

Transport Protocols in Wireless Sensor Networks

Mohammad Hossein Yaghmaee

Associate Professor

Department of Computer Engineering
Ferdowsi University of Mashhad (FUM)

Transport Protocol Objectives

- Transport protocols are used to :
 - Mitigate congestion,
 - Reduce packet loss,
 - Provide fairness in bandwidth allocation,
 - Guarantee end-to-end reliability.
- The traditional transport protocols (i.e., UDP and TCP) cannot be directly implemented for WSNs
 - UDP does not provide delivery reliability that is often needed for many sensor applications,
 - Nor does it offer flow and congestion control that can lead to packet loss and unnecessary energy consumption.

Drawbacks of TCP in WSNs

- Overhead associated with TCP connection establishment
- Flow and congestion control mechanisms in TCP
 - Result in unfair bandwidth allocation and data collections.
 - TCP assumes that packet loss is due to congestion
 - TCP has a degraded throughput in wireless systems
- TCP relies on end-to-end retransmission
 - Consumes more energy and bandwidth than hop-by-hop retransmission.
- TCP guarantees successful transmission of packets
 - Is not always necessary for event-driven applications in sensor networks.

Design Guidelines

- In WSNs several new factors, can result in congestion:
 - Convergent nature of upstream traffic
 - Limited wireless bandwidth
- Two reasons of packet loss in WSNs:
 - Packet loss due to congestion in intermediate nodes
 - Packet loss due to bit-error rate in wireless channel
- Two major problems that WSN transport protocols need to cope with:
 - Congestion
 - Packet loss.

Performance Metrics

- Transport protocols for WSNs should provide:
 - End-to-end reliability
 - End-to-end QoS
- Performance metrics :
 - Energy efficiency,
 - Reliability,
 - QoS
 - Packet-loss ratio,
 - Packet-delivery
 - Latency
 - Fairness.

Performance Metrics :

Energy Efficiency

- Sensor nodes have limited energy.
- Transport protocols should maintain high energy efficiency
 - To maximize system lifetime.
- For loss-sensitive applications,
 - Packet loss leads to retransmission
 - Inevitable consumption of additional battery power
- Therefore, several factors need to be carefully considered,
 - Number of packet retransmissions,
 - Distance (e.g., hop) for each retransmission,
 - Overhead associated with control messages.

Performance Metrics :

Reliability

- Reliability in WSNs can be classified into the following categories:
 - Packet reliability:
 - Applications are loss-sensitive and require successful transmission of all packets or at a certain success ratio.
 - Event reliability:
 - Applications require only successful event detection, but not successful transmission of all packets.
 - Destination-related reliability:
 - Messages might need to be delivered to sensor nodes:
 - That are in a specific subarea
 - That are equipped with a particular sensor type.

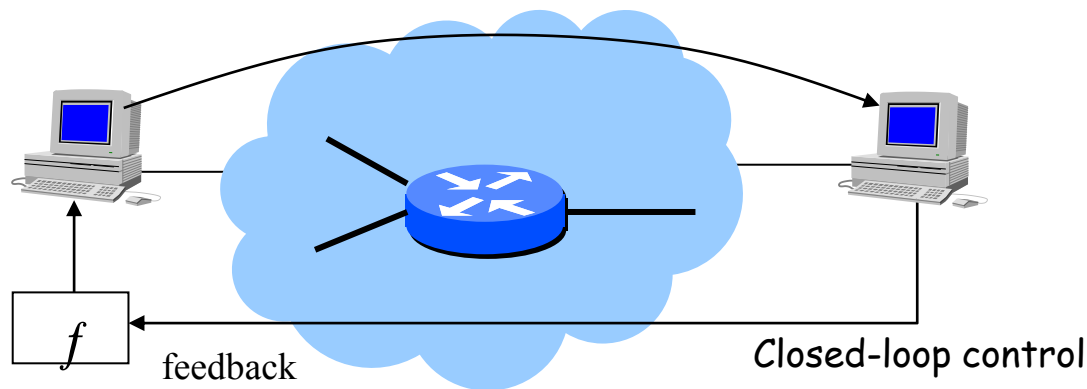
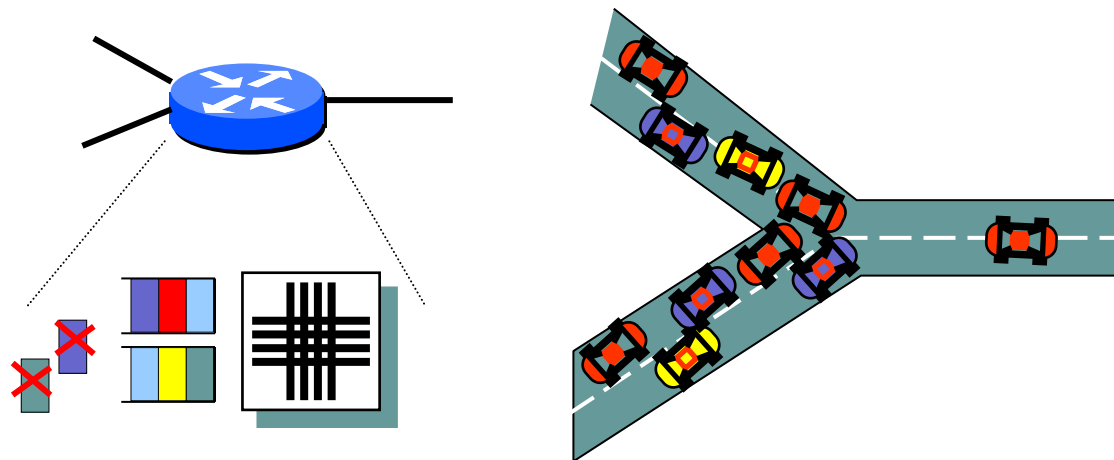
Performance Metrics: QoS Metrics

- QoS metrics include:
 - Bandwidth,
 - Latency or delay,
 - Packet-loss ratio.
- Depending on the application, these metrics or their variants could be used for WSNs.
 - Target tracking Application:
 - Generate high-speed data streams
 - Require higher bandwidth
 - For a delay-sensitive application:
 - May also require timely delivery data.

Performance Metrics: Fairness

- Sensor nodes are scattered in a geographical area.
- **Many-to-one** convergent nature of upstream traffic:
 - It is difficult for sensor nodes that are far away from the sink to transmit data.
- Transport protocols need to allocate bandwidth fairly among all sensor nodes
 - Sink can obtain a fair amount of data from all the sensor nodes.

