Transport layer and quality of service

In WSN one key requirement is reliability, which refers not only to eventual delivery of data packets (transport reliability) but also to the ability to detect physical phenomenon of first place. Therefore, coverage of sensor n/w is important consideration.

→ This transport layer focus on protocols and approaches to deal with reliability in sensor n/w.

Transport layer & QOS in WSNs

In internet a large no. of users run various applications and each user judges ctr QOS individually. A large bytes of data through multiple hop travel one place to other by using transport protocol (TCP). The measures used for judging the quality of service are related protocols not application. These measures are delay, jitter, Packet loss rate etc. (sensor n/w vs Internet)

→ These things in sensor n/ws are different as sensor n/w are not only visualized as an infrastructure to transport data but also (nodes are tasked for collboratively monitoring and controling physical environment) Hence, the sensor nodes process each other data locally & while transit to sink node they have to know the data they forward. Therefore, user expect another service than they get from Internet & n/w ability to support these application dependent tasks is one cornerstone of QOS besides traditional n/w. (sensor n/w)

→ In Internet, n/w layer & transport layer play important role in achieving data transport reliability. Network layer offers best effort service and (transport layer is responsible for achieving reliable and in sequence delivery, congestion & flow control.)

→ In transport layer it focuses on reliability as one of the key QOS measures. Reliability has many facets and encompasses more than reliable packet delivery.

Quality of service / reliability

The most important quality is reliability. In WSN reliability has many facets.

(i) Detection reliability: Here the question is whether the events is the n/wk is supposed to detect is actually detected or not. The main requirement for this is possible event locations are covered by sensing ranges of number of sensors. This leads to a coverage problem.

(ii) To have information accuracy: To attain this by taking multiple readings obtained at different time & space is considered to smooth out noise and detect outliers.

(iii) Reliable data transport: Considering that an event has been accurately detected this information is reported reliably from event location to sink nodes that are several hops away. Conversely, application like distribution of application code to the sensors require reliable data delivery from sink nodes to individual or groups of sensors. Hence, we have reliable data transport Problem.

Sense to sink (reliable data transport)
Sink to sensor (Distribution of application code)

(Means reliable data transport from sensors to sink node as well as transport of application code from sink to sensors.

Transport Protocols

Tasks attributed to transport Protocols are:

(i) Reliable data transport: This task requires ability to detect & repair losses of packet in multihop wireless networks.

(ii) Flow control: The receiver may temporarily unable to process incoming packets due to low memory. Flow control is used to solve this by making slow transmission of packets from transmitter

(iii) Congestion control: Congestion occurs when more packets are created than the n/wk can carry & the n/wk then drop packets. Packet drop causes energy waste. congestion control avoid this or react to it. Congestion is avoided by controlling the rate at which sensor nodes generates packets.

(iv) Network abstraction: Transport layer offers a programming interface to applications to shield the applications from many complexities.

Some challanges for transport Protocols in WSNs are:

→ WSNs are multihop n/wk of homogenous nodes, which is not an easy environment for TCP used for other wireless channels.

→ A transport protocol associated with energy constraints, memory constraints or computational constraints of sensor nodes.

→ Transport protocols are faced with variable topologies.

Coverage and deployment

In WSN the major task is surveillance of geographical areas, detect intruder, wildfire etc. for all such events to have accurate detection the sensors must be close enough to actual event. Two questions are:

→ How many points are covered by the sensors, i.e., coverage is an important aspect of QoS in WSNs.

→ For certain area and coverage requirements, how many sensors needed & where to be placed. This question is labeled as the deployment problem.

(Hence coverage & deployment have second important implicat? beside QoS. However, in some applications like wildfire detection, habitat monitoring with large number of sensors the deployment is loosely controlled process. Therefore large focus is given to coverage problem.

Sensing models: (Quality, directionality, variability o/p) There are three basic aspects for sensing models such as quality of sensed signal which generally decays with increasing distance from the event location. Secondly the directionality of sensing quality. In practical scenario the sensor offers higher sensitivity in some directions (ideally sensitivity is same in all directions) therefore these directions are preferred.

Thirdly, the possibility that some sensors generate different o/ps for same environmental stimulus at different time.

→ In general, the signal delivered by sensor for an external event at a distance is not fixed rather a distance dependent random variable (RV). This RV is model as a constant plus some zero mean Gaussian noise with either constant or distance dependent variance.

