RESEARCH ARTICLE OPEN ACCESS

A Study on Various Classifications of Transport Protocols for Wireless Sensor Networks

A.Ashok Kumar
Department of Computer Science, Alagappa Government Arts College, Karaikudi

ABSTRACT
In this paper I studied and compared the various types of transport protocols used in wireless sensor networks. A transport layer is needed in wireless sensor networks to control congestion and ensure reliable delivery of messages from the sensor nodes to the sink. In this paper I also considered the features and the design issues of protocols for WSNs. The classification of protocols presented in this paper based on the design issues. Finally, the protocols are compared with some of the features discussed in the protocols.
Keywords: - **Wireless** Sensor Networks, Transport Protocols, Congestion Control, Reliability, sensor.

I. INTRODUCTION

Wireless sensor networks (WSNs) are generally composed of one or more sinks 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 them to the sink. The sink in turn queries the sensor nodes for information. WSNs have several distinctive features[4,5]:

- **Unique network topology**: Sensor nodes are generally organized in a multihop star-tree topology that is either flat or hierarchical. The sink at the root of the tree is responsible for data collection and relaying to external networks. This topology can be dynamic due to the time-varying link condition and node variation.
- **Diverse applications**: WSNs may be used in different environments supporting diverse applications, from habitat monitoring and target tracking to security surveillance and so on. These applications may be focused on different sensory data and therefore impose different requirements in terms of quality of service (QoS) and reliability.
- **Traffic characteristics**: In WSNs, the primary traffic is in the upstream direction from the sensor nodes to the sink, although the sink may occasionally generate certain downstream traffic for the purposes of query and control. In the upstream this is a many-to-one type of communication. Depending on specific applications, the delivery of upstream traffic may be event-driven, continuous delivery, query-driven delivery, or hybrid delivery.
- **Resource constraints**: Sensor nodes have limited resources, including low computational capability, small memory, low wireless communication bandwidth, and a limited, usually non rechargeable battery.
- **Small message size**: Messages in sensor networks usually have a small size compared with the existing networks. As a result, there is usually no concept of segmentation in most applications in WSNs.

The transport layer is in charge of offering services such as end-to-end reliability for data transmission. In some cases, transport layer protocols also fulfil the requirements of congestion control mechanisms (e.g. congestion avoidance in TCP [16]). However, WSNs exhibit different requirements from those of traditional networks. Certain applications may require event detection, independently of the sensor nodes in the area that actually detect the event. In this case, the transport protocol has to guarantee event reliability rather than data reliability. On the other hand, in contrast to the traditional operation of TCP, congestion control may be better performed hop-by-hop, which conserves more energy and bandwidth. These facts have motivated the design of a family of new transport and congestion control protocols for WSNs.

The rest of the paper is organized as follows. Transport protocol design issues are discussed in Sec. 2. The major problems in WSN transport protocols are discussed in Sec. 3. The classification and the concepts of existing transport layer protocols that have been studied in the literature for WSNs are discussed in Sec. 4, followed by a comparison of the protocols in Sec. 5. Concluding remarks are given in Sec. 6.

II. TRANSPORT PROTOCOL DESIGN ISSUES

The design of transport protocols for WSNs should consider the following factors [3,4,5]:

- Perform congestion control and reliable delivery of data. Since most data are from the sensor nodes to the sink, congestion might occur around the sink. Reliable delivery in WSNs may have a different meaning than that in traditional networks; correct transmission of every packet is guaranteed. For certain sensor applications, WSNs only need to receive packets correctly from a fraction of sensors in that area, not from every sensor node in that area.

- Transport protocols for wireless sensor networks should simplify the initial connection establishment process or use a connectionless protocol to speed up the connection process, improve throughput, and lower transmission delay.
- Transport protocols for WSNs should avoid packet loss as much as possible since loss translates to energy waste. To avoid packet loss, the transport protocol should use an active congestion control (ACC) at the cost of slightly lower link utilization. ACC triggers congestion avoidance before congestion actually occurs. As an example of ACC, the sender (or intermediate nodes) may reduce its sending (or forwarding) rate when the buffer size of the downstream neighbors exceeds a certain threshold.
- The transport control protocols should guarantee fairness for different sensor nodes in order that each sensor nodes can achieve fair throughput. Otherwise the biased sensor nodes cannot report the events in their area and system may misunderstand there is no any event in the area.
- If possible, a transport protocol should be designed with cross-layer optimization in mind. For example, if a routing algorithm informs the transport protocol of route failure, the protocol will be able to deduce that packet loss is not from congestion but from route failure. In this case, the sender may maintain its current rate.

