Journal of Network and Computer Applications 36 (2013) 134–146

Contents lists available at SciVerse ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Review

Wireless Sensor Network transport protocol: A critical review

A.J.Dinusha Rathnayaka [*], Vidyasagar M. Potdar

Digital Ecosystems and Business Intelligence Institute, Curtin University of Technology, Perth, Western Australia, Australia

ARTICLE INFO

Article history:
Received 29 March 2011
Received in revised form
18 August 2011
Accepted 2 October 2011
Available online 21 December 2011

Keywords:
Wireless Sensor Network
Transport layer protocol
Reliability
Congestion control
Prioritization
Experimental/simulation settings
Benchmark

ABSTRACT

The transport protocols for Wireless Sensor Network (WSN) play vital role in achieving the high performance together with longevity of the network. The researchers are continuously contributing in developing new transport layer protocols based on different principles and architectures enabling different combinations of technical features. The uniqueness of each new protocol more or less lies in these functional features, which can be commonly classified based on their proficiencies in fulfilling congestion control, reliability support, and prioritization. The performance of these protocols has been evaluated using dissimilar set of experimental/simulation parameters, thus there is no well defined benchmark for experimental/simulation settings. The researchers working in this area have to compare the performance of the new protocol with the existing protocols to prove that new protocol is better. However, one of the major challenges faced by the researchers is investigating the performance of all the existing protocols, which have been tested in different simulation environments. This leads the significance of having a well-defined benchmark for the experimental/simulation settings. If the future researchers simulate their protocols according to a standard set of simulation/experimental settings, the performance of those protocols can be directly compared with each other just using the published simulation results. This article offers a twofold contribution to support researchers working in the area of WSN transport protocol design. First, we extensively review the technical features of existing transport protocols and suggest a generic framework for a WSN transport protocol, which offers a strong groundwork for the new researchers to identify the open research issues. Second we analyse the experimental settings, focused application areas and the addressed performance criteria of existing protocols; thus suggest a benchmark of experimental/simulation settings for evaluating prospective transport protocols.

Contents

[*] Corresponding author. Tel.: +61430612341.
E-mail addresses: abekoon.rathnayaka@postgrad.curtin.edu.au (A.J.Dinush. Rathnayaka), v.potdar@curtin.edu.au (V.M. Potdar).

1084-8045/$ - see front matter © 2012 Published by Elsevier Ltd.
doi:10.1016/j.jnca.2011.10.001

1. Introduction

Wireless Sensor Network (WSN) (Culler et al., 2004) is comprised of tiny embedded devices termed as “motes” that has inbuilt features for sensing, processing, and communicating information over wireless channels. Transport protocol has gained fundamental importance in WSN as it establishes end-to-end connections over the network, while offering services such as congestion control, flow control, fair allocation of bandwidth, reliability, packet-loss recovery, energy efficiency, and heterogeneous application support. However proven transport protocols like User Datagram Protocol (UDP) (http://tools.ietf.org/html/rfc768, 2008) and Transmission Control Protocol (TCP) (www.ibiblio.org/pub/docs/rfc/rfc793.txt, 2008) are inappropriate for WSN due to many constraints in terms of throughput and energy efficiency. One of the major limitations in TCP is that it involves with end-to-end reliability model and enables expensive retransmission mechanism along every hop of the path between the source and the sink, if the packet is lost. TCP exhibits low throughput since it assumes that packet drop occurs due to congestion only and reduces the transmission rate. In contrast, UDP does not offer reliability, flow control, and congestion control, which are critical in WSN applications. The recent research community has attempted to overcome the limitations of standard protocols by developing novel transport protocols targeting WSNs. However different transport protocols use different technical parameters and mechanisms in achieving steady data communication in WSNs. Based on those parameters and mechanisms, these existing contributions can be classified into three categories: (i) protocols that support reliability only, (ii) protocols that support congestion control only, and (iii) protocols that support both (Table 1). Section 2 of this article extensively analyses the technical features and mechanisms used by different protocols and present a generic framework for WSN transport protocol based on those diverse features and mechanisms. In literature, we can find several research articles that compare the technical features and mechanisms of different transport protocols (Justin and Atiquzzaman, 2007). However, those works have not attempted to investigate all the protocols that we have analysed in this article, or to present a generic framework for WSN transport protocol.

Generally the researchers verify the concepts and confirm the performance advancements offered by their models through simulations and/or experiments. However different researchers have used different experimental/simulation environments having different settings such as packet size, number of exploited sensors and their distribution in the field, buffer size, coverage area, simulation duration and initial traffic load. As a result, the new researcher cannot directly compare the performance simulation results of the new protocol with the published performance simulation results of existing ones. In addition, it is impractical to run all the existing protocols in the same simulation environment, which is used to simulate the new protocol, due to the difficulty in finding simulation source codes for all existing works and time constraints. In literature, we can find that certain researchers have compared their new protocol with TCP and its variants and sometimes compared with few other recent protocols, but not with all existing protocols (Alam and Hong, 2009; Wang et al., 2006a, b). Considering all these issues, we can say that one of the fundamental challenges faced by the researchers

Table 1
Existing transport protocols classification.

Both congestion control and reliability support	
ART (Tezcan and Wang, 2007)	Asymmetric and reliable transport
CRRT (Alam and Hong, 2009)	Congestion aware and rate controlled reliable transport
CTCP (Giancoli et al., 2008)	Collaborative transport control protocol
DST (Gungor and Akan, 2006)	Delay sensitive transport
ESRT (Sankarasubramaniam et al., 2003)	Event-to-sink reliable transport
Flush (Kim et al., 2007)	Flush
PORT (Zhou and Lyu, 2005)	Price-oriented reliable transport
RCRT (Paek and Govindan, 2007)	Rate-controlled reliable transport
$(RT)^2$	Real-time and reliable transport
STCP (Zhang et al., 2005)	Sensor transmission control protocol
TRCCIT (Shaikh et al., 2010)	Tunable reliability with congestion control for information transport
Reliability support only	
DTC (Dunkels et al., 2004)	Distributed TCP caching
DTSN (Marchi et al., 2007)	Distributed transport for sensor networks
ERTP (Le et al., 2009)	Energy-efficient and reliable transport protocol
GARUDA (Park et al., 2004)	GARUDA
PSFQ (Wan et al., 2003)	Pump slowly fetch quickly
RBC (Gouda, 2005)	Reliable bursty convergecast
RMST (Stann and Heideman, 2003)	Reliable multi-segment transport
Congestion control only	
ARC (Woo et al., 2001)	Adaptive rate control
CCF (Ee and Bajcsy, 2004)	Congestion control and fairness
CODA (Wan et al., 2003)	Congestion detection and avoidance
Fusion (Hull et al., 2004)	Fusion
PCCP (Wang et al., 2006)	Priority-based congestion control protocol
PHTCCP (Monowar et al., 2008)	Prioritized heterogeneous traffic-oriented congestion control protocol
Siphon (Wan et al., 2005)	Siphon
Trickle (Levis and Patel (2004))	Trickle

working in this area is the effective comparison of all these existing protocols. However, if there is a well defined benchmark for experimental/simulation settings, all the researchers tend to follow the same benchmark in testing their protocol, thus the new researcher can easily evaluate the performance of his new protocol just by comparing with the published simulation/experimental results of previous works. This leads the fundamental requirement of presenting a benchmark for experimental settings. In literature, to date, there are no published works that analyse the experimental/simulation settings used by different protocols or present a benchmark for simulation/experiment settings. Section 3 of this article addresses this issue by analysing the simulation/experimental settings, application areas and addressed performance criteria of existing protocols, thus suggesting a benchmark for simulation/experimental settings for evaluating future protocols.