Considering two omnidirectional sensing model (without randomness)

a) Boolean Sensing model:

In this model all sensors of the same sensor modality (temperature, Humidity, ...) have a common sensing range r. The events with in this range are detected & event outside these are not detected. Hence, the sensor o/p signal for a sensor node at position p observing an event at q has strength:

α: const sensor value

$\|\cdot\|_2 \rightarrow$ Euclidean dist.

$$S(p,q) = \begin{cases} \alpha & : \|p-q\|_2 \leq r \\ 0 & : \text{otherwise} \end{cases}$$

(b) General Sensing model

Here also sensor possesses certain maximal sensing range but with in that range the sensor o/p obeys a power law instead of being uniform

$$S(p,q) = \begin{cases} \dfrac{\alpha}{\|p-q\|_2^{\beta}} & : r_0 \leq \|p-q\|_2 \leq r \\ 0 & : \text{otherwise} \end{cases}$$

r_0: certain min distance & β is +ve real number depend on sensing modality & sensor technology.

Coverage measures:

Coverage measures refer to sensor n/wk deployed to monitor some specified terrain A with area $|A|$. In many practical scenario the actual deployments underlying stochastic process as well as parameters are unknown. If protocols & algorithms are available to do they areas with reduced coverage can be identified and additional sensors can be deployed there. Some techniques are

✓→ Determining K-coverage (Coverage area is K times of α (const. sensor value)

✓→ Determining worst-case coverage { It covers the path with largest possible distance to the sensor, hence it is a path through a n/wk that an intruder take to minimize the risk of detection.

Coverage of grid deployments

In such deployment, the entire area is divided into an array of squares of side length D and sensors are placed only at the centres of these squares.

Reliable data transport : (Due to reasons behind packet loss ← Transmission Error, N/w Congestion, Less Rx buffer.)

Reliable transport of data is essential because of three major sources of packet losses:

(i) The wireless channel introduces transmission errors, the packet of different node collide to lead to packet loss.

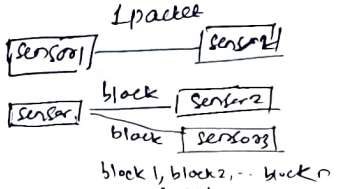
(ii) packets can be dropped on n/w due to congestion.

(iii) The receiver might drop packets if they arrive quickly.

The first issue is handled through error recovery.
The second is handled through congestion control
The third Problem is called the flow control.

Reliability requirements in sensor networks.

Several data transport task for WSN's are :



(i) Single packet versus block versus stream delivery:

There is a requirement for reliable transmission of data bet^n sensor nodes. In single packet case a single packet must be reliably transported between two nodes. In block ~~packet~~ delivery Problem, finite block comprising multiple packet must be delivered to a sensor or a set of sensors. In stream delivery problem an unbounded no. of packets has to be transported bet^n nodes.

(ii) Sink-to-sensors versus sensors-to-sink versus local-sensor to sensor

Most comm^n in sensor networks are not between arbitrary peer nodes but information flows either from sensor to single or few sink / gateway nodes or from sink to sensors

(iii) Guaranteed versus stochastic delivery:

In guaranted delivery it is expected that all transmitted Packet reach to destination. for example when a block code is distributed to a set of sensors, loss of any packet result in code block useless. Hence, guaranted delivery is challanging and costly interms of energy and bandwidth

energy & bandwidth expenditure.

→ The concept of stochastic delivery guarantees allows a limited amount of losses. For example the periodic data delivery with in every K subsequent packets at least m packets must reach the destination. Any number below m is a failure.

Single packet delivery

This problem is to deliver single packet from a source to sink node over multiple hops (because single hop requires trade-off betn achievable reliability and energy costs.) Therefore, it is more appropriate to consider stochastic guarantees & measure the reliability increase of packet delivery probability.

→ Depending the number of assumed o/p path betn. source & sink it can be a multipath solutions or single path solutions.

Using single path: — Single packet delivery (TX. detect loss)
Block or stream delivery (RX. detect loss)

Considering all single hop physical layer mechanism like FEC or variation of transmit power, the major objective is to improve packet delivery probability is through retransmissions and use of multiple paths.

→ The issues with retransmission are (i) who detect losses and what are the indicators used, (ii) who request retransmissions, and (iii) who carries the ~~information~~ the retransmissions.