Congestion Control



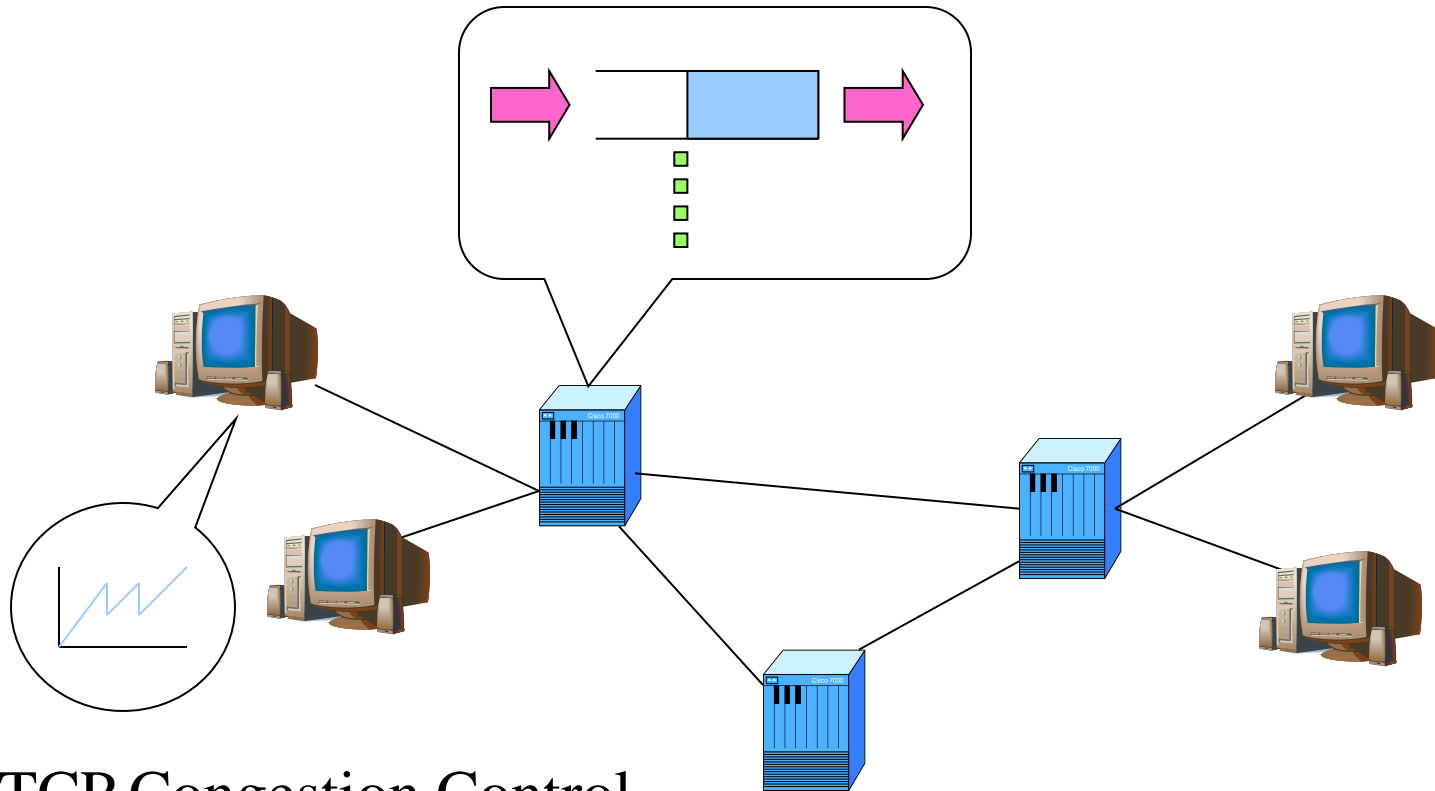
Feedback should be frequent, but not too much otherwise there will be oscillations
Can not control the behavior with a time granularity less than the feedback period

Effect of Congestion

- Packet loss
- Retransmission
- Reduced throughput
- Congestion collapse due to
 - Unnecessarily retransmitted packets
 - Undelivered or unusable packets

Congestion Control in the Internet

Active Queue Management (AQM)

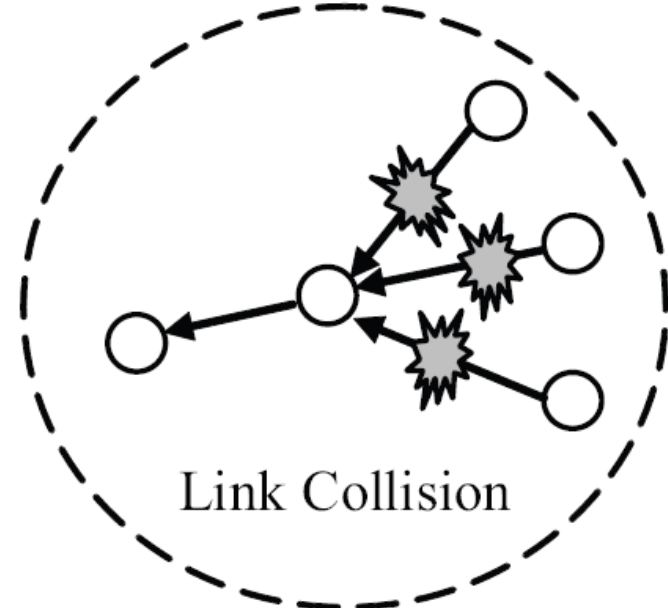
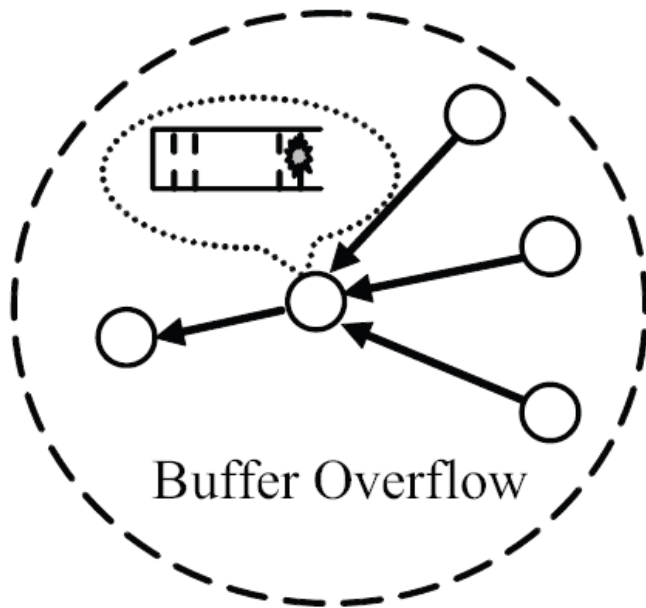


TCP Congestion Control

Causes For Congestion in WSNs

- Due to the packet-arrival rate exceeding the packet-service rate.
 - This is more likely to occur at sensor nodes close to the sink
- Link-level performance aspects such as:
 - Contention,
 - Interference,
 - Bit-error rate.
 - This type of congestion occurs on the link.