2. Generic structure of transport protocol

In this section we develop a framework for generic transport protocol (Fig. 1), amalgamating the technical features and functionalities of existing research work. Usual process in WSN data transfer is that the source mote synchronously senses the channel/environment and transfers the sensed information to the next hop. The intermediate nodes cache the information for post processing and transfer the scheduled queued data to the neighbouring nodes until it reaches the final destination or the sink. Here we divide the generic structure into three main functional modules: (i) congestion module, (ii) reliability module, and (iii) priority module, and evaluate the technical functionalities of existing transport protocols based on their proficiencies in these modules. As illustrated in Table 1 not all existing protocols present all three modules in their operation, but offer a combination of different capabilities. The uniqueness of each protocol more or less lies in one of the components or attributes discussed in generic structure. In next three sub sections we discuss these modules in detail, while referring relevant existing research contributions.

2.1. Congestion module

Congestion occurs when nodes transmit more combined upstream traffic resulting in packet-arrival rate to exceed the packet processing rate at the node. Congestion also arises when mote's data throughput exceeds the link's available data threshold limit and can also result due to wireless link issues such as contention, interference, and blind mote problem. Congestion causes packet drops and unnecessary packet retransmissions followed by significant network's energy depletion. The congestion module is activated to take corrective actions to reduce the congestion, hence to offer the desired reliability. As illustrated in Tables 1 and 2, TRCCIT, CRRT, CTCP, RT^2, ART, RCRT, Flush, DST, PORT, STCP, ESRT, PHTCCP, PCCP, Siphon, Fusion, CCF, Trickle, CODA, and ARC claim to achieve congestion control. Generally the Congestion module is composed of three sub-modules: (i) congestion detection, (ii) congestion notification, and (iii) congestion avoidance.

2.1.1. Congestion detection

Congestion detection refers to identification of possible events, which may build-up congestion in the network. Generally different protocols identify congestion by utilizing different combinations of the following parameters.

- *Buffer occupancy*: Buffer occupancy refers to the occupied buffer memory locations with respect to the maximum available memory. When the memory of the sensors reaches the

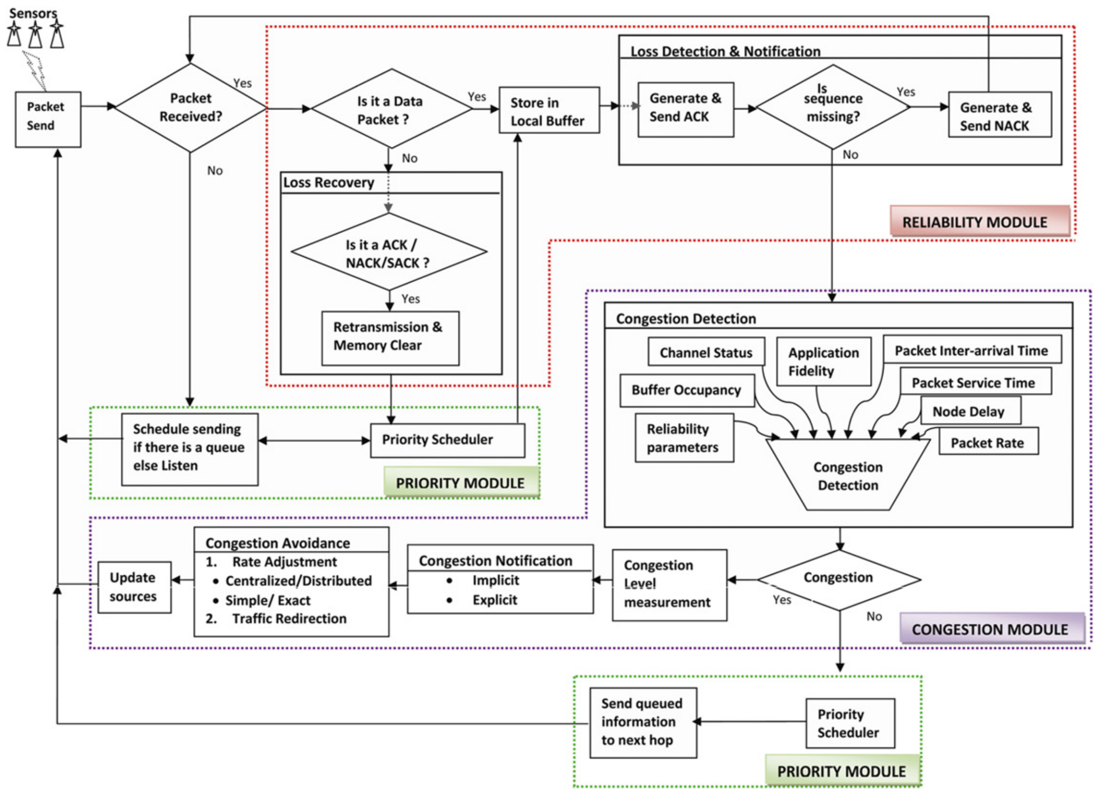

Fig. 1. Generic structure of transport layer protocol.

Table 2
Comparison of technical features of existing transport protocols.

Protocols	**Congestion detection**	**Congestion notification**	**Congestion avoidance**	**Reliability direction**	**Reliability level**	**Loss recovery**	**Loss detection and notification**	**Diverse traffic precedence**
Both reliability and congestion control								
TRCCIT (2010)	Packet rate	Imp	Rate adjustment	Up	Packet	Hop-by-hop	iACK, eACK	Yes
CRRT (2009)	Queue occupancy Packet rate	Imp	Rate adjustment	Up	Packet	End-to-End Hop-by Hop	NACK, MAC	–
CTCP (2008)	Transmission error loss Queue occupancy	Exp	Rate adjustment	Up	Packet	Hop-by-hop	eACK Double eACK	Yes
RT^2 (2008)	Node delay Queue occupancy	Imp	Rate adjustment	Up	Packet	Hop-by-Hop	SACK	Yes
ART (2007)	ACK received to set of nodes (core)	Imp	Reduce traffic of set of nodes (non-core)	Both	Event	End-to-End	eACK, NACK	–
RCRT (2007)	Time to recover loss	Imp	Rate Adjustment	Up	Packet	End –to End	NACK, cumulativeACK	–
FLUSH (2007)	Queue occupancy Link interference	Imp	Rate adjustment	Up	Packet	End-to-End	NACK	–
DST (2006)	Node delay Queue occupancy	Imp	Rate adjustment	Up	Event	End-to-End	–	Yes
PORT (2005)	Node price Link-loss rates	Imp	Traffic redirection Rate adjustment	Up	Event	–	–	–
ESRT (2005)	Queue occupancy	Imp	Rate adjustment	Up	Event	–	–	–
STCP (2005)	Queue occupancy	Imp	Rate adjustment Traffic redirection	Up	Packet	End-to-End	NACK, eACK	Yes
Reliability only								
ERTP (2009)	–	–	–	Up	Packet	hop-by-hop	iACK, eACK	–
GARUDA (2008)	–	–	–	Down	Packet Destination	Two-tier loss recovery	NACK	–
DTSN (2007)	–	–	–	Up	Packet	End-to-End	eACK, NACK	–
RBC (2005)	–	–	–	Up	Packet	Hop-by-hop	iACK	–
DTC (2004)	–	–	–	Up	Packet	Hop-by-Hop	eACK, SACK	–
RMST (2003)	–	–	–	Up	Packet	Hop-by-Hop End-to-End	NACK, MAC	–
PSFQ (2002)	–	–	–	Down	Packet	Hop by Hop	NACK	–
Congestion control only								
PHTCCP (2008)	Packet service ratio	Imp	Rate adjustment	–	–	–	–	Yes
PCCP (2006)	Packet inter-arrival time Packet service time	Imp	Rate adjustment	–	–	–	–	Yes
Siphon (2005)	Queue occupancy application fidelity	–	Traffic redirection	–	–	–	–	–
Fusion (2004)	Queue occupancy	Imp	Rate adjustment	–	–	–	–	–
CCF (2004)	Packet service time	Imp	Rate adjustment	–	–	–	–	–
Trickle (2004)	–	–	Polite gossip	–	–	–	–	–
CODA (2003)	Queue occupancy Channel status	Exp	Drop Packets Rate adjustment		–	–	eACK	–
ARC (2001)	Successful/unsuccessful delivery of packets	Imp	Rate adjustment	–	–	–	–	–

predefined threshold due to excessive incoming packets, probable congestion scenario can be forecasted, which gives negative consequences such as buffer overflows followed by packet collision and packet drops.