III. MAJOR PROBLEMS IN WSN TRANSPORT PROTOCOL

The transport protocol runs over the network layer. It enables end-to-end message transmission, where messages may be fragmented into several segments at the transmitter and reassembled at the receiver. This protocol provides the following functions: orderly transmission, flow and congestion control, loss recovery, and possibly qos guarantees such as timing and fairness. In wsns several new factors, such as the convergent nature of upstream traffic and limited wireless bandwidth, can result in congestion. Congestion impacts normal data exchange and may lead to packet loss. In addition, wireless channel introduces packet loss due to bit-error rate, which not only affects reliability, but also wastes energy. As a result [4], two major problems that WSN transport protocols need to cope with are congestion and packet loss.

3.1 Congestion Control

There are mainly two causes for congestion in wsns. The first is due to the packet-arrival rate exceeding the packet-service rate. This is more likely to occur at sensor nodes close to the sink, as they usually carry more combined upstream traffic. The second cause is link-level performance aspects such as contention, interference, and bit-error rate. This type of congestion occurs on the link. Congestion in wsns has a direct impact on energy efficiency and application qos. For example, congestion can cause buffer overflow that may lead to larger queuing delays and higher packet loss. Not only can packet loss degrade reliability and application qos, but it can also waste the limited node energy. Congestion can also degrade link utilization. Furthermore, link-level congestion results in transmission collisions if contention-based link protocols such as Carrier Sense Multiple Access (CSMA), are used to share radio resources. Transmission collision in turn increases packet-service time and wastes energy. Therefore, congestion in wsns must be efficiently controlled, either to avoid it or mitigate it. Typically, there are three mechanisms that can deal with this problem: congestion detection[4,5,6,7], congestion notification[2,4,5,6,7], and rate adjustment[2,5,6,7].

3.2 Loss Recovery

In wireless environments, both congestion and bit error can cause packet loss, which deteriorates end-to-end reliability and qos, and furthermore lowers energy efficiency. Other factors that result in packet loss include node failure, wrong or outdated routing information, and energy depletion. In order to overcome this problem, one can increase the source sending rate or introduce retransmission-based loss recovery. The first approach, which is also used in event-to-sink reliable transport (ESRT) [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 method is more active and energy efficient, and can be implemented at both the link and transport layers. Link-layer loss recovery is hop-by-hop, while the transport layer recovery is usually done end-to-end. Here we focus on loss recovery that consists of loss detection and notification and retransmission recovery.

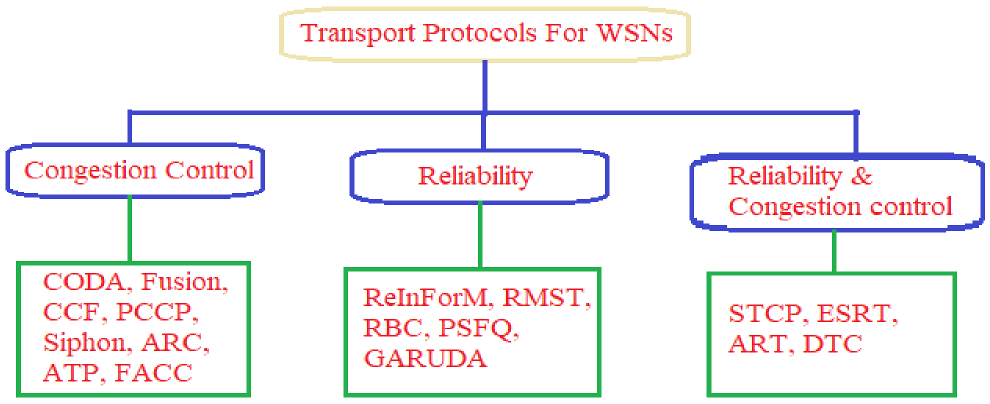
3.3 Reliability

In WSN, data is transferred in two directions. When mote detect an event, they send all the sensed information to the node at the sink. Sink then send the control packets to the sources. The transport protocols offer two directions of reliability; Upstream reliability is when data flow traffic is successfully delivered from source nodes to the sink; mostly it is unicast type of transmission, Downstream reliability is the delivery of control packets and queries successfully from sink to the source nodes, which is a multicast or the broadcast transfer. Reliability in wsns can be classified into the following categories: Packet reliability, Event reliability [4].

4. THE EXISTING TRANSPORT CONTROL PROTOCOLS FOR WSNs

Several transport protocols have been designed for WSNs (Figure 1), some of which have addressed congestion or reliability only, while others have examined both. We categorize them into three types:

- Congestion control protocols
- Protocols for reliability
- Protocols considering both congestion control and reliability


Transport Protocols For WSNs
Congestion Control
Reliability
Reliability & Congestion control
CODA, Fusion, CCF, PCCP, Siphon, ARC, ATP, FACC
ReInForM, RMST, RBC, PSFQ, GARUDA
STCP, ESRT, ART, DTC

4.1 Protocols for congestion control in WSNs

Congestion may occur in WSNs due to two main causes: i) the packet arrival rate at a node exceeding the node's service rate, and ii) PHY/MAC layer phenomena such as interference, errors and contention. This section presents various protocols designed for congestion control in WSNs. They basically differ in congestion detection, congestion notification and congestion mitigation mechanisms.

