will trigger retransmission. Otherwise it can relay loss information upstream till to the corresponding source node.

If we define the node with a cached packet as *cache point* and the node where packets are detected lost as *loss point*, the hop number between them can be referred to as *retransmission distance*. The retransmission distance reflects retransmission efficiency in terms of energy consumed by retransmission. In the end-to-end retransmission such as traditional TCP protocol, the cache point is always the source node. However, in the hop-by-hop retransmission, the cache point could be the



**Figure 1.** Existing transport protocols for WSNs

predecessor node of the loss point. Therefore the end-to-end retransmission usually has longer retransmission distance, and the hop-by-hop is more energy-efficient although the hop-by-hop requires intermediate nodes to cache certain packets. But hop-by-hop loss recovery can not assure message delivery in presence of node failure unless it is possible to take a quick local re-routing. In addition, The end-to-end approach allows for application-dependent variable reliability levels, like that realized by ESRT [2]. In contrast, the hop-by-hop recovery is better if 100% packet reliability is required although some applications in WSNs such as event-driven applications might not need 100% reliability from each sensor node.

Since end-to-end and hop-by-hop retransmission need to cache transmitted packets at cache points for potential future requests for retransmission, a question is how long a cache point would buffer a packet if the cache point receives no any acknowledgement. For end-to-end retransmission, the cache duration should be dependent on and close to round-trip-time (RTT). Under such wireless systems as using NACK-based acknowledgement, NACK messages could be lost or corrupted on the reverse path and the destination would send NACK more than once. In this case, source nodes need to buffer packet for certain time much longer than one RTT. For hop-by-hop loss detection and retransmission, the cache duration will be only influenced by the sum of local packet service time and one-hop packet transmission time.

We next discuss several issues related to hop-by-hop retransmission in WSNs. First, when to trigger retransmission? The retransmission can be triggered immediately upon the detection of a packet loss. This way results in shorter delay often desired by time-sensitive applications. However, if packet loss is caused by congestion, the immediate retransmission could aggravate the congestion and cause more packet loss. The second problem is related to the cache point. Where do the transmitted packets need to be cached? In the hop-by-hop retransmission, each packet could be cached at each and every intermediate node. Given the limited memory in sensor nodes, packets may only need to be cached at selected

nodes. The central issue is how to distribute cached packets among a set of nodes. The solutions such as DTC (Distributed TCP Cache) [9] balance buffer constraints and retransmission efficiency by using a probability-based selection of cache points. In order to optimize retransmission efficiency, another possible approach is to cache packets at the intermediate node that is closer to the potential congested node where packet loss more likely arises.

#### D. Design Guidelines

In order to design an efficient transport protocol for WSNs, several factors must be taken into consideration including the topology, diversity of applications, traffic characteristics, and resource constraints. The two most significant constraints introduced by WSNs are the energy constraints and fairness among different geographically placed sensor nodes. The transport protocol needs to provide high energy-efficiency and flexible reliability and sometimes the traditional QoS in terms of throughput, packet loss rate and end-to-end delay.

Therefore, transport protocols for WSNs should have components including congestion control and loss recovery, since the two components have direct impact on energy-efficiency, reliability, and application's QoS as explained above. There are generally two approaches to perform this task. First, design separated protocols or algorithms, respectively, for congestion control and loss recovery. Most existing protocols use this way and address either congestion control or reliable transport. With this separated and modular design, applications that need reliability can invoke only a loss recovery algorithm, or invoke a congestion control algorithm if they need to control congestion otherwise. They can so much as invoke both. For example, CODA (COngestion Detection and Avoidance) [5] is a congestion control protocol while PSFQ (Pump Slowly Fetch Quickly) [10] provides reliable transport. The joint use of them could provide full functions required by a transport protocol for WSNs. Second, design, if possible, a full-fledged transport protocol that provides congestion control and loss control in an integrated way. For example, STCP (Sensor Transmission

**Table 1.** Congestion Control Protocols for WSNs  
(CN: Congestion Notification; End-to-End: ETE; Hop-by-Hop: HBH)

Features Protocols	Congestion Detection	CN	Congestion Mitigation
STCP[1]	queue length	Implicit	AIMD-like ETE rate adjustment
Fusion[4]	queue length	Implicit	stop-and-start HBH rate adjustment
CODA[5]	queue length & channel status	Explicit	AIMD-like ETE rate adjustment
CCF[6]	packet service time	Implicit	exact HBH rate adjustment
PCCP[7]	packet inter-arrival time & packet service time	Implicit	exact HBH rate adjustment
ARC[8]	the event if the packets are successfully forwarded or not	Implicit	AIMD-like HBH rate adjustment
Siphon[11]	queue length & application fidelity	--	traffic redirection
Trickle[12]	--	--	polite gossip

Control Protocol) [1] implements both congestion control and flexible reliability in a single protocol. For different applications, STCP offers different control policies in a way to both guarantee application requirements and improve energy efficiency.