- *Packet rate*: The number of packets received or sent within specified time is referred as the packet rate. If the packet incoming rate exceeds the packet forwarding rate, the packet overflow is possible as the WSN motes have limited memory storage.
- *Packet service time/packet inter-arrival time*: Packet service time refers to the time interval between the arrival of the packet at the node and successful transmission of the last bit of the same packet, in fact the time duration it takes to process one packet at a node. The packet inter-arrival time is the time interval between the two sequential arriving packets from either source or for the transit traffic. If the packet service time exceeds the packet inter-arrival time, it leads to queue build up and packets will have to suffer from long queue delays.
- *Node delay*: Node delay refers to the delay expected by each packet at each node. Node delay at the sensors reveals how busy the surrounding area of sensor node is, and packets get delayed than expected if the congestion occurs.
- *Channel status*: The channel status gives an idea about how busy the channel is, and the interference of surroundings, which eventually reveals whether the channel is ready to transmit and receive data without resulting in congestion. Intra-path interference occurs when transmissions of the nodes interfere with the successor's reception, which prevents the reception of the following packet from a predecessor node.
- *Application fidelity*: Application fidelity is a concept of quality, which represents a range of operational measures including packet latency, number of successful event detections, data quality, and redundancy. If the fidelity of received data is below the perceived performance, the congestion is assumed. For example, if the packet latency at a node is higher than the expected latency value, there may be a congestion in the route.
- *Reliability parameters*: Reliability parameters for congestion detection is true if packets are dropped frequently and delayed retransmissions occur due to the congestion. The most common reliability parameters used by different protocols are time to recover the packet loss, loss rate, the number of transmission attempts made before a packet is delivered, and the reception of acknowledgements within time-out.

Now we discuss how different protocols use these parameters to detect congestion. STCP, ESRT, and Fusion solely detect the congestion when the buffer usage is higher than the predefined threshold, whereas RT2 and DST monitor the node delay threshold in addition to the buffer occupancy. CTCP uses both transmission error loss rates and the buffer usage. ARC, ART, PORT, and RCRT detect the congestion based on feedback parameters of the reliability module. For example, ART assumes congestion if ACK is not received to selected dominating sensors ("*essential nodes*") within timeout and similarly ARC monitors unsuccessful packet deliveries at sink. RCRT detects congestion based on the time to recover the packet loss and PORT uses the number of transmission attempts made before a successful delivery, also termed as 'node price' (Zhou and Lyu, 2005), and the loss rates of the links. CODA, Siphon, and Flush consider channel condition in congestion detection; Flush measures the intra-path interference (Kim et al., 2007) at each hop. In CODA and Siphon, sensors listen to the channel only when the buffer occupancy is high, trace the channel busy time and calculate the local channel loading. In addition, Siphon also checks whether the event detection rate at the sink is below the perceived application fidelity. In contrast, PCCP identifies the congestion when packet service time (PST) is higher than the packet inter-arrival time (PIAT) at the MAC layer. PHTCCP relies on the rate ratio calculated using PST at the MAC layer, and CCF uses PST at transport layer. TRCCIT identifies congestion when the packet incoming rate is higher than the packet outgoing rate and CRRT assumes congestion when nodes experience the reduced packet forwarding rate and excessive buffer usage.

Altogether, the most common techniques in congestion detection would be to use the buffer occupancy. But the mere prediction of congestion based on the high buffer usage of specific sensor is not sufficient. Because even though the buffer usage is low, the sensors may experience congestion due to the network traffic among other sensor nodes in the neighbourhood as shared communication medium nature of WSNs. Therefore it is necessary to realize the network channel condition around sensors. However other than the channel status and buffer occupancy, the other influential factors like PST, PIAT and node delay also must be considered to estimate an accurate congestion degree. Some protocols detect the congestion based on the reliability parameters. Solely depending on such factors like time to recover the loss or successful reception of packets is not very feasible, as a single packet drop can be occurred not due to the congestion but may be due to other factors like link quality issues. In such cases, the false detection of congestion may force the rate reduction, hence negatively affect the performance.

2.1.2. *Congestion notification*

Communication of the congestion occurrence to the neighbouring motes for further analysis and decision-making is essential in effective congestion control mechanism. This notification may be either single bit binary information, which gives whether there is congestion or not, or some measurement of congestion index/level. The congestion index related measurements can include buffer occupancy level, packet service time to packet inter-arrival time ratio, etc. The congestion warning is notified to other nodes explicitly or implicitly.

- *Implicit notification*: the congestion warning is embedded in the header of the normal data packets and the child nodes listen to their parent node to get the congestion information.
- *Explicit notification*: This is a special control packet that warns the congestion to its neighbouring nodes.

Most of the existing transport protocols implicitly notify the congestion, whereas CTCP and CODA send explicit notifications. CTCP generates control messages when congestion occurs as well as when congestion is resolved. In CODA, a suppression message is sent to their upstream neighbours via a backpressure method. When we compare two notification techniques; implicit and explicit, the former is more power efficient as it avoids the overhead associated with control messages.

2.1.3. *Congestion avoidance*

Congestion avoidance means to alleviate the network congestion, in fact to avoid the bottlenecks in smooth data transfer of wireless link. Once the congestion notification is received at the nodes, the control loop for congestion avoidance is initiated and the sensors are updated with the congestion mitigation decision in order to ease down the network congestion. Transport protocols are designed with three different congestion avoidance techniques, which two common techniques; rate adjustment and traffic redirection, and one rarely used mechanism; polite gossip policy.

- *Rate adjustment*: Rate adjustment refers to regulating the transmission rate of the congested sensors upon the reception of the congestion notification. Based on the location at which the rate adjustment plans are implemented, the rate adjustment schemes are categorized as either centralized or distributed. In centralized rate adjustment, the control decisions are made centrally, usually at the sink. In distributed rate adjustment, the control decisions are made at each hop of the network. From existing protocols, CRRT, RCRT, ESRT, and DST, follow centralized rate adjustment scheme, whereas STCP, CODA, Flush, ARC, ART, Fusion, CCF, PCCP, RT2, TRCCIT, and PHTCCP use decentralized scheme. When we compare the centralized scheme with distributed scheme, the centralized scheme may take unbiased decision about the rate, since the sink has the broader view of the network and controls the aggregate rate of the network. It is also more energy efficient to perform decision making tasks at the base station, as the sensor nodes are energy constrained and limited in computational abilities. On the other hand, distributed scheme may reduce the congestion quicker as the rate is adjusted at each hop.

On the other hand, different transport protocols use different rate control algorithms, which can be broadly categorized in to two; simple rate adjustment like additive increase multiplicative decrease (AIMD) and exact rate adjustment. In simple rate adjustment, merely a single congestion notification bit is used to notify the congestion. The congestion bit is enabled and cleared based on the congestion occurrence. For example, protocols like STCP, Flush, CODA, ESRT, RCRT use AIMD policy or its variants for rate adjustment, which increases the reporting rate in additive manner if successful data transmissions occur, and reduces the rate in multiplicative style if congestion occurs. Additive increase additive decrease (AIAD) is another such notion used by CRRT, in which the rate is reduced in additive manner in congested scenarios, avoiding aggressive multiplicative rate reduction in AIMD. Fusion also uses similar method, which reduces the packet sending rate for congested nodes to zero when the congestion is heard, which avoids the packet transmission to congested nodes. ART also temporarily terminates the traffic of "non essential" nodes (Tezcan and Wang, 2007).