Types of Congestion in WSNs



- **Node Level Congestion:**

- It is due to the packet-arrival rate exceeding the packet-service rate.
- This is more likely to occur at sensor nodes close to the sink.

- **Link Level Congestion:**

- It aspects such as contention, interference, and bit-error rate.

Effects of Congestion in WSNs

- Energy efficiency:
 - Waste the limited node energy
- Application QoS:
 - Degrade reliability and application QoS
- Buffer overflow
 - Larger queuing delays
 - Higher packet loss.
- Degrade link utilization.
- It results in transmission collisions if CSMA, is used
 - increases packet-service time
 - wastes energy.

Congestion Control Approaches

- There are two general approaches to control congestion:
 - **Network resource management** :
 - tries to increase network resource to mitigate congestion
 - **Traffic control**:
 - implies to control congestion through adjusting traffic rate at source nodes or intermediates nodes

Traffic Control Methods

- **End-to-end** :
 - Can impose exact rate adjustment at each source node
 - Simplify the design at intermediate nodes
 - It results in slow response and relies highly on the round-trip time (RTT).
- **Hop-by-hop** :
 - It has faster response.
 - Difficult to adjust the packet forwarding rate at intermediate nodes
 - Because packet forwarding rate is dependent on MAC protocol and could be variable.

Congestion Control Parts

- **Congestion detection**
 - Monitor buffer/queue size
 - Monitor channel busy time, estimate channel's load
 - Monitor the inter-packet arrival time (data, ctrl)
- **Congestion notification**
 - Explicit congestion notification in packet header, then broadcast (but then energy-consuming!)
- **Rate Adjustment**
 - Dynamic reporting rate depending on congestion level
 - In-network data reduction techniques (aggressive aggregation) on congestion

Congestion Detection

- In TCP:
 - Congestion is observed at the end nodes based on a **timeout** or **redundant Acknowledgments**.
- In WSNs:
 - Proactive methods are preferred.
- Congestion indicators:
 - Queue length
 - Packet service time
 - The ratio of packet service time over packet interarrival time

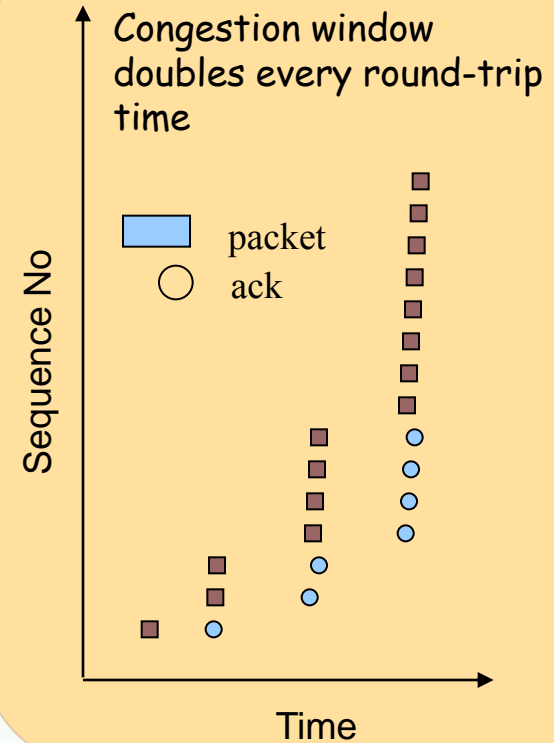
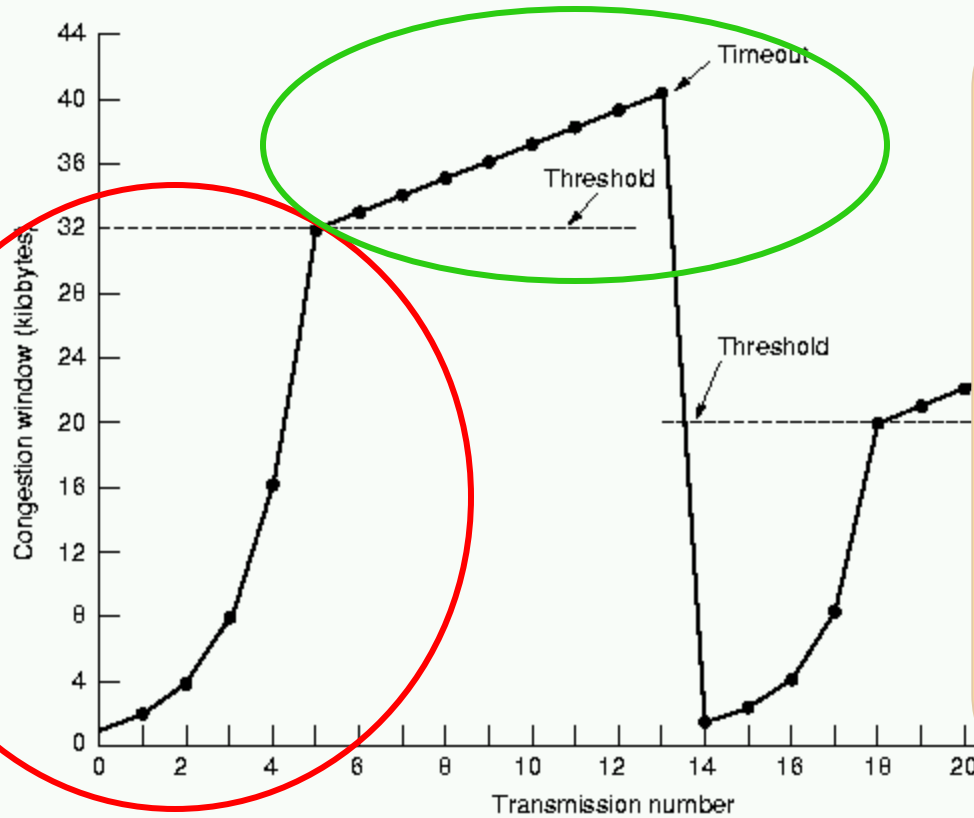
Congestion Notification

- Propagation of congestion information from the congested node
 - To the upstream sensor nodes
 - To the source nodes that contribute to congestion
- Congestion information
 - Congestion Notification (CN) bit,
 - Or more information such as allowable data rate, or the congestion degree.
- Disseminating congestion information:
 - **Explicit**
 - Uses special control messages to notify the involved sensor nodes of congestion
 - **Implicit**
 - Piggybacks congestion information in normal data packets.