→ In single packet delivery, the data packet can get lost. Hence, only the transmitting node has chance to detect this. Also it is the transmitting node who requests and perform retransmissions.

→ The transmitter is convinced about successful packet delivery after it receives +ve acknowledgement from receiver.

→ In case of block or stream delivery, it is possible to let the receiver detect losses (by checking holes in received sequence numbers) and request retransmission of missing packets by using -ve acknowledgement (NACK) packets. NACKs also carry implicit acknowledgements of other packets, then it is not necessary to send +ve acknowledgements of every packets. packets & saves energy.

For single hop delivery in WSN two standard approaches are using positive acknowledgements are:

1) MAC-layer retransmissions: When a node on path forward data packet, it expects to receive MAC layer acknowledgement. The transmitter tries for bounded number of trials for successful data packet forwarding then it drops the packet. However, these acknowledgements leads to significant overhead, when small data packets are sent on exceptionally good channels.

2) End-to-end retransmissions: The source node needs to buffer the packet until an acknowledgement arrives from sink node. Also the number of trials made by source node is limited. End-to-end retransmissions can be can be combined with MAC-layer retransmission.

Using multiple paths:

Existence of multiple paths bet? a source & sink node can be exploited in different ways as:

(i) Providing alternative routes:

First multiple routes are set, then the preferred route is choosen from these multiple routes and then switched to another route when preferred route fails. These paths can be ~~@~~ braided paths.

→ A good route is beneficial at first place that is energetically feasible routes that have low per hop error rates. Estimation of per-hop error rates can be done from physical layer attributes like ~~(illegible)~~ signal strength or from counting acknowledgements and retransmission from MAC & link layer protocols

→ Instead of sending a single packet over one of these paths, there is option to send multiple packets over multiple paths. One such technique is ReInForM. ReInForM sends multiple copies of same packet.

ReInForM:

Here multiple copies of same packet is sent over multiple, random routes. Therefore, there is packet duplication both at source and ~~dest~~ ~~neighbor~~ intermediate nodes.

→ The number of duplicates to be created is decided by all nodes on the basis of local error rates, the hop distance to the sink node and required reliability.

Protcol work:

→ Each node i knows its hop distance n_i to sink and also knows its neighbors j and their respective hop distance n_j. Node i classifies its neighbors into three sets H_i^-, H_i^0 and H_i^+. H_i^- contains all neighbors j with $n_j = n_i - 1$, H_i^0 contains the neighbors j with $n_j = n_i$ and H_i^+ contains those for which $n_j = n_i + 1$. The node i also has estimate of local packet error rate e_i.

→ The operation starts at source s and goal is to deliver a packet with reliability γ_s. To decide the no. of paths P, the locally estimated error rate e_s and the assumptions of a BSC and independent path is used. Probability that a single packet fails over a path with n_s hops is $1-(1-e)^{n_s}$. If P packets are sent over P paths, the probability that none of the copies reach the sink is $(1-(1-e)^{n_s})^P$ and hence the probability that at least one packet reaches the sink is $1-(1-(1-e)^{n_s})^P$. This probability at minimum is γ_s and the required number of paths P can be obtained as

$$P = \frac{\log(1-\gamma_s)}{\log(1-(1-e)^{n_s})}$$

→ s's neighbor selection for starting path is done in way such that source select a distinguished next hop neighbor t from H_s^-, this is guaranteed in the existing work. The next of neighbor forward the packet unconditionally, all other neighbors do not. The probability that t receives it's copy & become forwarder is $1-e_s$. Hence, the no. of paths distributed over remaining neighbors of s is given as $P' = P-(1-e_s)$. These paths are allocated to the nodes H_s^-, H_s^0 and H_s^+ such that H_s^- has precedence over H_s^0, which inturn have precedence over H_s^+.

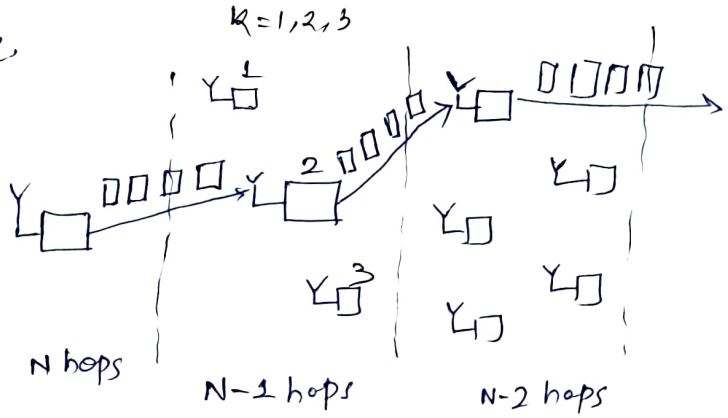
→ The source computes P_s^-, P_s^0, and P_s^+ determining the number of paths that shall be created from each of s's neighbor in the sets H_s^-, H_s^0, and H_s^+ respectively. Finally the source prepares a packet containing its hop distance n, its local error rate e_s, the next-hop neighbor t, and three values P_s^-, P_s^0, and P_s^-. Finally, this packet is broadcast.