On the other hand, in exact rate adjustment method, the rate is adjusted based on the information feedback obtained from the neighbours, implementing more accurate rate adjustment plan. Nodes provide congestion related estimations other than just notifying whether there is congestion or not. This estimated information includes congestion degree, acceptable data rate and delay parameters, etc. For example, CCF, and TRCCIT inform the allowable data rate, which should be updated in next data transmission. PCCP and PHTCCP inform the congestion degree and RT^2 and DST inform the delay constrained reliability parameter (Gungor and Akan, 2006), which the rate is adjusted accordingly.

When we evaluate the two techniques; simple rate adjustment technique is not feasible since it is difficult to precisely adjust the transmission rate using limited information given by the binary congestion notification bit. And also AIMD like local rate control policies also result greater sending rates at the sources closer to the sink when compared to other nodes, causing uneven number of packet reception at the sink, while negatively influencing the fairness and link utilization. Therefore the exact rate control technique is more suitable to implement more accurate rate adjustment plan.

- *Traffic redirection*: In traffic redirection, the nodes dynamically allocate its outgoing traffic to the uncongested paths. The congested paths are avoided using the feedback information such as high loss rates of those links, which are obtained from neighbouring nodes. For example, Siphon solely depends on traffic redirection for congestion control and, PORT and STCP consider it in addition to the rate adjustment. PORT selects the alternative paths, based on the node prices and the loss rate feedback obtained from neighbours. STCP uses the congestion bit enabled acknowledgement packets to realize the congested path. Siphon distributes virtual sinks (Wan et al., 2005) across the sensor network, which siphons the events with high traffic loads.
- *Polite gossip policy*: Polite Gossip policy works in such a way that, each node tries to broadcast a summary of its data to local neighbours periodically, but if nodes hear identical data from neighbours it "politely" suppress its own broadcasting. If a new code of data is received, nodes shorten their broadcast period to broadcast the new code sooner. When motes hear older data than its own, the protocol sends small piece of packets to update nodes. This method is only utilized by Trickle to avoid congestion.

2.2. Reliability module

Reliability in the context of transport protocols refers to the successful delivery of each segment that the sources generate to the ultimate destination. The reliability module must efficiently detect the packet drops and retransmit these packets to relevant sources. The packet drops can be incurred due to the numerous reasons such as, congestion scenarios, poor channel conditions like low signal to noise ratio (SNR) at receiver, and blind mote problem, where the interfering mote is in the receiving radio of the receiving mote communicating with another mote. As illustrated in Table 2, TRCCIT, CRRT, CTCP, RT^2, ART, RCRT, Flush, DST, PORT, ESRT, STCP, ERTP, GARUDA, DTSN, RBC, DTC, RMST, and PSFQ protocols claim to achieve reliability.

2.2.1. Reliability direction

In WSN, data transfers occur in two directions. When sensors detect an event, they inform their sensed information to the sink node. Sink also sends control packets or query messages to the sources. To satisfy the reliability for these scenarios, transport protocols offer upstream reliability and downstream reliability, respectively.

- *Upstream reliability*: Upstream reliability refers to the successful delivery of dataflow traffic from sources to sink, which is mostly unicast/convergecast transmission. All the protocols except PSFQ and GARUDA offer upstream reliability.
- *Downstream reliability*: Downstream reliability refers to the successful delivery of control packets and queries from sink to sources, which is mostly multicast/broadcast transfer. Only PSFQ and GARUDA offer downstream reliability.
- *Bidirectional reliability*: Bidirectional reliability means satisfying reliability in both directions, upstream and downstream. In reliability point of view it will be more perfect if the bidirectional reliability can be achieved, but only ART satisfies bidirectional reliability.

2.2.2. Reliability level

The level of reliability means the extent of reliability supported by the protocol. Three levels of data reliability can be defined as follows:

- *Packet reliability*: Packet reliability refers to the successful delivery of all the packets to the destination. It is necessary in certain control driven application scenarios, e.g. continuous humidity monitoring for a control process, etc., every sensed information is of pivotal nature and any loss of information may result in process malfunction. All the protocols discussed in this article, concern on packet reliability except PORT, ART, ESRT, and DST.

- *Event reliability*: Event reliability refers the successful event detection. For example, if more than one sensor in the field senses the temperature and reports to the sink, it is expected that at least one packet will be delivered and the successful delivery of each and every packet is not necessary. Only PORT, ART, ESRT, and DST offer event level reliability.
- *Destination reliability*: This concerns to send the message successfully only to the specific nodes or a selected cluster in entire WSN network. GARUDA offers destination reliability in addition to the packet reliability.

As a whole, the packet reliability is much more trustworthy than the event reliability as it guarantees the delivery of each and every bit of information. But on the other hand, packet level reliability involves more energy utilization. Therefore the protocol design should be more flexible to adapt both event and packet reliability depending on the targeted application.

2.2.3. Loss detection and notification

In reliable data transport, every packet loss should be identified by the receiver and should inform to the corresponding data storage mote or to the relevant source for retransmission. When a packet is dropped, a common mechanism for the packet loss detection would be to use packet sequence numbers in identifying packet drops. This is done in such a way that the source embeds packet header with two fields; source identifier and sequence number. Upon the reception of packets, the destination checks the sequence number and once a gap is detected in the sequence numbers, it determines the packet corresponding to the missing sequence number is lost. Generally, the protocols notify the packet losses, using two types of feedbacks namely positive acknowledgements (ACK) and its variants and negative acknowledgements (NACK). Here we discuss how existing protocols use different loss notification techniques to achieve the desired loss recovery.

- *ACK based system*: Positive acknowledgements are sent as explicit control packets (eACK) or as implicitly (iACK) in order to confirm the successful reception. The node generates eACK control packets for all the packets received or single packet for multiple fragments received (Cumulative eACK). Implicit acknowledgement (iACK) is the interpretation for the transmitter's ability to overhear the forwarding transmissions in physical wireless links, which iACK piggybacks ACK in the packet header. Another variant of ACK is SACK, which the receiver sends SACK to inform the sender about all segments that have arrived successfully, effectively this notifies the last fragment received in-order. So the sender needs to retransmit only the segments that have actually been lost. For example, CTCP achieves two hop by hop reliability levels; fist level is obtained using ACKs and the second higher reliability level is obtained using both ACKs and 'double ACKs'. The term 'double ACK' means that a node sends ACKs to the preceding node once the data are received from it and also when the ACK is obtained from the following node for transmitted data. RBC uses iACK with windowless block acknowledgement scheme. ERTP use stop-and-wait hop-by-hop iACKs that the sender retransmits the packet if it does not hear the iACK within the timeout and also use ACKs to confirm the packet reception at sink. TRCCIT also uses timer driven iACK in similar way and the receiver notifies the reception of the retransmitted packet using ACK. On the other hand DTC, an attempt to optimize TCP, identifies successful packet delivery by using SACK and ACK. RT^2 also achieves loss recovery solely using SACK.
- *NACK based system*: NACKs are sent for the missing sequence numbers in received stream. NACK can be generated as single or a range of lost fragments which is referred as notion of loss window. For example, in Flush, the sink sends a single cumulative NACK packet, which can hold up to three sequence numbers corresponding to three missing packets. CRRT achieves end-to-end reliability using NACK and hop-by-hop reliability using MAC layer retransmissions. Loss detection in RMST is timer driven and implements both end-to-end and hop-by-hop selective request NACK. RMST also considers MAC level Automatic Repeat Request (ARQ) for the reliability of control packets. GARUDA utilizes a NACK-based two stage recovery process; which the first stage deals with recovering all lost packets in a set of dominating nodes (core nodes). In second stage, non-core nodes requests retransmission from core nodes, only after the completion of all the retransmissions of its core node. In PSFQ, loss repair request is made using NACK messages. If the packets are lost in a bursty event, single fetch would be sent, by aggregating the loss windows with missing sequence number pairs.
- *ACK and NACK based system*: STCP that differentiates the dataflow as continuous and event-driven uses NACKs for continuous flow and ACKs for event-driven flow to ensure the successful data delivery. ART recovers upstream data using ACK, and downstream queries using NACK. DTSN achieves full reliability level based on selective repeat ARQ, employing both ACK and NACK and differentiated reliability level using ARQ together with Forward Error Correction (FEC) strategy. RCRT employs NACK for loss recovery, but also sink sends a cumulative ACK sequence number to clear the retransmit buffer.