CODA (COngestion Detection and Avoidance) [19] is an upstream congestion control technique that consists of three elements: congestion detection, open-loop hop-by-hop backpressure, and closed-loop end-to-end multisource regulation. CODA attempts to detect congestion by monitoring current buffer occupancy and wireless channel load. If buffer occupancy or wireless channel load exceeds a threshold, it implies that congestion has occurred. The node that has detected congestion will then notify its upstream neighbor to reduce its rate, using an open-loop hop-by-hop backpressure. The upstream neighbor nodes trigger reduction of their output rate using methods such as AIMD. Finally, CODA regulates a multisource rate through a closed-loop end-to-end approach, as follows: (1) When a sensor node exceeds its theoretical rate, it sets a ‘‘regulation’’ bit in the ‘‘event’’ packet; (2) If the event packet received by the sink has a ‘‘regulation’’ bit set, the sink sends an ACK message to the sensor nodes and informs them to reduce their rate; and (3) if the congestion is cleared, the sink will send an immediate ACK control message to the sensor nodes, informing them that they can increase their rate. CODA's disadvantages are its unidirectional control, only from the sensors to the sink; there is no reliability consideration; and the response time of its closed-loop multisource control increases under heavy congestion since the ACK issued from the sink will probably be lost.

Fusion[10] method which is implemented by three techniques that span different layers of the traditional protocol stack: hop-by-hop flow control, rate limiting source traffic when transit traffic is present, and a prioritized medium access control (MAC) protocol. The first technique is hop-by-hop flow control, in which nodes signal local congestion to each other via backpressure, reducing packet loss rates and preventing the wasteful transmissions of packets that are only destined to be dropped at the downstream node. Hop-by-hop flow control has two components: congestion detection and congestion mitigation. A simple way to detect congestion relies on monitoring a sensor's queue size: if the fraction of space available in the output queue falls below a high water mark α , the congestion bit of outgoing packets is set; otherwise the congestion bit is cleared. Congestion mitigation is the mechanism by which nodes in a given radio neighborhood throttle their transmissions to prevent queues at their next-hop node from overflowing. When a sensor overhears a packet from its parent with the congestion bit set, it stops forwarding data, allowing the parent to drain its queues. Without such a feedback mechanism, packet buffers could easily be overrun when a wave of traffic flows through the network. If a path experiences persistent congestion, hop-by-hop backpressure will eventually reach the source, allowing application-level flow control to throttle the source rate. The second technique is a source rate limiting scheme to alleviate the serious unfairness toward sources that have to traverse a larger number of wire-less hops. The rate limiting scheme each sensor listens to the traffic its parent forwards to estimate N, the total number of unique sources routing through the parent. We then use a token bucket scheme to regulate each sensor's send rate. A sensor accumulates one token every time it hears its parent forward N packets, up to a maximum number of tokens. The sensor is allowed to send only when its token count is above zero, and each send costs one token. This approach rate-limit the sensor to send at the same rate of each of its descendants. The third technique is a prioritized MAC layer that gives a backlogged node priority over non-backlogged nodes for access to the shared medium, thus avoiding buffer drops. A carrier sense multiple access (CSMA) MAC can aid congestion control. It is imperative that congested sensors have prioritized access to the wireless medium. To address this issue, we adopt a technique that Aad and Castelluccia advocate [1], making the length of each sensor's randomized backoff (before every transmit cycle) a function of its local congestion state. If a sensor is congested, its backoff window is one-fourth the size of a non-congested sensor's backoff window, making it more likely that a congested sensor will win the contention period, allowing queues to drain and increasing the likelihood congestion control information will propagate throughout a sensor's neighborhood. This combination of these techniques into a strategy called Fusion. In isolation, each technique helps somewhat, but when acting in concert, Fusion dramatically improves network efficiency, fairness, and channel loss rates.

Congestion Control and Fairness (CCF)[8] is a distributed congestion control algorithm for tree based communications in wireless sensor networks, that seeks to adaptively assign a fair and efficient transmission rate to each node. Based on the difference of the two, a node then decides either to increase or decrease the bandwidth allocable to a flow originating from itself and to those being routed through it. Since the application requirements in sensor network follows no common trait, our design abstracts the notion of fairness, allowing for the development of a generic utility controlling module. Such separation of the utility and fairness controlling modules enables each one to use a separate control law, thereby portraying a more flexible design. The working of congestion control is independent of the underlying routing algorithm and is designed to adapt to changes in the underlying routing topology.

Priority-based Congestion Control Protocol (PCCP) [18] aims at controlling congestion in a WSN, while addressing the fact that nodes may have different priorities depending on their location and/or function. PCCP is composed of three main components: i) Intelligent Congestion Detection (ICD), ii) Implicit Congestion Notification (ICN), and iii) Priority-based Rate Adjustment (PRA).