The first approach divides a problem into several sub-problems and is more flexible. The second approach can possibly optimize congestion control and loss recovery since loss recovery and congestion control in WSNs are often correlated. For example, congestion on contention-based wireless links can certainly lead to packet loss. The combination of CODA [5] and PSFQ [10] can possibly achieve both congestion control and reliability, yet it is not well documented in literatures that such congestion control protocols and reliability protocols can be seamlessly integrated together in an energy-efficient way. We believe there is a tradeoff between the architectural/modular design (the first approach) and integrated design with performance optimization (the second approach). The same tradeoff could also be observed between the traditional protocol stack and the cross-layer optimization in recent years.

### III. THE EXISTING TRANSPORT PROTOCOLS FOR WSNs

Several transport protocols have been designed for WSNs (Fig. 1). Some of which addressed congestion or reliability only, others examined both of them. We categorize them into three types: 1) congestion control protocols; 2) protocols for reliability; 3) protocols considering both congestion control and reliability. Due to the space constrains, it is hard to present detailed description of all the existing protocols. Please refer to corresponding references for more details.

#### A. Protocols for Congestion Control

Several congestion control protocols have been proposed for

upstream convergent traffic in WSNs. They differ in congestion detection, congestion notification, or rate-adjustment (Table 1).

Among them, Fusion [4] and CODA [5] detect congestion based on queue length at intermediate nodes, while CCF (Congestion Control and Fairness) [6] infers congestion based on packet service time. PCCP (Priority-based Congestion Control Protocol) [7] calculates congestion degree as the ratio of packet inter-arrival time and packet service time. In Siphon [11] congestion is still inferred, besides using the same method in CODA, based on the perceived application fidelity at the sink. CODA [5] uses explicit congestion notification, while others [4][6][7] take implicit congestion notification. In ARC (Adaptive Rate Control) [8], there is neither congestion detection nor notification. It controls congestions using a simple way, by which an intermediate node increases its sending rate by a constant  $\alpha$  if it overhears the successful packet forwarding by its parent node. Otherwise the intermediate node multiplies its sending rate by a factor  $\beta$ , where  $0 < \beta < 1$ . ARC maintains two independent sets of  $\alpha$  and  $\beta$ , respectively for source traffic and transit to guarantee fairness between them. In contrast, Fusion [4] controls congestion in a stop-and-start non-smooth way that makes neighboring nodes stop forwarding packets to the congested node immediately when congestion is detected and notified. CODA [5] adjusts sending rate in an AIMD way, while CCF [6] and PCCP [7] use an exact rate adjustment. Compared to CCF, PCCP provides priority-based fairness and overcomes the drawbacks caused by the use of non-work conservative scheduling in CCF [6]. However, there is no rate adjustment in Siphon [11]. When congestion occurs, Siphon redirects traffic to virtual sinks (VSs) that, beside the primary low-power mote radio, has another long-range radio used as a shortcut or “siphon” to mitigate congestion. Trickle [12] uses “Polite Gossip” to control traffic. In Trickle, each node tries to broadcast a summary of its data

**Table 2.** Reliable Transport Protocols for WSNs  
(LDN: Loss Detection and Notification; LR: Loss Recovery)

Protocols Features	End-to-End (ETE)		Hop-by-Hop (HBH)			
	STCP[1]	ESRT [2]	RMST[13]	RBC [14]	GARUDA [3]	PSFQ [10]
Direction	upstream	upstream	upstream	upstream	downstream	downstream
LDN	ACK&NACK	No	NACK	IACK	NACK	NACK
LR	ETE	No	HBH	HBH	two-tier two-stage loss recovery	HBH
Reliability	event & packet reliability	event reliability	packet reliability	packet reliability	packet reliability & destination-related reliability	packet reliability

periodically. In each period, each node can “politely” suppress its own broadcasting if the number of the same metadata, which this node receives from neighboring nodes, exceeds a threshold. On the other hand, if nodes receive new code or metadata, they can shorten the broadcast period and therefore leads to broadcast the new code earlier.

#### B. Protocols for Reliability

As shown in Table 2, some transport protocols [1] [2] [13] [14] examine upstream reliability, others [3] [10] investigate downstream reliability.