2.2.4. Loss recovery

The loss recovery means repairing the packet drops by means of packet retransmission. The loss recovery can be categorized into two as follows:

- *End-to-end loss recovery*: In end-to-end loss recovery mechanisms, the end points are responsible in loss detection and notification. Only the source caches the packet information and the generation of repair requests occurs only at sinks. Relevant source retransmits the packet upon the reception of repair request. As we see CRRT, RCRT, Flush, STCP, ART, DST, and DSTN offer end-to-end loss recovery.
- *Hop-by-hop loss recovery*: In hop-by-hop loss recovery method, the intermediate nodes cache packet information and perform loss detection and notification. The loss packet recovery requests are initiated at each hop, both caching and non-caching nodes. Once the cashing node obtains the repair request, it initiates the retransmission. TRCCIT, CRRT, DTC, CTCP, RT2, PSFQ, RBC, ERTP, RMST, DTC, and RBC use hop-by-hop method for loss.

In comparison of above two methods, end-to-end loss recovery is not very feasible in large networks with multiple hops due to energy consuming retransmissions. This also causes the loss of control messages as it flows through number of hops. Hop-by-hop mechanism consumes lesser amount of energy as only two adjacent nodes involve in loss recovery. Eventually it alleviates the congestion at a quicker rate. But the hop-by-hop technique utilizes more memory since each intermediate sensor maintains a cache to store data. From the transport protocols discussed, PORT, DST and ESRT dealing with event level reliability, do not focus on loss recovery mechanisms.

2.3. Priority module

Source prioritization differentiates diverse sensors by means of introducing precedence levels to different sensors, by assigning flow-identifiers or application-identifiers to reflect the importance of each sensor. For example; it is important to assign higher priority to the event driven information compared to synchronous information sensing. Data prioritization is critical in WSN that supports heterogeneous applications having mixed traffic flows to achieve the application specific QoS objectives. In present research community, packet prioritization has achieved more attention due to its capability to implement weighted fairness. In weighted fairness more bandwidth is assigned for sensors handling more critical applications, whereas in simple fairness equal bandwidth is allocated among all sensor nodes. Therefore the importance of weighted fairness outweighs the conventional simple fairness in combined traffic flows. As illustrated in Table 1, TRCCIT, PCCP, PHTCCP, STCP, CTCP, DST, and RT^2 claim to achieve flexible differentiation of source traffic by means of precedence levels or application/flow identifiers. These are assigned based on nature of flows, importance of application, and packets' remaining time to deadline.

2.3.1. Priority scheduler

The prioritization scheduler differentiates the source information based on the nature of the flow or the application [CTCP, STCP, TRCCIT], the precedence level of the source [PHTCCP, PCCP] and information of time to live or the remaining time to deadline [RT^2]of the packets, etc. The schedulers at intermediate nodes arrange the received packets based on the above factors and forward the scheduled queued data to the next hop. For example, PCCP defines local source traffic priorities and transit traffic priorities at each node. Similarly PHTCCP also assigns the precedence levels to diverse sensors. DST and RT^2 determine the packet's remaining time to deadline and the event packets are given high priority at the nodes, as their remaining time to deadline decreases. In TRCCIT, the sources define tunable reliability levels based on the application requirement. STCP differentiates the source traffic as continuous and event-driven and obtains two different reliability levels. Similarly CTCP also assigns two different reliability levels for heterogeneous applications (two different).

3. Benchmarking

As illustrated in Section 2, different protocols are featured with different technical attributes and functionalities. The uniqueness of each new protocol more or less lies in one of those features. According to the literature, the researchers have demonstrated the uniqueness of their newly designed protocols by showing the performance improvements in different simulation/experimental environments. However the simulation/experimental settings used by different researchers are dissimilar. Therefore the direct performance comparison among the protocols is difficult, hence to prove one's protocol is better than the rest is challenging. In this section, we study the experimental design space of different protocols and attempt to suggest a benchmark for experimental/simulation settings.

3.1. Experimental/simulation settings

In this section, we discuss the experimental/simulation settings of diverse protocols such as number of sensors, sensor deployment, packet size, simulation area, buffer size and power levels (Table 3). Most of the performance overhead claimed by researchers depends on these parameters. However we came across with some contributions, which did not publish some of their simulation settings they used to claim their performances.

The number of sensor nodes distributed in network is mainly determined by the factors like the nature and the size of the area of interested network. For example, in indoor network applications like home automation, not many sensors are required to cover a small area with fewer obstacles. If it is an outdoor network like rainforest or an agricultural field, the size of the region as well as the amount of disturbances may be higher. Therefore it may require more sensors. As illustrated in Table 3, most protocols [TRCCIT, RT2, ART, DST, STCP, ESRT, ERTP, GARUDA, PHTCCP, CCF] have attempted to perform their simulations using higher number of sensors, which is equal or more than 100 sensors. Ability to perform successfully in huge network proves the scalability of the design.

Deploying large number of sensors needs a careful design of topology. Sensor nodes may be deployed in physical environment either in random locations [CRRT, CTCP, RT2, ART, Flush, DST, PORT, STCP, ESRT, ERTP, PHTCCP, CCF, CODA] or in deliberately selected locations based on pre defined plans (e.g. tree, grid) [TRCCT, RCRT, GARUDA, RBC, RMST, PSFQ, PCCP, Siphon, Fusion]. The ad-hoc distribution, which is utilized by many protocols here, is used in most practical scenarios, mostly for the establishments where human interaction is low.

The coverage area is referred to the area covered by the effective communication range of the sensors. Based on the coverage area presented by different protocols, we can deduce the suitability of the protocol for different physical locations. For example we can assume that the protocols: TRCCIT, CRRT, CTCP, RT2, ART, DST, STCP, ESRT, ERT, PSFQ, PHTCCP, Siphon, which show lesser covering area (let's take the area less than $300 \times 300\ \text{m}^2$) may be suitable for the applications with small and medium size establishments like buildings and bridges, etc. The protocols, PORT and GARUDA, that exhibit higher area may be suitable for larger regions like forests and agricultural fields.

It is necessary to ensure that transport layer protocol is capable of handling large size data packets, while maintaining desired performance. The simulation with large packet size evidences its capability to handle complex application packets like multimedia without loss of the quality of output, which can be resulted due to fragmentation. From these protocols, very few offer higher packet size [ART, GARUDA], and all other protocol simulations have been done with packet sizes less than 100 bytes. Buffer size in a WSN mote means the maximum available storage locations in memory. The buffer motes use to store the incoming packets and initiate retransmission in loss recovery process. When the buffer level is low, the possibility of congestion occurrence is high. Since the practical WSN motes contain limited storage, the transport protocol simulation done in software environment also should utilize low buffer lengths to match with the actual environment.

Simulation time is the time period which the protocol has been run before obtaining the performance results. In fact, if the protocol runs for a longer duration without a failure, and if the protocol performance remains unaffected for a long duration of time, it shows the stability and sustenance of the WSN. However according to Table 2, very few protocols (CCF, Siphon, STCP, RCRT, CTCP, RT^2) have run their simulation for at least 1000 s. Another factor that decides the performance of the WSN network is that its ability to handle a large traffic load. This factor is decided by the number of source nodes that transmit simultaneously, and the packet rate of source nodes. As given in the table, it is visible that different protocols have used different initial packet rates for their

Table 3
Comparison of design space of existing transport protocols.