Rate Adjustment

- Upon receiving a congestion indication, a sensor node can adjust its transmission rate.
- If a single CN bit is used:
 - Additive Increase Multiplicative Decrease (AIMD)
- If additional congestion information is available:
 - Accurate rate adjustment can be implemented

TCP Congestion Control



cwnd grows exponentially (**slow start**), then linearly (**congestion avoidance**) with 1 more segment per RTT

If loss, divides threshold by 2 (multiplicative decrease) and restart with cwnd=1 packet

Loss Recovery

- Reasons of packet loss in wireless environments:
 - Congestion
 - Bit error
 - node failure,
 - wrong or outdated routing information,
 - Energy depletion.
- How to overcome this problem:
 - Increase the source sending rate
 - Works well for guaranteeing event reliability
 - Is not energy efficient
 - Introduce retransmission-based loss recovery.
 - Is more active and energy efficient
 - Can be implemented at both the link and transport layers.
 - Link-layer loss recovery is hop-by-hop, while the transport layer recovery is usually done end-to-end.

Loss Detection and Notification

- A common mechanism is to include a *sequence number* in each packet header.
- The continuity of sequence numbers can be used to detect packet loss.
- Loss detection and notification can be:
 - *End-to-end*
 - *Hop-by-hop*.

End-to-End Approaches

- End-points (destination or source) are responsible for loss detection and notification.
- Drawbacks
 - Is **not energy efficient**.
 - The control messages would utilize a return path consisting of several hops
 - Control messages travel through multiple hops
 - Could be lost with a high probability due to either link error or congestion.
 - Leads to end-to-end retransmissions for loss recovery.

Hop-by-hop Loss Detection and Notification

- Intermediate nodes detect and notify packet loss.
- A pair of neighboring nodes are responsible for loss detection.
- Is more energy efficient.
- Two categories:
 - *Receiver-based*
 - Receiver infers packet loss when it observes out-of-sequence packet arrivals.
 - *Sender based*
 - Sender detects packet loss on either a timer-based or overhearing mechanism.

Methods to Notify the Sender

- Special control messages:
 - ACK (Acknowledgment)
 - NACK (Negative ACK)
- Piggybacking ACK in the packet header
 - IACK (Implicit ACK) using overhearing
 - Avoids control message overhead
 - More energy efficient.
 - Sensor nodes must have the capability to overhear the physical channel.
 - Is not feasible when:
 - Transmission is corrupt
 - Channel is not bidirectional
 - Sensor nodes access the physical channel using Time Division Multiple Access (TDMA)-based protocols

Retransmission-Based Loss Recovery

- End-to-end
 - The source performs retransmission.
- Hop-by hop.
 - An intermediate node that intercepts loss notification searches its local buffer.
 - If it finds a copy of the lost packet, it retransmits the packet.
 - Otherwise it relays loss information upstream to other intermediate nodes.
- The retransmission distance
 - The hop number between *cache point* and *loss point*

Comparisons

- End-to-end retransmission :
 - The cache point is the source node.
 - Has a longer retransmission distance
 - Allows for application-dependent variable reliability levels
- Hop-by-hop retransmission:
 - The cache point could be the predecessor node of the loss point.
 - Is more energy-efficient
 - Requires intermediate nodes to cache packets.
 - Is preferred if 100 percent packet reliability is required
 - Cannot assure message delivery in the presence of node failure

How Long Should a Cache Point Buffer?

- **End-to-end retransmission:**
 - The cache duration should be close to round-trip-time (RTT).
 - **NACK-based Acknowledgments:**
 - NACK messages could be lost or corrupted
 - Destination would be required to send NACK more than once.
 - Source nodes need to buffer a packet for a time duration which is longer than RTT.
- **Hop-by-hop retransmission:**
 - The cache duration is only influenced by the total local packet-service time and one-hop packet transmission time.

Issues Related to Hop-by-hop Retransmission

- Immediate retransmission
 - Retransmission can be triggered immediately upon the detection of a packet loss.
 - Results in shorter delay
 - If packet loss is caused by congestion it could aggravate the congestion situation and cause more packet losses.
- Distributed TCP Cache (DTC)
 - Given the limited memory in sensor nodes, packets may only need to be cached at selected nodes.
 - How to distribute cached packets among a set of nodes?
 - It balance the buffer constraints and retransmission efficiency by using probability-based selection for cache points.

Design Guidelines

- Several factors must be taken into consideration:
 - Topology
 - Diversity of applications
 - Traffic characteristics
 - Resource constraints
- Transport protocols components
 - Congestion control
 - Loss recovery
- Two approaches
 - Design separate protocols or algorithms, respectively, for congestion control and loss recovery.
 - Provides congestion and loss control in an integrated way
 - The joint use of these two protocols may provide the full functionality required by the transport protocols for WSNs.

The Existing Transport Protocols for WSNS

- **Protocols for Congestion Control**
 - Congestion Detection and Avoidance (CODA)
 - Control and Fairness (CCF)
 - Pump Slowly Fetch Quickly (PSFQ)
 - Priority-based Congestion Control Protocol (PCCP)
 - Siphon
 - Adaptive Rate Control (ARC)
 - Trickle
- **Protocols for Reliability**
 - Reliable Multi- Segment Transport (RMST)
 - Reliable Bursty Convergecast (RBC)
 - Event-to-Sink Reliable Transport (ESRT)
 - GARUDA
- **Protocols for Congestion Control and Reliability**
 - Sensor Transmission Control Protocol (STCP)

WSN Congestion Control Protocols

● STCP:

- Queue length,
- Implicit congestion notification,
- AIMD-like end-to-end rate adjustment

● Fusion:

- Queue length,
- Implicit congestion notification,
- Stop-and-start hop-by-hop rate adjustment

● CODA :

- Queue length and channel status,
- Explicit congestion notification ,
- AIMD-like end-to-end rate adjustment

● CCF :