ICD detects congestion according to packet inter-arrival and packet service times at the MAC layer of a node. Packet inter-arrival is defined as the time between two consecutive arriving packets, which can either be generated by a different node, required to be forwarded by this node, or alternatively can be packets created by this node. The packet service time is defined as the time difference between the instant at which the packet arrives at the MAC layer and the instant at which the last bit of the packet is transmitted. PCCP uses a parameter called congestion degree, which is the ratio of average packet service time over average packet inter-arrival time. If the congestion degree is greater than one, the node experiences congestion.

In order to propagate congestion information, PCCP uses ICN, which is based on piggybacking congestion information in the header of data packets. The transmission of these packets can be overheard by the nodes within the range of the corresponding sender. Hence, the children of a node can know when their parent is congested.

Finally, PRA requires the introduction of a scheduler with two sub-queues between the network layer and MAC layer. One sub-queue is for traffic generated in the node and the other is for transit traffic. On the basis of three priority index values, the scheduler gives the appropriate weight to each sub-queue and the scheduling rate is calculated in order to avoid or mitigate congestion. This calculation uses the congestion degree and priority index values for accurate rate adjustment.

Siphon [20] was also developed assuming a WSN used for data collection from sensor nodes to a sink node. In particular, it was designed as an alternative to other congestion control mechanisms, which are based on decreasing transmission rates and even dropping packets when congestion occurs. In effect, application data delivery ratio at the sink node may be compromised by these mechanisms in certain scenarios. Siphon comprises a set of algorithms that enable congestion to be detected and then mitigated by using virtual sink nodes, which are equipped with at least a secondary, long range, radio in addition to the primary one (e.g. cellular packet data services). The virtual sink nodes are defined as nodes with a different purpose than that of the physical sink node. The latter is the sink that typically exists in data collection WSNs. When congestion is detected in some area of the WSN, part of the traffic is transmitted to the virtual sink nodes, which forward the traffic to the physical sink node using the secondary radio. Hence, the virtual sink nodes fool the rest of nodes since they appear as new destination devices (i.e. additional sinks), but they actually by-pass traffic to the physical sink node using the secondary radio. Authors of Siphon justify that the benefits of the presence of some devices with two radios within a WSN can compensate their financial cost. In Siphon, congestion can be detected either by a sensor node or by the physical sink node.

Adaptive Random Clustering (ARC) [21], for large-scale WSNs with randomly deployed nodes. ARC, there is no congestion detection or notification; congestion control works as follows: an intermediate node increases its sending rate by a constant α if it overhears successful packet forwarding by its parent node. Otherwise, the intermediate node multiplies its sending rate by a factor β, where $0 < \beta < 1$. ARC maintains two independent sets of α and β, respectively, for source traffic and transit traffic in order to guarantee fairness. For the original ARC protocol: there are several novel features been exploited: First, instead of using location information, ARC forms a multi-hop cluster network with a required connectivity by using a novel cluster head competition scheme and proper transmit power settings. Second, required coverage is achieved by cluster heads and activated nodes, and thus less redundant nodes are activated than the existing algorithms which employ a coverage-first and connectivity-second activation procedure. Third, the lifetime of a WSN is prolonged through balancing energy consumption by updating cluster heads periodically, reducing redundancy of activated nodes by adaptively adjusting activation threshold, and reducing energy consumption by collision avoidance mechanism. Finally, ARC is suitable for practical applications of large-scale or high-density WSNs due to its distributed processing, scalable cluster topology, and easy management. It uses a very limited number of transmission channels to support a large number of clusters.

Aggregation and Transmission Protocol (ATP) [21] works based on a receiver-and network-assisted end-to-end feedback control algorithm. It uses selective ACKs (SACKs) for packet loss recovery. In ATP, intermediate network nodes compute the sum of exponentially distributed packet queuing and transmission delay, called D. The required end-to-end rate is set as the inverse of D. The values of D are computed over all packets that traverse a given sensor node, and if it exceeds the value that is piggybacked in each outgoing packet, it updates the field before forwarding the packet. The receiver calculates the required end-to-end rate (inverse of D) and feeds it back to the sender. Thus, the sender can intelligently adjust its sending rate according to the value received from the receiver. To guarantee reliability, ATP uses selective ACKs (SACKs) as an end-to-end mechanism for loss detection. ATP decouples congestion control from reliability and as a result, achieves better fairness and higher throughput than TCP. However, energy issues are not considered for this design, which raises the question of optimality of ATP for an end-to-end control scheme.

Fairness Aware Congestion Control (FACC) [22] is a congestion control mechanism, which controls the congestion and achieves fair bandwidth allocation for each flow of data. FACC detects the congestion based on packet drop rate at the sink node. In FACC nodes are divided in to two categories near sink node and near source node based on their location in WSNs. When a packet is lost, then the near sink nodes send a Warning Message (WM) to the near source node. After receiving WM the near source nodes send a Control Message (CM) to the source node. The source nodes adjust their sending rate based on the current traffic on the channel and the current sending rate. After receiving CM, flow rate would be adjusted based on newly calculated sending rate.