Protocols	Applications	Sensor deployment	Number of sensors	Packet size	Coverage area (m²)	Buffer size	Simulation time	Traffic load (initial tx rate)
Both reliability and congestion control								
TRCCIT (2010)	Heterogeneous concurrent multiple applications	Grid	100	29 bytes	60 × 60	36	–	20 msgs/s
CRRT (2009)	High-rate applications: imaging, acoustic localization	Ad-hoc	80	32 bytes	100 × 100	40	250 s	0.5–4 pkts/s
CTCP (2008)	Heterogeneous concurrent multiple applications	Ad-hoc	25	216 bits	50 × 50	–	1000 s	1/50 pkts/s
RT^2 (2008)	Heterogeneous concurrent real-time applications: target tracking, chemical attack detection	Ad-hoc	200/sources: 41, 62, 81, 102	30 bytes	200 × 200	65	1000 s	1, 5, 10, 15, 20 msg/s
ART (2007)	Mission critical applications like country border security	Ad-hoc	100	100 bytes	300 × 300	50	150 s	Query frequency: 2–10 s. Event frequency: 0.1–1 s
RCRT (2007)	High-rate applications: imaging, acoustic localization	Tree	40	64 bytes	–	–	1800–3600 s	Up to 1.2 pkts/s
FLUSH (2007)	Bulk data collection applications: volcanic activity monitoring	Ad-hoc	79 sensors	35 bytes	–	–	> 45 s	100, 200, 400 pkts/s
DST (2006)	Heterogeneous real-time applications: border surveillance and intrusion detection	Ad-hoc	200/sources: 41, 62, 81, 102	30 bytes	200 × 200	65	–	1, 0.1, 0.01, 0.001 pkts/s
PORT (2005)	General sensing application	Ad-hoc	100	36 bytes	1350 × 1350	50	500 s	–
ESRT (2005)	Event detection applications: signal estimation/tracking	Ad-hoc	200/sources 41, 52, 62	30 bytes	100 × 100	65	60 s	10, 1, 0.1, 0.01, 0.001 pkts/s
STCP (2005)	Heterogeneous concurrent multiple applications	Ad-hoc	50,100	–	100 × 100	–	5000 s	1/50 pkts/s
Reliability only								
ERTP (2009)	Data streaming applications: weather and habitant monitoring	Ad-hoc	200	40 bytes	180 × l80	–	200 s	1/60 pkts/s
GARUDA (2008)	Downstream reliability applications	grid	100	1 kb	650 × 650	–	–	25 pkts/s
RBC (2005)	High-volume bursty traffic applications	Grid	49	–	–	–	40 s	Up to 14 pkts/s
RMST (2003)	Applications require fragmentation/reassembly like multimedia	Grid	21	50–100 bytes	–	–	–	–
PSFQ (2002)	Downstream slow fetch reliability applications	Linear	13	50 bytes	100 × 100	–	100 ms	100 pkts/s
Congestion control only								
PHTCCP (2008)	Heterogeneous concurrent multiple applications	Ad-hoc	100	29, 33, 41, 64 bytes	100 × 100	10	60 s	4–16 pkts/s
PCCP (2006)	Heterogeneous concurrent multiple applications	Tree, Linear	7	–	–	–	60 s	–
Siphon (2005)	Generic data dissemination application	Grid	48	–	–	–	1800 s	1/6 pkts/s
Fusion (2004)	High-volume bursty traffic applications and fairness	Tree	55	–	1493	–	–	0.25–4 pkts/s
CCF (2004)	Fairness in applications: large area temperature monitoring	Ad-hoc	116	30 bytes	–	10	50,000 s	0.5 pkts/s
CODA (2003)	General sensing application	Ad-hoc	30	64 bytes	–	–	30 s	–

performance evaluation. Please note that, here we capture only the initial transmission rates and this value can be changed during the simulation due to the rate change plans developed by certain protocols.

3.2. Focussed application overview

In recent research community, WSNs support an extensive range of constructive applications. Some of the many potential applications include military sensing, traffic surveillance, video surveillance, industrial and manufacturing automation, process control, inventory management, weather sensing, environment monitoring, national border monitoring, and building and structures monitoring. Different transport protocols are designed to focus on specific group of application support [Flush, and ERTP] and some researches [STCP, RT2, PHTCCP, PCCP, and DST] have attempted to build concurrently running multiple heterogeneous applications in one network.

To obtain optimal performance of the application, the transport protocol must have both efficient and effective congestion control as well as the reliability support mechanisms. Generally most of the WSN applications at least require one of those for successful performance. For example, the end-to-end packet drop rate and effective loss recovery mechanism would be an important measure of performance in the applications that are designed to detect discrete, non-repeating events. Such application would be structure monitoring, where the each and every bit of data from all measuring points is necessary to build a model. Therefore we can assume that the protocols offering efficient loss recovery models are more suitable for such applications. On the other hand congestion creates more adverse effects in applications like environment and habitat monitoring, where the data is periodically collected by sensor and route towards the sink. Here the congestion scenario is possible mainly due to the funnelling effect, which many sensors simultaneously attempt to transmit data towards one sink. Thus we can conclude that the protocols that implement congestion control techniques are more feasible for such applications. But all most all the re-tasking and critical time-sensitive monitoring and surveillance operations essentially require both congestion control and reliability.

As illustrated in Table 3, some existing protocols are designed to achieve the specific complex requisites of their focused application or the application group, other than just targeting the basic application needs like reliability support or congestion control. For example, the high end military applications such as battle field surveillance, biological or chemical attack detection and intrusion detection requires real time data transfer, timely reporting of sensor values, and reliable event detection [RT2, DST]. The protocols designed for data streaming applications [ERTP] such as WSNs work independently for long duration of time to monitor the weather conditions may consider energy efficiency, end to end reliability and long-term operation and does not consider much on the end-to-end transmission latency as it is not a key concern in such applications. For the mission critical applications like country border security, etc., which are driven by the queries from the sink, the protocols that offer high downstream reliability [GARUDA, PSFQ] or bidirectional reliability [ART] are more suitable. Certain protocols focus on high-rate loss-intolerant applications [CRRT, RCRT, and RBC] like imaging, structural monitoring and acoustic localization. In these applications several sensors transmit large amount of information simultaneously towards the sink, resulting congestion and the protocols must be capable to handle bulk data without congestion collapse and to offer desired reliability. This high volume bursty traffic can also be resulted due to volcanic activity monitoring, which collect data in a coordinated fashion and may cover larger physical areas and have large network depths (Fusion). Certain applications may require the same amount of measurements from geographically dispersed sensors over a large period of time in large physical area such as temperature and humidity monitoring in rainforest. To cater such applications, the protocols must highly consider the data transmission fairness [CCF, Flush, PHTCCP, PCCP].

3.3. Experimental results

In this section, we attempt to compare the performance statistics published by prior research contributions on WSN transport layer protocols. According to those performance evaluations, the performance criteria used by different protocols are dissimilar and Table 3 illustrates the performance criteria addressed by different protocols.

Please note that sometimes the protocol proficiencies are indirectly shown by the researchers; but in this analysis we mostly consider the direct and clear performance results based on the criteria defined in Table 4. For examples, even though the protocols have not properly presented the number of collisions in their published results, one can assume that showing the success rate or the packet drop rate has some impact on the number of collisions. For instance, if the protocol exhibits a higher packet drop rate, one reason behind this can be the higher number of collisions followed by a weak loss detection, notification and retransmission schemes.

Accordingly, it is visible that different protocols have been evaluated in different performance criteria, which are derived from their application objectives. For example, ERTP develops a protocol for real-time reliable data streaming, hence the corresponding researchers have stressed the performance statistics on energy consumption, success rate, and latency. However it is complicated to directly compare the numerical performance results published by different protocols and conclude which protocol shows the highest performances. The main reason behind this is the different simulation/experimental settings used by different protocols. As an example, CRRT shows 1.5–1.75 packets/s average throughput over the 250 s simulation time with source data rates between 1.5 pkts/s onwards to 4 pkts/s, 80 sensor nodes, buffer length of 40 and packet size of 32 bytes. CRRT shows end-to-end delay of 300–350 ms and 90–95% of packet delivery ratio at 4 pkts/s source rate with maximum of 2 retransmission limits (CCRT performance is evaluated with 1–4 retransmission limits of dropped packets). On the other hand, TRCCIT has shows the performance at 0–20 msgs/s information rate, and achieves around 80% packet success rate and around 3 s of delay at 4 msgs/s. Although TRCCIT performances are appeared as lower performance than CRRT, it cannot be validated due to the different simulation settings that TRCCIT has used (100 sensor nodes, packet size of 29 bytes and buffer length of 36). Similarly, if we compare STCP with CTCP, STCP claims 2.78 J at 100% reliability mode, and CTCP claims around 12 mJ of energy consumption at its highest reliability mode (reliability level 2, the packet success rate is approximately 100%). Similar to the previous case, the direct comparison of STCP and CTCP performances also not practical due to the completely diverse settings utilized by these two protocols. For instance, although both protocols use same packet rate of 1 packet per 50 simulation seconds, the number of nodes and simulation time used by STCP (100 nodes and 5000 simulation seconds) is much higher than that of CTCP (25 nodes and 1000 simulation seconds).