- Packet service time
- Implicit
- Exact hop-by-hop rate adjustment

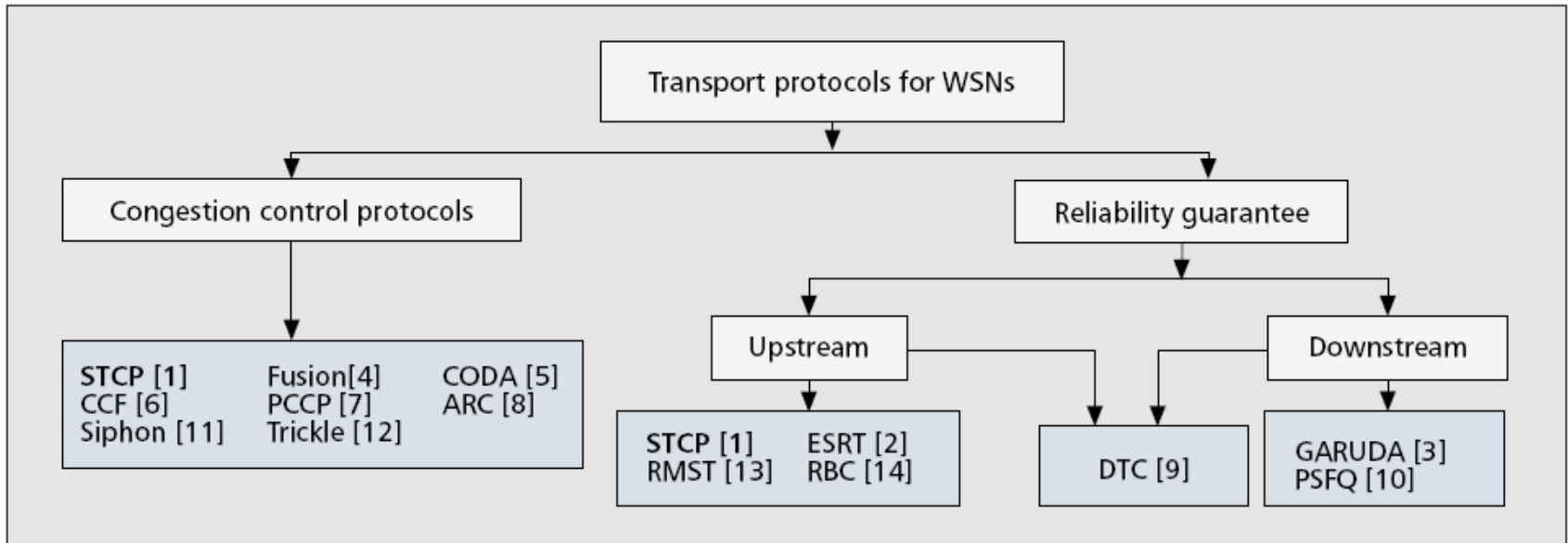
● PCCP :

- Packet interarrival time and packet service time,
- Implicit congestion notification,
- Exact hop-by-hop rate adjustment

● ARC :

- The event if the packets are successfully forwarded or not,
- Implicit congestion notification,
- AIMD-like hop-by-hop rate adjustment

Existing WSNs' Transport Protocols



PCCP: Priority Based Congestion Control Protocol

C. Wang, B. Li, K. Sohraby, M. Daneshmand, and Y. Hu,

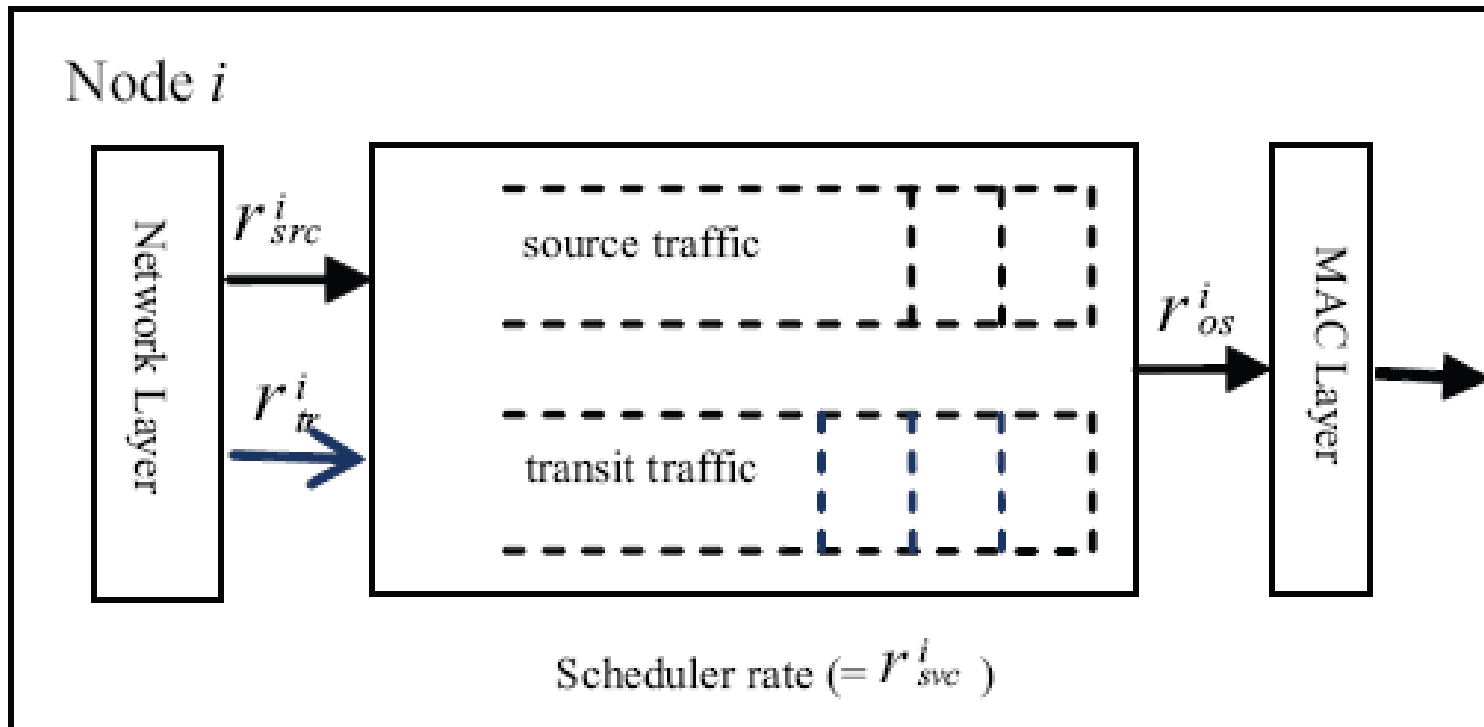
University of Arkansas

IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS,
VOL. 25, NO. 4, MAY 2007

PCCP: Priority Based Congestion Control Protocol

- PCCP detects congestion based on packet inter-arrival time and packet service time.
- PCCP uses implicit congestion notification:
 - To avoid transmission of additional control messages
 - To help improve energy-efficiency.
- PCCP designs a priority-base algorithm employed in each sensor node for rate adjustment:
 - The node with higher priority index gets more bandwidth and proportional to the priority index
 - The nodes with the same priority index get equal bandwidth.
 - A node with sufficient traffic gets more bandwidth than one that generates less traffic.