→ When next hop node t receives the packet, it accept role of next Source and behaves in exactly same way the source has; t becomes the forwarder. Any node u receiving packet first determines the values P_u according to class to which P_u belongs, if $n_u = n_s - 1$, then $P_u = P_s^-$, Node u decides to become a forwarder when $P_u \geq 1$. If $P_u < 1$, node u becomes a forwarder only with Probability P_u. Hence, before forwarding a packet any forwarder u computes its local required reliability as

$$r_u = 1 - \left(1 - (1 - e_s)^{n_s - 1}\right)^{P_u}$$

(iii) HHB and HHBA

Hop-by-Hop Broadcast (HHB) and Hop-by-Hop Broadcast with Acknowledgment (HHBA) Protocols. In // (HHR) the source picks one neighbor and sends the same packet in unicast mode to this neighbor 'N' times, hoping that atleast one packet gets through. As shown in figure N=4 packet in each arc.



→ If source would broadcast the packets. It would suffice if any of k=3 neighbors having a n-1 hop distance to sink receives any of the packet copies correctly.

→ Using the mediums broadcast property, we can reduce the number N of copies generated in each hop, sending only $N' < N$ packets. With packet error rate of e_s, the delivery probability is given by $1 - e_s^{kN'}$. Given with required delivery probability r_s, this is solved for N' and we get $N' = N/k$. Hence the source sends less packets.

In HHBA, Protocol the source also sends N' packets but with larger spacing than with HHB. This larger spacing is sufficient to accomodate acknowledgement packets. (Specifically only a node that has decided to become a forwarder sends back an acknowledgement. After receiving the acknowledgement the source stops transmitting duplicates.) Here, it may happen that the source needs to send fewer than N' packets.

Multiple receiver: The approaches discussed previously targeted to deliver single packet to single receiver. The task to reliably transmit single packet to multiple receiver is need to be fully addressed (future). One problem is using positive acknowledgements lead to the ack implosion problem. (Can be dealt in future).

Summary:

→ Pure end-to-end recovery is advantageous when single hop shows no errors. For handling real life error rate, local error recovery based on MAC layer acknowledgements is advisable. MAC-layer can also support selection of feasible routes.

→ Higher desired delivery probability or larger number of hops leads to increased energy consumption.

→ Flooding scheme achieve excellent delivery probability with high price.

→ Positive acknowledgement ensures higher reliability but has problems as (i) they are sent even if channel is good, and (ii) there is ack implosion problem when the packet is to be delivered to multiple receivers.

Block delivery: (Use stochastic delivery for segments/packets.)

For delivery of large data items such as transportation of time series data from sensors to sink or when sink sends new application code or new user queries to retask the n/w block delivery is used.

→ Here the block data is split into multiple packets/segments/fragments to reduce the packet error rates and deal with packet size limitations of transceiver.

→ Important feature of such block transfer is that NACKs can be used. NACKs potentially reduces the number of acknowledgement packets.

→ NACK provides a retransmission request send by receiver. such a request when intermediate nodes cache the segments, they serve ~~as~~ but with benefit that the NACK and the retransmitted segment don't need to travel whole distance betn source & sink. This node is called a (recovery sensor) Hence, all nodes in the n/wk spend some buffer for caching. The schemes that incorporate such ideas are:

(i) PSFQ: block delivery in the sink-to-sensor case: (Uses broadcast operation pump operation: downstream node fetch operation: upstream nodes)

The Pump Slowly Fetch Quickly (PSFQ) Protocol addresses the case where an ordered block of packets is to be delivered from one sink node (user terminal) to a set of sensor nodes. A major application of this protocol is distribution of new application or protocol code for retasking of the n/wk. Hence, the sensor node has to receive entire code block before working on it without any considerable loss.