In literature, we can find some protocols that attempt to compare the performance of new protocol with existing protocols. However most researchers have compared their protocols with commonly available TCP variants such as TCP-ELFN,

Table 4
Comparison of technical features of existing transport protocols.

Protocol	Throughput	Number of collisions	Success rate	Delivery latency	Energy consumption/ efficiency	Simulation environment
TRCCIT	X	X	✓	✓	✓	TOSSIM
CRRT	✓	X	✓	✓	✓	NS2
CTCP	X	X	✓	X	✓	TOSSIM
RT^2	✓	X	✓	✓	✓	NS2
ART	X	X	✓	✓	✓	NS2
RCRT	✓	X	✓	X	X	TinyOS, Experimental testbed
FLUSH	✓	X	✓	X	X	TinyOS, Experimental testbed
DST	✓	X	✓	✓	X	NS2
ESRT	✓	X	X	X	✓	NS2
PORT	X	X	X	X	✓	NS2
STCP	X	X	X	✓	✓	TOSSIM
ERTP	X	X	✓	✓	✓	NS2
GARUDA	✓	X	✓	✓	✓	NS2
DTSN	✓	X	✓	X	✓	OMNET++
RMST	X	X	✓	X	X	NS2
PSFQ	X	X	✓	✓	X	NS2
PHTCCP	✓	X	✓	X	✓	NS2
PCCP	✓	X	✓	X	X	–
Siphon	X	X	X	X	✓	NS2 and Experimental testbed
Fusion	✓	X	✓	✓	X	TinyOs Experimental testbed
CCF	✓	X	✓	X	X	Experimental testbed
CODA	✓	X	✓	X	✓	NS2

Table 5
Suggested benchmark for experimental settings.

Criteria	**Standard**			
Standard number of sensor nodes	*Small* 25	*Medium* 100		*Large* 200
Convergence area	*Low coverage* 50×50 m^2	*Medium coverage* 100×100 m^2		*High coverage* 300×300 m^2
Standard packet size		*Scalar* 30 bytes	*Multimedia/query* 1kb	
Topology		Ad-hoc		
Simulation time		1000 s		
Data transfer rate	*Usual scalar traffic* Starting rate: 10 pkts/s Step increment: 10 pkts/s		*High volume bursty scalar traffic/multimedia* Starting rate: 100 pkts/s Step increment: 100 pkts/s	

TCP-New Reno, etc., and very few researchers have attempted to show the higher performance of their protocols by comparing their protocol with one or two recently emerged protocols, which we focus in this article. For instance, the authors of CRRT have compared the performance with RCRT and the authors of PHTCCP have evaluated their protocol against CCF. Even though the best method of comparing the performance is simulating all the existing protocols in the same simulation domain used for the new protocol, this has become impractical due to the many reasons such as availability of higher number of WSN transport protocols, difficulty of obtaining the simulation source codes from the authors, unavailability of all the experimental/simulation settings in the publications, time constraints, etc. Therefore the best option would be to keep a single benchmark of experimental/simulation settings to simulate the future protocols.

3.4. Benchmark for experimental settings to evaluate the transport protocols

Here, we suggest a basic benchmark for the experimental/simulation settings based on the analysis we have performed in previous subsections (Table 5). One can claim that the researchers should have the choice to select the preferred set of simulation/experimental settings based on their application targets; but the real benefit here is that if all the researchers follow the same benchmark when simulating their protocols, it will be easier for the future researcher to compare his new protocol with the existing protocols just by referring to the published simulation results of exiting protocols. Hence they can easily find out the areas of improvements of their new protocols. In addition, this will save the invaluable time of the researcher.

The selection of values for this benchmark is based on the experimental settings used by prior researches and the general application scenarios applicable for different experimental settings.

Usually the small scale networks like small home networks or in-vehicle networks require around 5–20 sensor nodes for general control applications. For example, the usual number of sensors used in a basic in-vehicle environment is six sensors, namely the oxygen sensor, the air pressure sensor, the air temperature sensor, the engine temperature sensor, the throttle position sensor, and the knock sensor. However in currently available advanced in-vehicle networks, the number of sensors has been increased as the newly added reverse sensors, multi-media application sensors, curtain control sensors, etc. In addition, the current home networks are also equipped with at least 15–20 sensors and even more to perform different sensing and controlling applications. Therefore we decide to keep the standard of 20 sensor nodes for the small scale network simulations. Likewise our recommendation is to use 100 nodes to simulate the medium sized networks like schools and hotels, which are located in a larger geographical area and also have higher disturbances due to many buildings and inhabitants compared to the small scale networks. In addition, for the researchers who focus on large networks like rain forests and agricultural fields, our recommendation is to simulate the networks with 200 nodes. Here we assume that 200 sensor nodes are sufficient to fulfil very large scale applications as the protocols like DST and RT^2 use 200 nodes in their simulations and claim the suitability for large scale application areas like border surveillance and chemical attack surveillance.

As we discussed in previous section, the coverage area indicates how well the area is tracked by the sensors. If a higher area like very large rainforests or large country boarders is required to be tracked, we recommend using $300 \times 300\ m^2$ coverage areas in simulation settings. One reason to select this value is that, many protocols (RT^2, ERTP, ART, and DST) claim to satisfy the requirements of large scale applications like weather monitoring, and boarder surveillance using the coverage area between $150 \times 150\ m^2$ and $300 \times 300\ m^2$. Similarly we also define the coverage area as $100 \times 100\ m^2$ for a medium area like bridges and $50 \times 50\ m^2$ for a small area.

We define the standard packet size for the scalar data as 30 bytes, as this is the most commonly used data size by previous works. For the multimedia and large control and query applications, we define the standard of 1 kb packet size, as the body of the multimedia data and control queries is higher if sent in single or few numbers of packets. In addition, especially in large control queries, the fragmentation of a large packet to smaller packets can cause the loss of critical parts of the control information. In fact, GARUDA uses 1 kb packet size for the protocol simulation, claiming to tackle the problems faced by smaller packet sizes in large query and control message transfers.

The standard topology we used here is ad-hoc topology. This is because the ad-hoc topology is the most widely used topology by critical WSN applications compared to pre-planned topologies. For instance, the preplanned topologies like tree and grid are suitable for establishments where humans can easily interact to locate the sensor nodes. However, the human-accessibility to the most critical WSN application areas like chemical attack detection, and volcanic activity monitoring, is difficult. For such WSN networks, the practical topology is the ad-hoc topology.

We select 1000 s as the standard simulation time. If the protocol runs for a longer duration without a failure, and if the protocol performance remains unaffected for a long duration of time, it shows the stability and the sustenance of the protocol in long-term. However due to the time constraints in performing simulations, we limit this value to 1000 s as researchers (CTCP, RT^2, CCF) have successfully simulated their protocol for more than 1000 s, making this value practical in performing simulation.