In upstream direction, ESRT [2] essentially discusses fidelity of the event stream and only guarantee *event reliability* through end-to-end source rate adjustment. In contrast, RMST (Reliable Multi-Segment Transport) [13] and Reliable Bursty Convergecast (RBC) [14] provides *packet reliability* through hop-by-hop retransmission-based loss recovery. The end-to-end source rate adjustment in ESRT follows two basic rules as: 1) if the current reliability perceived at the sink exceeds the desired value, ESRT will multiplicatively reduce the source rate of source nodes; 2) otherwise the source rate will be additively increased if the required reliability is not satisfied, unless there is congestion in network. RMST [13] jointly uses selective NACK and timer-driven mechanism for loss detection and notification, while RBC designs a windowless block acknowledgement and uses IACK. RBC still proposes intra-node packet scheduling and inter-node packet scheduling to avoid retransmission-based congestion.

In downstream direction, traffic is more like one-to-many multicast. The explicit loss detection and notification will meet the same problem of control message implosion as that in conventional reliable IP multicast. But the existing approaches for reliable IP multicast are inappropriate for WSNs [10]. Both GARUDA [3] and PSFQ [10] use NACK-based loss detection and notification and local retransmission for loss recovery, but they design different mechanisms to provide scalability. GARUDA [3] constructs a two-tier topology and proposes two-stage loss recovery. The two-tier topology consists of two layers, respectively, for core nodes and non-core nodes. The *hop-count* of each core node from the sink is a multiple of 3.

Then the first stage loss recovery is used to guarantees that all core nodes successfully get all transmitted packets from the sink, while the second stage is for non-core nodes to recover lost data from core nodes. GARUDA [3] further studies destination-related reliability. In contrast, PSFQ [10] contains three components: *pump operation*, *fetch operation*, and *report operation*. Firstly, the sink slowly and periodically broadcasts packets to its neighbors until all data fragments have been sent out. Second, a sensor node goes into fetch mode once a sequence number gap in a file fragment is detected. Then it sends a NACK in reverse path to recover the missing fragment. PSFQ does not propagate NACK messages in order to avoid message implosion. Specifically, the received NACK at an intermediate node will not be relayed unless the number of the same NACK that this node have received exceeds a predefined threshold and the lost segments requested by this NACK are simultaneously unavailable in this node. Third, the sink can make sensor nodes feedback information on data delivery status through a simple and scalable hop-by-hop report mechanism. PSFQ can be configurable to use all the bandwidth and in turn to overcome the delay caused by the slow pump.

#### C. Protocols for Congestion Control and Reliability

STCP [1] is a generic end-to-end upstream transport protocol for WSNs. It provides both congestion control and reliability and puts most work at the sink. In STCP, intermediate nodes detect congestion based on queue length and notify the sink by setting a bit in packet headers. It is a kind of network-assisted but end-to-end congestion control. One of the novelties in STCP is that it provides controlled variable reliability utilizing the diversity in applications. For example, STCP used NACK-based end-to-end retransmission for applications producing continuous flows, and ACK-based end-to-end retransmission for event-driven applications.

#### D. Open Problems

The above protocols have studied either congestion control or reliability guarantee, except STCP [1] that examines both problems. Some protocols use end-to-end control and others use hop-by-hop control. Some protocols guarantee event

reliability and others provide packet reliability. However, the existing protocols for WSNs have two primary limitations.

First, sensor nodes in a WSN might have different priorities since sensor nodes could be installed with different kinds of sensors and deployed in different geographical locations. Therefore sensor nodes can generate sensory data with different characteristics and have different priorities in reliability and bandwidth requirement. However, most existing protocols do not consider node priority, although our recent approach PCCP [7] provides a priority-based congestion control protocol. For example, most congestion control protocols guarantee only simple fairness, which means that the sink needs to get the same throughput from all nodes. In addition, most reliability protocols use a single and identical loss recovery algorithm for all nodes and applications, except STCP [1]. However, the nodes and the applications in a WSN could diversely have different features and priorities, which require flexible loss recovery in a way to optimize energy-efficiency.

Second, the existing transport protocols for WSNs basically assume that single-path routing is used in network layer and do not consider the scenarios with multi-path routing. It is not well documented if they can be directly applied to WSNs with multi-path routing. For example, regarding congestion control protocol when multi-path routing is used, a problem will be how a sensor node adjusts its own sending rate and the sending rate of its child nodes in a fair and scalable way since nodes have multiple parent and multiple paths to the sink at this time. This problem could be more complicated if some nodes in a WSN have multiple paths while others do not.

#### IV. CONCLUSIONS AND FUTURE DIRECTIONS

This article presents an overview of the transport protocol design in WSNs. We are planning to study their performance comparisons through simulation or experiments. The ideal transport protocol for WSNs should have high energy-efficiency, provide flexible reliability, and guarantee application-dependent QoS. Although some transport protocols have been proposed, there are several possible future works for performance optimization.