Trickle[12], an algorithm for propagating and maintaining code updates in wireless sensor networks. Trickle can scale to thousand-fold changes in network density, propagate new code in the order of seconds, and impose a maintenance cost on the order of a few sends an hour. Trickle uses "polite gossip" to exchange code metadata with nearby network neighbors. It breaks time into intervals, and at a random point in each interval, it considers broadcasting its code metadata. If Trickle has already heard several other motes gossip the same metadata in this interval, it politely stays quiet: repeating what someone else has said is rude. When a mote hears that a neighbor is behind the times (it hears older metadata), it brings everyone nearby up to date by broadcasting the needed pieces of code. When a mote hears that it is behind the times, it repeats the latest news it knows of (its own metadata); following the first rule, this triggers motes with newer code to broadcast it. More formally, each mote maintains a counter c, a threshold k, and a timer t in the range $[0,\tau]$. k is a small, fixed integer (e.g., 1 or 2) and τ is a time constant.

4.2 Protocols for reliability in WSNs

Reliability in WSNs can be classified into two categories: i) packet reliability, where applications require all packets to be successfully received, and ii) event reliability, where applications require event detection, but reception of all packets is not needed. This section presents various protocols of each category.

Reliable Information Forwarding (ReInForM) [6] is a protocol that offers stochastic packet reliability in WSNs, that is, packets are delivered with a certain probability. ReInForM uses neither Automatic Repeat reQuest (ARQ) mechanisms nor queues. ReInForM requires that nodes know some network parameters, such as the hop distance between themselves and the sink node, as well as the hop distance between their neighbours and the sink node. In addition, a node must know the channel error probability. In order to do this, the sink node periodically broadcasts a packet called routing update. When a node receives this packet, the node can learn from this packet what the hop distance is from the sink, since a field in this packet is updated accordingly every time it is forwarded by a node. Through this packet, the node also discovers who its neighbours are and their hop distance from the sink node.

When a node has a packet to transmit to the sink node, the first step is to assign a priority level. ReInForM defines n priority levels, each one of which corresponds to a certain delivery probability. According to the hop distance between the node and the sink node and the channel error probability, the node calculates the number of copies of the packet that are required. The copies are primarily sent to one of the neighbours of the node which are one hop closer to the sink node, should such neighbours exist. Otherwise, they are sent to one of the neighbours that are at the same distance from the sink node, should they exist. Finally, if all neighbours are one hop further from the sink node than this node, the copies of the packet are sent to one of these neighbours. In each of these three options, the selected next hop is chosen randomly, which allows the load for the network nodes to be balanced and node lifetime to be maximized.

Reliable Multi-Segment Transport (RMST) [19] belongs to upstream reliability guarantee. It is designed to run above Directed Diffusion (to use its discovered path from sensors to sink) in order to provide guaranteed reliability from sensors to sink (delivery and fragmentation/ reassembly) for applications. RMST is a selective NACK-based protocol. RMST basically operates as follows. Firstly, RMST uses timer-driver mechanism to detect data loss and send NACK on the way from detecting node to sources (Cache or non-Cache mode).Secondly, NACK receivers are responsible for looking for the missing packet, or forward NACK on the path toward sink if it fails to find the missing packet or in non-cache mode. RMTS is designed to run above directed diffusion [13], which is a routing protocol, in order to provide guaranteed reliability for applications. Problems with RMST are lack of congestion control, energy efficiency, and application-level reliability.

Reliable Bursty Converge-cast (RBC) [23] was designed for WSN applications where the detection of an event generates a large burst of packets which need to be transported reliably to a sink node and with low delay. The need for transmitting a large number of packets in a short time leads to channel contention, which is magnified by the fact that packets are transmitted several times via multi-hop routes until they reach the sink node. RBC uses a hop-by-hop, window-less block acknowledgment scheme that guarantees continuous packet forwarding independent from packet or ACK losses. Otherwise, window-based mechanisms may suffer transmission stalls that lead to throughput decrease.

RBC uses virtual queues at each node, which are managed without any window-based control and allow newly arrived packets to be sent immediately, instead of waiting for the previously sent packets to be acknowledged. A node R that forwards data packets received from node S, includes in each transmitted packet the maximal sequence of packets without any loss in the middle. Node S can then overhear node R’s transmissions and learn which packets have been correctly received. After sending a packet, the sender starts a retransmission timer. If the timer expires and the packet has not been acknowledged, it is retransmitted.

RBC has a mechanism for detecting and dropping duplicate packets. Duplicate packets may be generated when ACKs for correctly received packets are lost and the same packets are retransmitted. To enable this mechanism, each queue has a counter that is incremented every time a new packet is stored in the queue. Nodes maintain the last counter value piggybacked in the last packet from that queue. When channel contention occurs, retransmissions can further contribute to that contention. To ameliorate this, RBC accounts with a distributed contention control scheme that schedules packet retransmissions. Within a node, the retransmission of a packet experiences a delay that grows with the number of times the packet has already been retransmitted. Across nodes, those having more packets to transmit of similar freshness (which is piggybacked to the data packets it sends) are allowed to transmit earlier.