Queuing Model of PCCP



PCCP Protocol

- t_a^i mean packet inter-arrival *time* at sensor node i .
- t_s^i mean packet service times at the MAC layer of sensor node i
- Congestion degree: $d(i) = \frac{t_s^i}{t_a^i}$
- Updating:
$$t_a^i = (1 - w_a) \cdot t_a^i + w_a \cdot \frac{T_{N_p}}{N_p}$$
$$t_s^i = (1 - w_s) \cdot t_s^i + w_s \cdot t_s^i$$
- Both w_a and w_s are constant factors between 0 and 1 (usually are set to 0.1)

PCCP Rate Adjustment Algorithm

- Suppose p_i is the parent of node i .
- Periodically, the parent node measures its congestion degree and informs to all its childees.
- If new $d(p_i)$ is less than last $d(p_i)$ then there is no congestion in the parent node:
 - Child nodes Increase their scheduling and source rate as:

$$r_{svc}^i = \frac{r_{svc}^i}{d(p_i)} \quad r_{src}^i = r_{src}^i \cdot \frac{SP(i)}{GP(i)} \quad \begin{array}{l} GP(i) = \text{Global Priority of node } i. \\ GP(p_i) = \text{Global Priority of node } i\text{'s parent.} \end{array}$$

- Else there is congestion in the parent node:
 - Child nodes decrease their scheduling and source rate based on their priority

$$r_{svc}^i = base_rate \cdot \frac{GP(i)}{GP(p_i)} \quad r_{src}^i = r_{src}^i \cdot \frac{SP(i)}{GP(i)} = base_rate \cdot \frac{SP(i)}{GP(p_i)}$$

PCCP Rate Adjustment Algorithm

- **If no congestion detected**
 - Child nodes Increase their scheduling and source rate as:

$$r_{src}^i = \frac{r_{svc}^i}{d(p_i)} \cdot \frac{SP(i)}{GP(i)}$$

GP(i) = Global Priority of node i .

GP(p_i) = Global Priority of node i 's parent.

d(p_i) = Congestion degree in parent node

- **If congestion detected**
 - Child nodes decrease their scheduling and source rate based on their priority

$$r_{src}^i = base_rate \cdot \frac{SP(i)}{GP(p_i)}$$

PCCP Problem

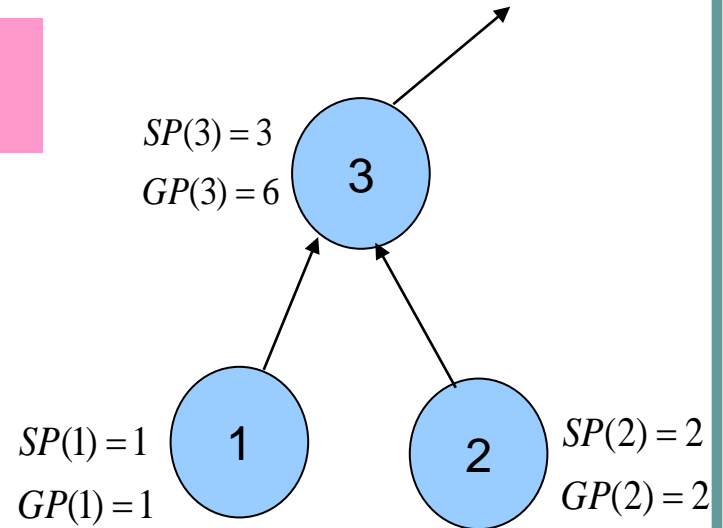
Suppose $d(3) = 0.5$ \rightarrow No Congestion



Both node 1 and node 2 increase their rates.

At node 1:
$$r_{src}^1 = \frac{r_{svc}^1}{d(p_3)} \cdot \frac{SP(1)}{GP(1)} = \frac{r_{svc}^1}{0.5}$$

At node 2:
$$r_{src}^2 = \frac{r_{svc}^2}{d(p_3)} \cdot \frac{SP(2)}{GP(2)} = \frac{r_{svc}^2}{0.5}$$



Although node 1 and node 2 have different priorities but they increase their scheduling rate and their source rate in a similar manner.

QCCP-PS: Queue Based Congestion Control Protocol with Priority Support

M.H.Yaghmaee and Donald Adjeroh

IEEE WoWMoM 2008

QCCP-PS: Queue based Congestion Control Protocol with Priority Support

- **Motivation**

- PCCP can't provide relative fairness in the case of decreasing congestion.
- QCCP-PS increases or decreases the source rate of each node directly based on its priority.

- **Characteristics**

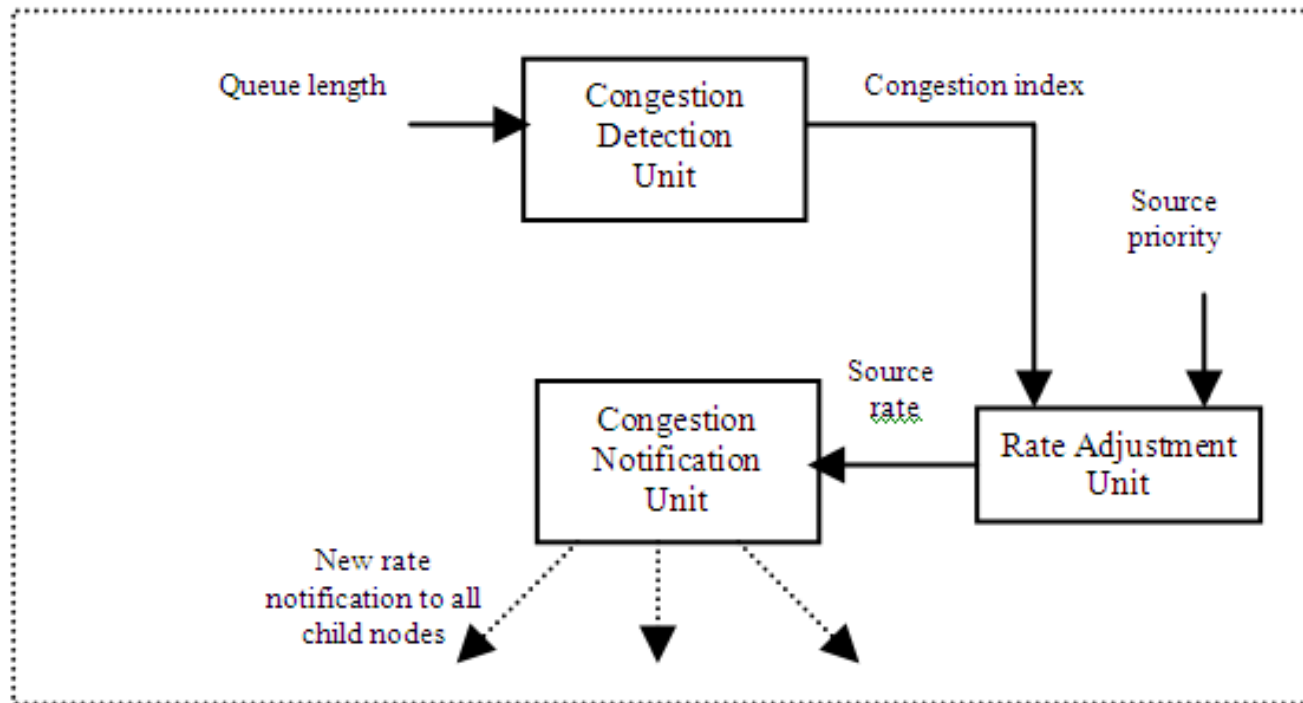
- Queue length congestion indicator
- Priority-base rate adjustment
- Implicit congestion notification
- Exact hop-by-hop rate adjustment