First, we are particularly interested in designing protocols to support node priority. The existing transport protocols, except STCP [1], only consider a single type of sensing devices. In reality, in a WSN, it is not uncommon that a node can be equipped with multiple types of sensors, e.g., temperatures and humidity. Thus nodes can have different priorities and can generate sensory data with different features and requirements in terms of loss and delay. Different mechanisms need to be designed in the transport protocol to deal with this diversity.

Second, the existing transport protocols consider only single-path routing. When multi-path routing is used in network layer, some issues such as fairness will be arisen and need to be examined further

Third, all the existing schemes either address congestion control or loss recovery; none of them except STCP [1],

investigates both problems systematically. In fact, a proper congestion control can reduce packet loss and provide better throughput. Loss recovery no doubt can enhance the reliability. Therefore a transport protocol should consider both together with considerations of performance optimization on the energy-efficiency and other performance metrics.

Finally, the existing transport protocols rarely consider the cross-layer interaction. In a WSN, the link level performance such as bit error rate can significantly impact the performance in a transport layer protocol; routing can certainly affect the hop-by-hop retransmission mechanism. Such cross-layer optimizations are therefore highly desirable.

#### REFERENCES

- [1] Y. G. Iyer, S. Gandham, and S. Venkatesan, "STCP: A generic transport layer protocol for wireless sensor networks," in *Proceedings of IEEE ICCCN 2005*, Oct. 17-19, San Diego, California, USA.
- [2] Y. Sankarasubramaniam, O. B. Akan, and I. F. Akyildiz, "ESRT: Event-to-sink reliable transport in wireless sensor networks," in *Proceedings of ACM MobiHoc '03*, Jun. 1-3, 2003, Annapolis, Maryland, USA.
- [3] S.-J. Park, R. Vedantham, R. Sivakumar, and I. F. Akyildiz, "A scalable approach for reliable downstream data delivery in wireless sensor networks," in *Proceedings of ACM MobiHoc'04*, May 24-26, 2004, Roppongi, Japan.
- [4] B. Hull, K. Jamieson, and H. Balakrishnan, "Mitigating congestion in wireless sensor networks," in *Proceedings of ACM Sensys'04*, Nov. 3-5, 2004, Baltimore, Maryland, USA.
- [5] C.-Y. Wan, S. B. Eisenman, and A. T. Campbell, "CODA: Congestion detection and avoidance in sensor networks," in *Proceedings of ACM Sensys'03*, Nov. 5-7, 2003, Los Angeles, California, USA.
- [6] C.-T. Ee and R. Bajcsy, "Congestion control and fairness for many-to-one routing in sensor networks," in *Proceedings of ACM Sensys'04*, Nov. 3-5, 2004, Baltimore, Maryland, USA.
- [7] C. Wang, K. Sohraby, V. Lawrence, and B. Li, "Priority-based congestion control in wireless sensor networks," Accepted to appear in *The IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC2006)*, June 5-7, 2006, Taichung, Taiwan.
- [8] A. Woo and D. C. Culler, "A transmission control scheme for media access in sensor networks," in *Proceedings of ACM Mobicom'01*, Jul. 16-21, 2001, Rome, Italy.
- [9] A. Dunkels, J. Alonso, T. Voigt, and H. Ritter, "Distributed TCP caching for wireless sensor networks," in *Proceedings of The Third Annual Mediterranean Ad Hoc Networking Workshop*, Jun. 27-30, 2004, Bodrum, Turkey.
- [10] C.-Y. Wan, A. T. Campbell, "PSFQ: A reliable transport protocol for wireless sensor networks," in *Proceedings of ACM WSN'02*, Sept. 28, 2002, Atlanta, Georgia, USA.
- [11] C.-Y. Wan, S. B. Eisenman, A. T. Campbell, and J. Crowcroft, "Siphon: Overload traffic management using multi-radio virtual sinks in sensor networks," in *Proceedings of ACM SenSys'05*, Nov. 2-4, 2005, San Diego, California, USA.
- [12] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks," in *Proceedings of First Symposium on Networked Systems Design and Implementation (NSDI'04)*, Mar. 29-31, San Francisco, California, USA.
- [13] F. Stann and J. Heidemann, "RMST: Reliable data transport in sensor networks," in *Proceedings of IEEE SNPA'03*, May 11, 2003, Anchorage, Alaska, USA.
- [14] H. Zhang, A. Arora, Y. Choi, and M. G. Gouda, "Reliable bursty convergecast in wireless sensor networks," in *Proceedings of ACM MobiHoc'05*, May 25-28, Urbana-Champaign, Illinois, USA.