Pump Slowly Fetch Quickly (PSFQ) [8] aims to distribute data from sink to sensors by pacing data at a relatively slow-speed, but allowing nodes that experience data loss to fetch (recover) any missing segments from immediate neighbors very aggressively (local recovery, “fetch quickly”). It belongs to downstream reliability guarantee. The motivation of PSFQ isto achieves loose delay bounds while minimizing the loss recovery cost by localized recovery of data among immediate neighbors. It contains three components: Pump operation, Fetch operation, and Report operation. Firstly, sink slowly broadcasts a packet (with such fields-file ID, file length, sequence number, TTL, and report bit) to its neighbors every T until all the data fragments has been sent out. Secondly, a sensor can go into fetch mode once a sequence number gap in a file fragment is detected and issue NACK in reverse path to recover missing fragment. The NACK don’t need to be relayed unless the number of times the same NACK is heard exceeds a predefined threshold while the missing segments requested by the NACK message are no longer retained in a sensor’s cache. Thirdly, sink can make sensors to feedback data delivery status information to it through a simple and scalable hop-by-hop report mechanism. PSFQ has the following disadvantages: It cannot detect packet loss for single packet transmission; it uses a slow pump, which results in a large delay; and hop-by-hop recovery with cache necessitates larger buffer sizes.

GARUDA [18] is in the downstream reliability group. It is based on a two-tier node architecture; nodes with 3i hops from the sink are selected as core sensor nodes (i is an integer). The remaining nodes (noncore) are called second-tier nodes. Each noncore sensor node chooses a nearby core node as its core node. Noncore nodes use core nodes for lost packet recovery. GARUDA uses a NACK message for loss detection and notification. Loss recovery is performed in two categories: loss recovery among core sensor nodes [18], and loss recovery between noncore sensor nodes and their core node. Therefore, retransmission to recover lost packets looks like a hybrid scheme between pure hop by hop and end to end. GARUDA designs a repeated wait for first packet (WFP) pulse transmission to guarantee the success of single or first packet delivery. Furthermore, pulse transmission is used to compute the hop number and to select core sensor nodes in order to establish two-tier node architecture. Disadvantages of GARUDA include lack of reliability in the upstream direction and lack of congestion control. Published results on GARUDA at the time of this writing did not include reports of any results on reliability or a performance comparison with other algorithms, such as PSFQ.

4.3 Protocols for congestion control and reliability in WSNs

This section describes the most relevant protocols that have been designed for both congestion control and reliability in WSNs. Both protocols assume a WSN where data are collected by sensor nodes and are transmitted to the sink node, as well as coping with congestion control and reliability for this traffic pattern.

Sensor Transmission Control Protocol (STCP) [11] provides a generic, scalable and reliable transport layer paradigm for sensor networks. Majority of STCP functionalities are implemented at the base station. Each node might be the source of multiple data flows with different characteristics such as flow type, transmission rate and required reliability. STCP supports networks with multiple applications and provides additional functionalities such as controlled variable reliability and congestion detection and avoidance. Before transmitting packets, sensor nodes establish an association with the base station via a Session Initiation Packet. The session initiation packet informs the base station of the number of flows originating from the node, the type of data flow, transmission rate and required reliability. When the base station receives the session initiation packet, it stores all the information, sets the timers and other parameters for each flow, and acknowledges this packet. It is important for the sensor node to wait for the ACK to ensure that the association is established. The nodes can now start transmitting data packets to the base station. In the reverse path, the base station transmits an ACK or NACK depending on the type of flow. Congestion detection and avoidance is an important aspect in sensor networks. The random early detection (RED) mechanism designed by Floyd and Jacobson [9] proposes that an intermediate node drop a packet when it experiences congestion. The source is, therefore, effectively notified by a subsequent timeout or a NACK. Since dropping of packets is detrimental to sensor networks, we consider other solutions. In Ramakrishnan and Jain’s DECbit [15], intermediate nodes monitor the load experienced and explicitly notify the end nodes by setting a binary congestion bit in the packets. STCP adopts this method of explicit congestion notification with some modification. Each STCP data packet has a congestion notification bit in its header. Every sensor node maintains two thresholds in its buffer: t_{lower} and t_{higher}. When the buffer reaches t_{lower} , the congestion bit is set with a certain probability. The value of this probability can be determined by an approach similar to that employed in RED. When the buffer reaches t_{higher} , the node will set the congestion notification bit in every packet it forwards. On receiving this packet, the base station informs the source of the congested path by setting the congestion bit in the acknowledgement packet. On receiving the congestion notification, the source will either route successive packets along a different path or slow down the transmission rate.