QCCP-PS Objectives

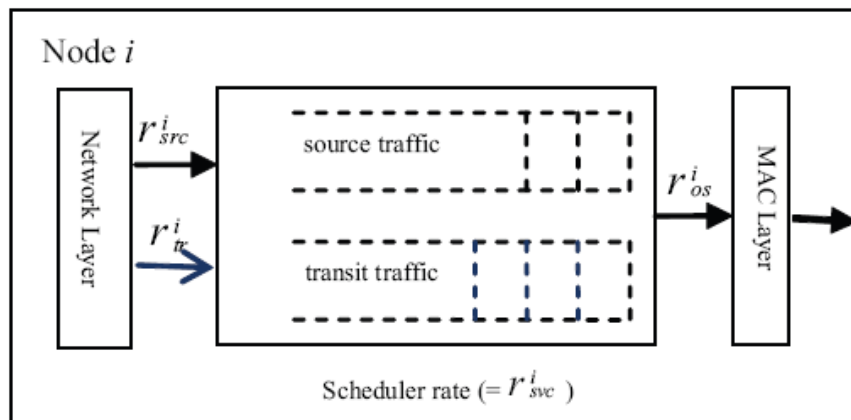
- To have better control on the sending rate of each child node.
- To provide better achieved priority and fairness for each sensor node based on its source priority.
- To prevent packet loss by controlling the queue length of each sensor node.

QCCP-PS Components

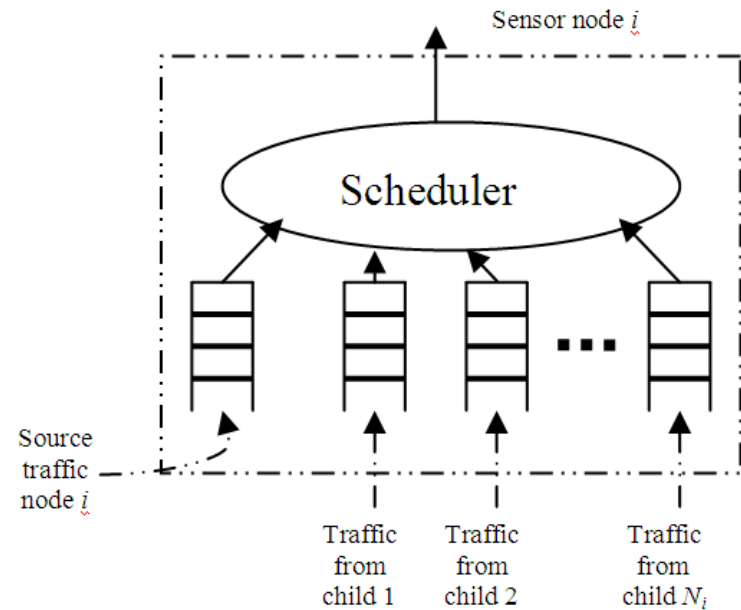
Sensor node



Queuing Model of PCCP and QCCP-PS



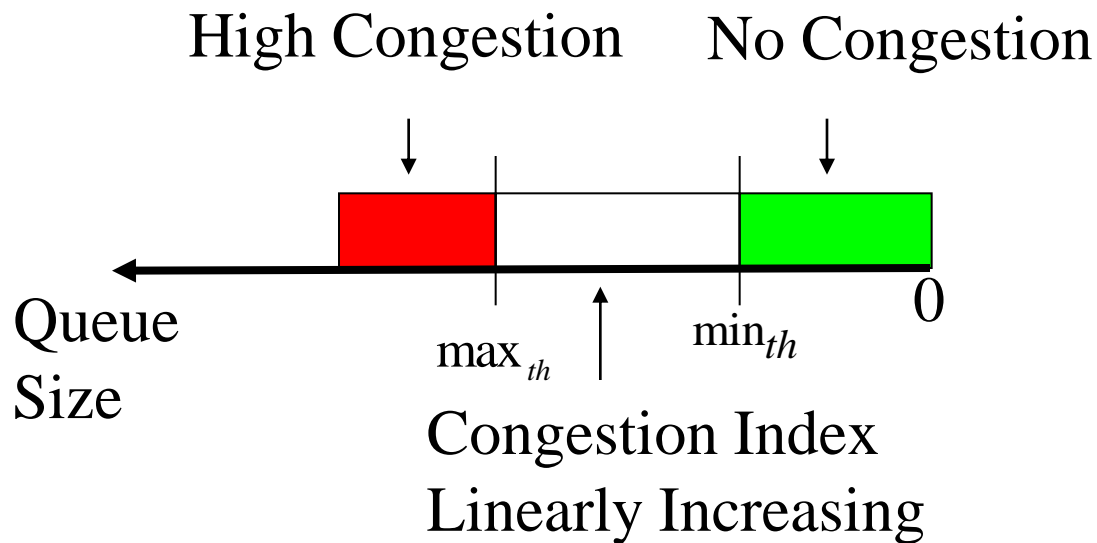
PCCP



QCCP-PS

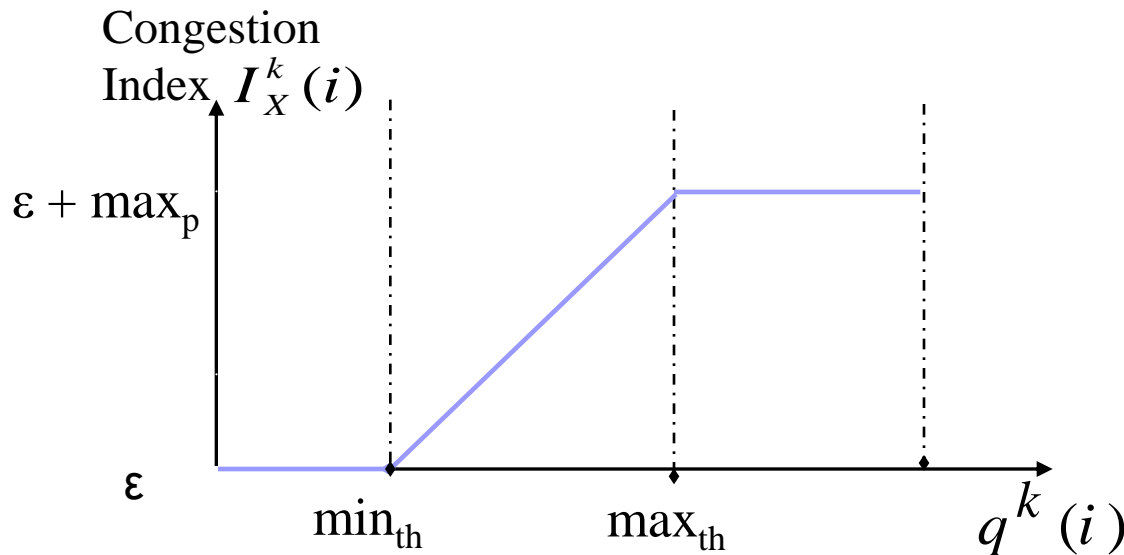
Congestion Detection

- Queue based congestion detection
- At each intermediate node, two different fixed thresholds \min_{th} and \max_{th} are defined.