The initial packet transmission rate decides the total traffic load transmitted to the network in start-up. Considering the initial packet rates used by different protocols, we set the standard to use 10 pkts/s initial packet transmission rate for usual scalar applications. If higher packet rates are required to satisfy high volume bursty traffic information (RBC, Flush), we recommend to use 100 pkts/s. We also recommend simulating the protocols in different initial packet rates; standard increment of 10 pks/s for usual scalar data, and 100 pkts/s for high volume bursty data, to find out the protocols proficiency in withstanding higher initial traffic loads. Please note that this packet transmission rate can be changed during the simulation process due to the different instructions offered in the algorithms such as efficient rate planning methods.

4. Conclusions

In this article, we present a generic framework for WSN transport protocol, an evaluation of technical and experimental attributes of existing transport protocols and a benchmark for experimental settings.

Based on the technical feature review, it is clear that even though there is huge number of transport protocol designs, several research issues such as cross-layer optimization, weighted fairness, and active queue monitoring in congestion control, require further attention in future researches. Among these protocols very few transport protocols such as RT^2, PCCP and PHTCCP enhance the performance using cross layer interaction. Some other protocols such as RMST and DTSN also rely on link layer ARQs but do not take much benefit from cross layering in increasing performance. Achieving application-specific QoS and weighted fairness that allocates more bandwidth for imperative sensors are crucial in handling heterogeneous applications. But only few protocols such as PCCP, PHTCCP DST, RT^2, STCP, and CTCP address the node prioritization and variable reliability levels for diverse sensors. Most congestion control mechanisms in current protocols monitor the channels and dynamically regulate the data transmission rate only when the congestion is detected. But it is vital to monitor the channel intelligently to control the possible anticipated congestion scenarios, before the real congestion occurs. Only few protocols like TRCCIT, and RT^2 concern on intelligent queue monitoring.

In addition to the technical feature review, we also review design space and the application overview of different protocols and suggest a benchmark for experimental settings. Since the different protocols have used different experimental settings in protocol simulation/experiment, it is difficult to directly compare the performance figures achieved by different protocols. This fact makes the necessity of having well-defined benchmark for simulation/experimental settings. If all the protocols are simulated based on a standard set of simulation/experimental settings, the future researchers can compare the performance of new protocols with existing ones by just referring to the published simulation results of existing protocols.

References

Alam M, Hong CS. CRRT: congestion-aware and rate-controlled reliable transport in wireless sensor networks. IEICE Transactions on Communications 2009;E92(B):184–9.

Culler D, Estrin D, Srivastava M. Overview of sensor networks. IEEE Computer 2004;37(9):41–9.

Dunkels A, Voigt T, Ritter H, Alonso J. Distributed TCP caching for wireless sensor networks. In: Proceedings of the 3rd annual mediterranean ad hoc networking workshop. Turkey, 2004.

Ee CT, Bajcsy R. Congestion control and fairness for many-to-one routing in sensor networks. In: Proceedings of 2nd international conference on Embedded networked sensor systems. Baltimore, MD, USA, 2004. p. 148–61.

Giancoli E, Jabour F, Pedroza A. CTCP: Reliable Transport Control Protocol for sensor networks. In: International conference on intelligent sensors, sensor networks and information processing, 2008. p. 493–8.

Gouda MG. Reliable bursty convergecast in wireless sensor networks. In: Proceedings of the 6th ACM international symposium on mobile ad hoc networking and computing (ACM Mobihoc), USA, 2005. p. 266–76.

Gungor VC, Akan OB. DST: delay sensitive transport in wireless sensor networks. In: Proceedings of seventh IEEE international symposium on computer networks. Istanbul, Turkey, 2006. p. 116–22.

Hull B, Jamieson K, Balakrishnan H. Mitigating congestion in wireless sensor networks. In: Proceedings of the 2nd ACM conference on embedded networked sensor systems (ACM SenSys). USA, 2004. p. 134–47.

Justin J, Atiquzzaman M. Transport protocols for wireless sensor networks: state-of-the-art and future directions. International Journal of Distributed Sensor Networks 2007;3(1):119–33.

Kim S, Fonseca R, Dutta P, Tavakoli A Culler D Levis P et al. Flush: a reliable bulk transport protocol for multihop wireless networks. In: Proceedings of the 5th international conference on Embedded networked sensor systems. Sydney, Australia, 2007. p. 351–65.

Le T, Hu W, Peter Corke, Jha S. ERTP: energy-efficient and reliable transport protocol for data streaming in wireless sensor networks. Computer Communications 2009;32:1154–71.

Levis P, Patel N, Culler D, Shenker S. Trickle: a self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In: Proceedings of the 1st conference on symposium on networked systems design and implementation. San Francisco, California, USA, 2004. p. 29–31.

Marchi M, Grilo A, Nunes M. DTSN—distributed transport for sensor networks. In: Proceedings of IEEE symposium on computers and communications (ISCC). Aveiro, Portugal, 2007.

Monowar MM, Rahman MO, Pathan ASK, Hong CS. Congestion control protocol for wireless sensor networks handling prioritized heterogeneous traffic. In: Proceedings of the 5th annual international conference on mobile and ubiquitous systems: computing, networking, and services. Dublin, Ireland, 2008.

Paek J, Govindan R. RCRT: rate-controlled reliable transport for wireless sensor networks. In: Proceedings of the 5th international conference on embedded networked sensor systems. Sydney, Australia, 2007. p. 305–19.

Park SJ, Vedantham R, Sivakumar R, Akyildiz IF. A scalable approach for reliable downstream data delivery in wireless sensor networks. In: Proceedings of the 5th ACM international symposium on mobile ad hoc networking and computing (ACM Mobihoc). Japan, 2004. p. 78–9.

Sankarasubramaniam Y, Akan OB, Akyildiz IF. ESRT: event-to-sink reliable transport in wireless sensor networks. In: Proceedings of the 4th ACM international symposium on mobile ad hoc networking and computing (ACM Mobihoc). USA, 2003. p. 177–88.

Shaikh FK, Khelil A, Ali A, Suri N. TRCCIT: tunable reliability with congestion control for information transport in wireless sensor networks. In: Proceedings of the international wireless internet conference (WICON). Singapore, 2010.

Stann F, Heideman J. RMST: reliable data transport in sensor networks. In: Proceedings of the 1st IEEE international workshop on sensor network protocols and applications (SNPA). USA, 2003. p. 102–13.

TCP. Retrieved August 2008: 〈http://www.ibiblio.org/pub/docs/rfc/rfc793.txt〉.

Tezcan N, Wang W. ART: an asymmetric and reliable transport mechanism for wireless sensor networks. International Journal of Sensor Networks 2007;2:188–200.

UDP. Retrieved August 2008: 〈http://tools.ietf.org/html/rfc768〉.

Wan CY, Eisenman SB, Campbell AT. CODA: congestion detection and avoidance in sensor networks. In: Proceedings of the 1st ACM conference on embedded networked sensor systems (ACM SenSys). USA, 2003. p. 266–79.

Wan CY, Eisenman SB, Campbell AT, Crowcroft J. Siphon: overload traffic management using multi-radio virtual sinks in sensor networks. In: 3rd international conference on Embedded networked sensor systems. San Diego, California, USA, 2005. p. 116–29.

Wang C, Sohraby K, Lawrence V, Li B, Hu Y. Priority-based congestion control in wireless sensor networks. Trustworthy Computing 2006a;1:22–31.

Wang C, Sohraby K, Li Bo, Daneshmand M, Yueming HuA. survey of transport protocols for wireless sensor networks. IEEE Networks 2006b;20(3):34–40.

Woo A, Culler DE. A transmission control scheme for media access in sensor networks. In: Proceedings of the 7th annual international conference on Mobile computing and networking. Rome, Italy, 2001. p. 221–35.

Zhang H, Arora A, Choi YR, Iyer YG, Gandham S, Venkatesan S. STCP: a generic transport layer protocol for wireless sensor networks. In: Proceedings of the 14th IEEE international conference on computer communications and networks (ICCCN). USA, 2005. p. 449–54.

Zhou Y, Lyu MR. PORT: a price-oriented reliable transport protocol for wireless sensor network. In: Proceedings of 16th IEEE international symposium on software reliability engineering. Chicago, 2005. p. 117–26.