Event-to-Sink Reliable Transport (ESRT) [1] which provides reliability and congestion control, belongs to the upstream reliability guarantee group. It periodically computes a reliability figure (r) , representing the rate of packets received successfully in a given time interval. ESRT then deduces the required sensor reporting frequency (f) from the reliability figure (r) using an expression such as f =G(r). Finally, ESRT informs all sensors of the values of (f) through an assumed channel with high power. ESRT uses an end-to-end approach to guarantee a desired reliability figure through adjusting the sensors’ reporting frequency. It provides overall reliability for the application. The additional benefit of ESRT is energy conservation through control of reporting frequency. Disadvantages of ESRT are that it advertises the same reporting frequency to all sensors (since different nodes may have contributed differently to congestion, applying different frequencies would be more appropriate) and considers mainly reliability and energy conservation as performance measures.

Asymmetric and reliable transport (ART) [17], a series of nodes which are called essential nodes, are selected. These nodes are selected in a way to be able to cover the whole area, and then a sub-network of these nodes is formed and merely is involved in reliable transmission and congestion control.

Distributed TCP Caching (DTC) [7] is a modified TCP for WSNs. The aim of DTC is to reduce the energy consumption of WSN nodes by decreasing the number of end-to-end retransmissions within the WSN, while offering the interoperability advantages of using TCP/IP. In fact, DTC is based on a hop-by-hop retransmission scheme. In DTC, the end devices operate by using regular TCP. However, intermediate nodes cache segments of end-to-end communications if they have available memory space. If node A is an intermediate node and has been able to cache a TCP segment, it forwards the segment to the next hop, starts a timer and waits for the reception of a link-layer acknowledgment sent by the corresponding next hop. If the timer expires and the acknowledgment has not been received, node A retransmits the segment. These mechanisms enable the number of times in which the retransmission time-out of the TCP sender expires to be reduced.

In effect, DTC adapts the main idea behind Snoop [2] for WSNs. Snoop was designed as a TCP proxy placed at the base station for wireless cellular networks. Snoop caches TCP segments transmitted in the downlink (i.e. to the mobile client) and maintain local timers so as to locally retransmit the lost data segments. In this way, losses in the wireless link can be hidden from the sender and its retransmission and congestion control mechanisms can be avoided. While in DTC the described mechanisms may be present in various intermediate nodes between sender and receiver, Snoop resides only in the base station, that is, a single element between sender and receiver.

IV. PERFORMANCE OF TRANSPORT CONTROL PROTOCOLS

In this section a quantitative comparison of WSN transport protocols for various classifications is presented. The attributes direction which is used for reliability guarantee, the supportiveness of the congestion mechanism, the congestion detection and notification support of the protocols, reliability support of the protocols used for WSNs shown in Table 1.

	Direction	Congestion support	Congestion detection	Reliability support
CODA	Upstream	Yes	Buffer	No
CCF	Downstream		Packet	-
PCCP	Upstream	Yes	Packet	-
ARC	Upstream	Yes	Packet	-
ATP	-	Yes	Packet	-
Siphon	-	-	Buffer	-
Fusion	-	-	Buffer	-
Trickle	-	Yes	-	-
ReInForM	-	-	Packet	Yes
RMST	Upstream	No	-	Yes
RBC	Upstream	-	-	Yes
PSFQ	Downstream	No	-	Yes
GARUDA	Downstream	No	-	Yes
STCP	Upstream	Yes	Buffer	Yes

ESRT	Upstream	Passive	Buffer	Yes
FACC	-	Yes	Packet	-
ART	Upstream & Downstream	Yes	Service Time	Yes
DTC	Downstream		Segment	Yes

The loss detection of the data transmitted in WSNs which is calculated for end-to-end and hop-by-hop cases and the energy efficiency of the protocols , as discussed earlier is shown in Table 2.

	Loss detection end-to-end o Hop-by-Hop	Congestion Notification	Energy Efficient
CODA	-	Explicit	Good
CCF	HbH	Implicit	
PCCP	HbH	Implicit	No
ARC	-	Implicit	-
ATP	E2E	-	-
Siphon	Traffic redirection	-	No
Fusion	HbH	Implicit	No
Trickle	-	-	No
ReInForM	HbH	-	-
RMST	HbH	-	Good
RBC	HbH	-	No
PSFQ	HbH	-	No
GARUDA	Hybrid(HbH and E2E)	-	Good
STCP	E2E	Implicit	No
ESRT	E2E	Implicit	Fair
FACC	HbH	Explicit	No
ART	E2E	Implicit	No
DTC	HbH	-	Fair

.
HbH: Hob-by-Hob; E2E: End-to_End.

6. CONCLUSION

In this paper I presented an overview of the transport control protocol for wireless sensor networks. The issues and problems in transport protocols for WSNs were discussed. A review of several existing wireless sensor transport control protocols was also provided based on the classifications of congestion control, congestion detection and reliability, and several problems in the existing protocols were described. When designing transport control protocols for wireless sensor networks, one should consider carefully such issues as: Protocol effectiveness and the efficiency of congestion control mechanisms, Reliability in the transport layer, Fairness among sensor nodes within different distances from the sink and Utilization of some type of cross-layer optimization to improve performance.