→ In PSFQ Protocol, data source pumps the packets making up the code block one after other into n/wk using a large period and a broadcast or directed broadcast method. Nodes after receiving these packets store them in an internal buffer in a sequence and forward them to downstream nodes.

→ The intermediate node if receives an out of sequence segment doesn't forward it immediately but quickly requests the missing ~~packets~~ segments from upstream neighbors. This entire operation is fetch operation & corresponds to a NACK. Between two pumps multiple fetch trials are made, therefore Pumping is slow & fetching is quick. As soon as missing packets arrive the intermediate node continues to pump packets in sequence into the n/wk. using a broadcast operation. This protocol assumes that the losses are entirely due to channel errors and not due to congestion and PFSQ contains no mechanism to deal with congestion.

PFSQ also handles:

(a) Behavior of the data source: The code block is split into series of packets or segments. Each packet (inject message) contains four additional fields as file id (identifies the ~~block~~ code blocks), file length (length of code blocks, sequence number (identifies particular packets within code block),

and Time to Live (TTL). The data source broadcasts packet one after other with spacing T_{max} secards. T_{max} must be large enough to accomodate sufficient fetch operations. With larger time interval intermediate and end nodes have sufficient time to process the cicaccing Segments. Third there must be sufficient time for downstream nodes to repcamp the segments.

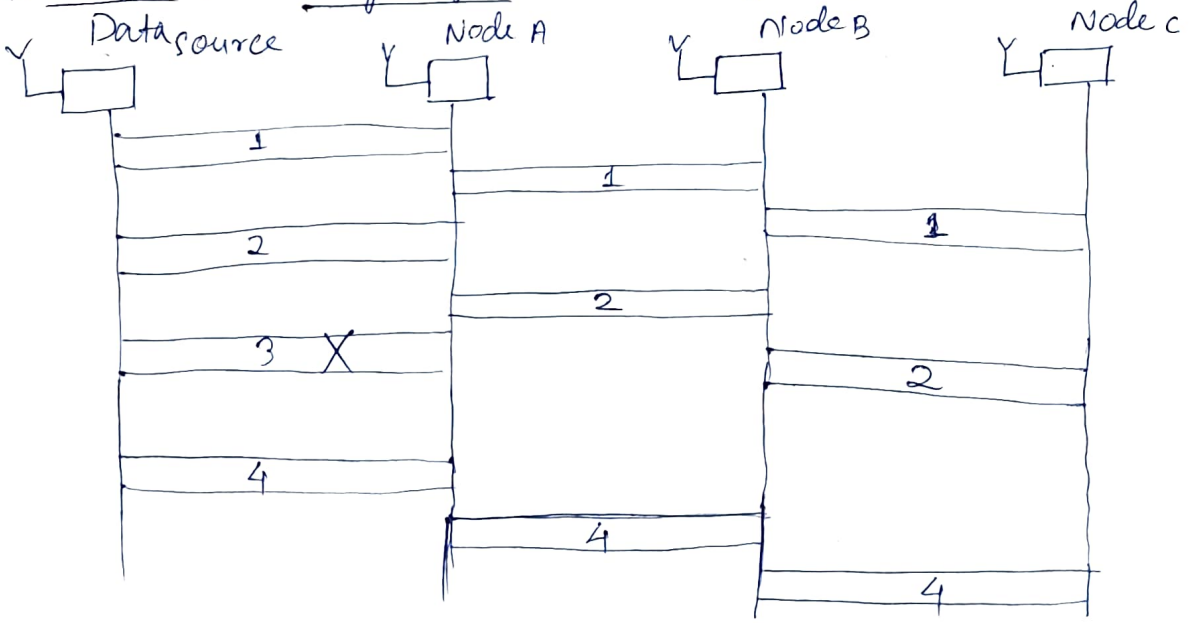
(b) Handling duplicate Packets: The intermediate node checks the duplicacy of the receiving segment/packet by checking the file id & segment number in the cache. If packet is found it is dropped. If packet has not yet received, it's TTL field is decremented. If the field is zero, the node stops forwarding the packet. Otherwise it starts forwarding process by looking at the sequence number of packet.

(c) Handling in sequence Packet:

A new packet is received in sequence when all packet of same code block ~~[illegible]~~ with lower sequence number have been received. Once the packet is received it is scheduled for repumping to downstream nodes.

(d) Handling out-of-order packet (Repumping takes all segments again from sink)

Receiving out-of-order packet is more complex. General strategy followed by PFSQ is not repump the packet, rather node tries to request (fetch) the missing segments as quickly as possible.



Supression of Pcamp operation for time being is described through above figure. The loss of Packet 3 triggers fetch operation at node A as soon as A receives packet 4. When packet 4 is forwarded to B and C it detect the same loss event and themselves start fetch for packet 3. Hence, three node start fetch operation for single loss event. Therefore, the fetch operation at B & C are useless as long as A doesn't have the missing segment. Therefore, Supressing Pcamp operation Prevents loss Propagation.

(e) Proactive fetch: Here node A sets timer to a value T_{pro} each time it receives a new Packet. After expiry of timer node A enters the fetch mode and requests all missing Packets from the upstream neighbors.

(f) Report operation: The PFSQ Protocol also specifies a report facility that allocas data source to have access of the nodes that have completely received code block. The sink node request reporting by setting a reserved bit in TTL field of injected message.

RMST: block delivery in the Sensors-to-Sink Case

→ For guaranteed delivery of large blocks of data from sensors to sink the protocol Reliable Multiagent Transport (RMST) is used.

→ Large data blocks is fragmented by source nodes into number of fragments before transmission. The sink node deliver the whole block as soon as it receive all fragments. RMST is not designed for explicit congestion control, to guarantee in sequence delivery to sink, or to obey any time bounds.

Design of RMST

Design of RMST combines repair mechanisms on different layers.

(i) It exploits MAC-layer retransmission to increase chance of data Packets to make it over single hop. For unicast frames (data frames) the MAC layer acknowledgements and virtual carrier sense operation is used

However, for broadcast operation (in interest dissemination), none of the MAC-layer acknowledgement and virtual carrier sense is used.

(ii) In cache mode all nodes in n/wk and those in reinforced path between Source and Sink cache the fragments and check missing ones. Intermediate nodes in cache mode Periodically check missing fragments. After detecting missing fragments, a NACK packet indicating the missig fragments is sent back as MAC layer unicast packet. (Intermediate node give NACK)

(iii) In a non cached mode the intermediate nodes maintain no caches, only the Subscribing Sink has to collect fragments. Here the Sink is entirely responsible to detect losses and issue NACK Packets. (Sink Send NACK)

(iv) On Application layer, Source regularly Send out all fragments consisting the block and continues until Sink explicitely unsubscribes.

(v) The diffusion routing mechanism keeps track of node failures and choosing good routes: the Source node regularly send messages, which leads to establishment of a new reinforced path

RMST use directed diffusion which consists of additional attributes as

RMSTNO attribute: Here a data block is uniquely identified.

FRAGNO attribute: fragment within a data block is identified through this.

MAXFRAG attribute: Total number of fragments making up data block is given by this attribute.

CODA congestion control framework (Sink to Source)

The congestion Detection and Avoidance (CODA) combines a congestion detection mechanism with two congestion control mechanism working on different time scales. This mechanism considers two congestion scenarios. One is open-loop-hop-by-hop backpressure mechanism and closed-loop regulation mechanism.

Open-loop hop-by-hop backpressure mechanism

Here when a node detects congestion, it regularly (broadcasts) backpressure messages as long as congestion is present. Also some application dependent policy like dropping some of the packets or reducing rate of measurement at node etc.

→ A node B receiving a backpressure message from A has many options. For example B can start to drop packets or reduce rate. The node B can also forward backpressure message further to the data source.

→ If node B is also congested, it increment a counter in backpressure message before forwarding it. When the message reaches noncongested node the size of congested area is infered from counter value.

Closed loop regulation mechanism

(more reception of data at sink, or at any node in n/w causes hotspot leads to congestion)

A sink node receives data from multiple sources, so a hotspot is created close to sink or in other regions of n/w. Hence, a mechanism is required to reduce packet generation rates. If nodes receive no backpressure message they continue packet forwarding and which eventually leads to congestion state. Furthermore, the backpressure message need to travel several hops from sink to source nodes that causes more n/w load.

→ Closed loop mechanism is requested by sources when packet generation rate exceeds a given fraction γ of locally available channel capacity. (to this source a (specific bit) in data packets, that trigger acknowledgement packets in sink.)

→ A source that requests acknowledgements receive a min^m no. of acknowledgements ~~when it dicsonises~~ over certain period.