REFERENCES

1. Aad, I., and Castelluccia, C. Differentiation Mechanisms for IEEE 802.11. In Proc. of the IEEE INFOCOM Conf. (Anchorage, AK, April 2001), pp. 209–218
2. H. Balakrishnan et al. “Improving TCP/IP Performance Over Wireless Networks”, in proceedings of Mobicom’95, Berkeley, California, USA, November 1995.
3. Chonggang Wang, Kazem Sohraby, Bo Li, and Weiwen Tang, "Issues of Transport Control Protocols for WirelessSensor Networks", IEEE Proceedings International Conference on Communications, Circuits and Systems, 2005.
4. Chonggang Wang, Kazem Sohraby, Bo Li, Mahmoud Daneshmand,Yueming Hu,"A Survey of Transport Protocols for Wireless Sensor Networks", IEEE Network , May/June 2006
5. Chonggang Wang, Mahmoud Daneshmand , Bo Li, Kazem Sohraby, “ A Survey of Transport Protocols for Wireless Sensor Networks”, IEEE Network, 2006
6. B. Deb, S. Bhatnagar, B. Nath, “ReInForM: Reliable Information Forwarding Using Multiple Paths in Sensor Networks”, in proceedings of IEEE LCN, Bonn, Germany, October 2003.
7. A. Dunkels et al., “Distributed TCP Caching for Wireless Sensor Networks,” Proc. 3rd Annual Mediterranean Ad Hoc Net. Wksp., Bodrum, Turkey, June 27–30, 2004
8. C.-T. Ee and R. Bajcsy, “Congestion Control and Fairness for Many-to-One Routing in Sensor Networks,” Proc. ACM Sensys ’04, Baltimore, MD, Nov. 3–5, 2004.

9. S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking, August 1993
10. B. Hull, K. Jamieson, and H. Balakrishnan, “Mitigating Congestion in Wireless Sensor Networks,” Proc. ACM Sensys ’04, Baltimore, MD, Nov. 3–5, 2004
11. Y. G. Iyer, S. Gandham, and S. Venkatesan, “STCP: A Generic Transport Layer Protocol for Wireless Sensor Networks,” Proc. IEEE ICCCN 2005, San Diego, CA, Oct. 17–19, 2005.
12. P. Levis et al., “Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks,” Proc. 1st Symp. Networked Sys. Design and Implementation, San Francisco, CA, Mar. 29–31.
13. Nicholas S. Samaras; Foteini S. Triantari, "On Direct Diffusion Routing for Wireless Sensor Networks", Advances in Wireless and Optical Communications (RTUWO) proceedings, 3-4 Nov. 2016.
14. Raheleh Hashemzehi1 Reza Nourmandipour2 ,Farokh koroupi,” Congestion in Wireless Sensor Networks and Mechanisms for Controling Congestion”, Indian Journal of Computer Science and Engineering, Vol. 4 No.3 Jun-Jul 2013, 204-207.
15. K. Ramakrishnan and R. Jain. A Binary Feedback Scheme for Congestion Avoidance in Computer Networks. ACM Transactions on Computer Systems, May 1990.
16. W. Stevens, “TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms”, RFC 2001, January 1997.
17. Tezcan N and Wang W(2007). ART: an asymmetric and reliable transport mechanism for wireless sensor networks, International Journal of Sensor Networks, 2(3-4), 2007 ,188–200
18. C. Wang et al., “Priority-Based Congestion Control in Wireless Sensor Networks” (to appear), IEEE Int’l. Conf. Sensor Networks, Ubiquitous, and Trustworthy Comp., Taichung, Taiwan, June 5–7, 2006.
19. C.-Y. Wan, S. B. Eisenman, and A. T. Campbell, “CODA: Congestion Detection and Avoidance in Sensor Networks,” Proc. ACM Sensys ’03, Los Angeles, CA, Nov. 5–7, 2003
20. C.-Y. Wan et al., “Siphon: Overload Traffic Management Using Multi-Radio Virtual Sinks in Sensor Networks,” Proc. ACM SenSys ’05, San Diego, CA, Nov. 2–4, 2005.
21. A. Woo and D. C. Culler, “A Transmission Control Scheme for Media Access in Sensor Networks,” Proc. ACM Mobicom ’01, Rome, Italy, July 16–21, 2004.
22. Xiaoyan Y, Xingshe Z, Zhigang L Shining, L (2009). A Novel Congestion Control Scheme in Wireless Sensor Networks. In: 5th International Conference on Mobile Ad-hoc and Sensor Networks, Fujian, pp. 381–387 (2009)
23. H. Zhang et al., “Reliable Bursty Convergecast in Wireless Sensor Networks,” Proc. ACM Mobihoc ’05, Urbana-Champain, IL, May 25–28, 2005.