Per Child Congestion Index

- Let $q^k(i)$ denotes the current queue size of the k -th queue in node i .
- *Two thresholds $\min^k_{th}(i)$ and $\max^k_{th}(i)$ are defined.*



Normalized Congestion Index

- Suppose that sensor node i has N_i child nodes. So it has $N_i + 1$ queues.
- For each queue k in sensor node i , the congestion index $I_X^k(i)$ is calculated, then $\bar{I}_X^k(i)$ is obtained as:

$$\bar{I}_X^k(i) = \frac{\sum_{j=1}^{N_i+1} I_X^j(i) - I_X^k(i)}{N_i \cdot \sum_{j=1}^{N_i+1} I_X^j(i)}$$

Note that $\sum_{k=1}^{N_i+1} \bar{I}_X^k(i) = 1$

Note when congestion is increased $I_X^k(i)$ is also increased while $\bar{I}_X^k(i)$ is decreased.

Source Priority

- Now, suppose each node i has a different priority.
- Let $SP(i)$ denote the source priority at sensor node i .
- We define the total priority, $TP(i)$ as the sum of priorities of all nodes in the subtree rooted at node i .

$$TP(i) = \sum_{j \in C(i)} TP(j) + SP(i)$$

where $C(i)$ is the set of node i 's child nodes

Example

The red nodes (B,C,D) belong to the subtree of green node (node A).

Suppose:

$$SP(A)=1$$

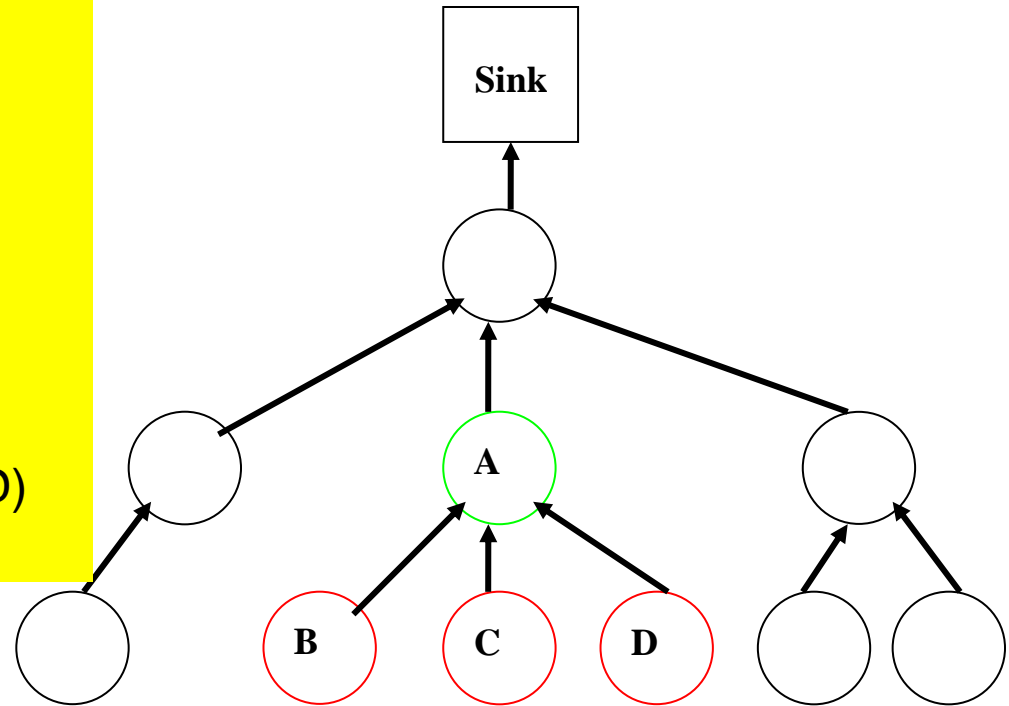
$$SP(B)=4$$

$$SP(C)=6$$

$$SP(D)=2$$

THEN:

$$TP(A)=SP(A)+SP(B)+SP(C)+SP(D) \\ =1+4+6+2=13$$



Per Child Rate Adjustment

- In each queue k in node i , the weight and the rate are calculated as:

$$w_i^k = \frac{p_i^k \bar{I}_X^k(i)}{\sum_{j=1}^{N_i+1} p_i^j \bar{I}_X^j(i)} \quad \left\{ \begin{array}{l} p_i^1 = \frac{SP(i)}{TP(i)} \\ p_i^k = \frac{TP(k)}{TP(i)}, k = 2, 3, \dots, N_i + 1 \end{array} \right.$$

Where r_i is the maximum output rate of node i

Note that: $\sum_{k=1}^{N_i+1} w_i^k = 1$

Rate Assignment Policy

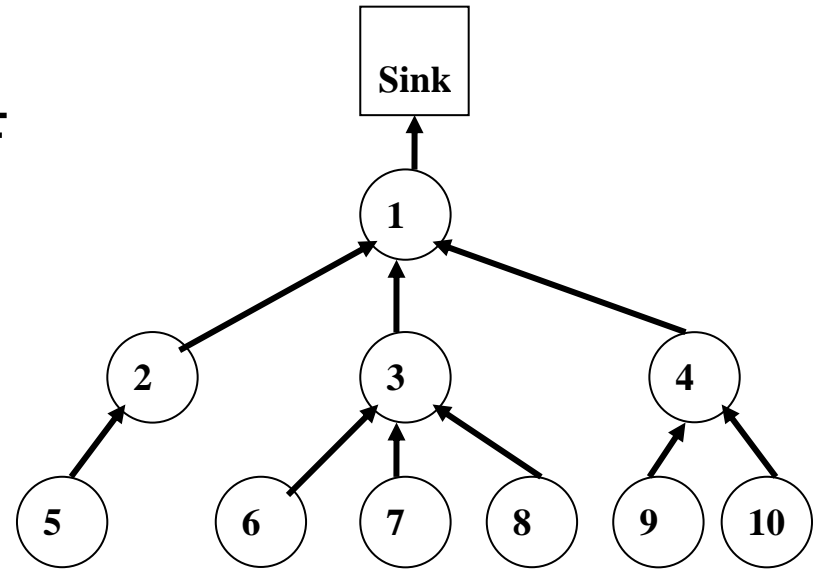
$$w_i^k = \frac{p_i^k \bar{I}_X^k(i)}{\sum_{j=1}^{N_i+1} p_i^j \bar{I}_X^j(i)}$$

$$r_i^k = w_i^k \cdot r_i$$

The child nodes with high priority and low congestion (high value of $\bar{I}_X^k(i)$) get more rate than the other nodes.

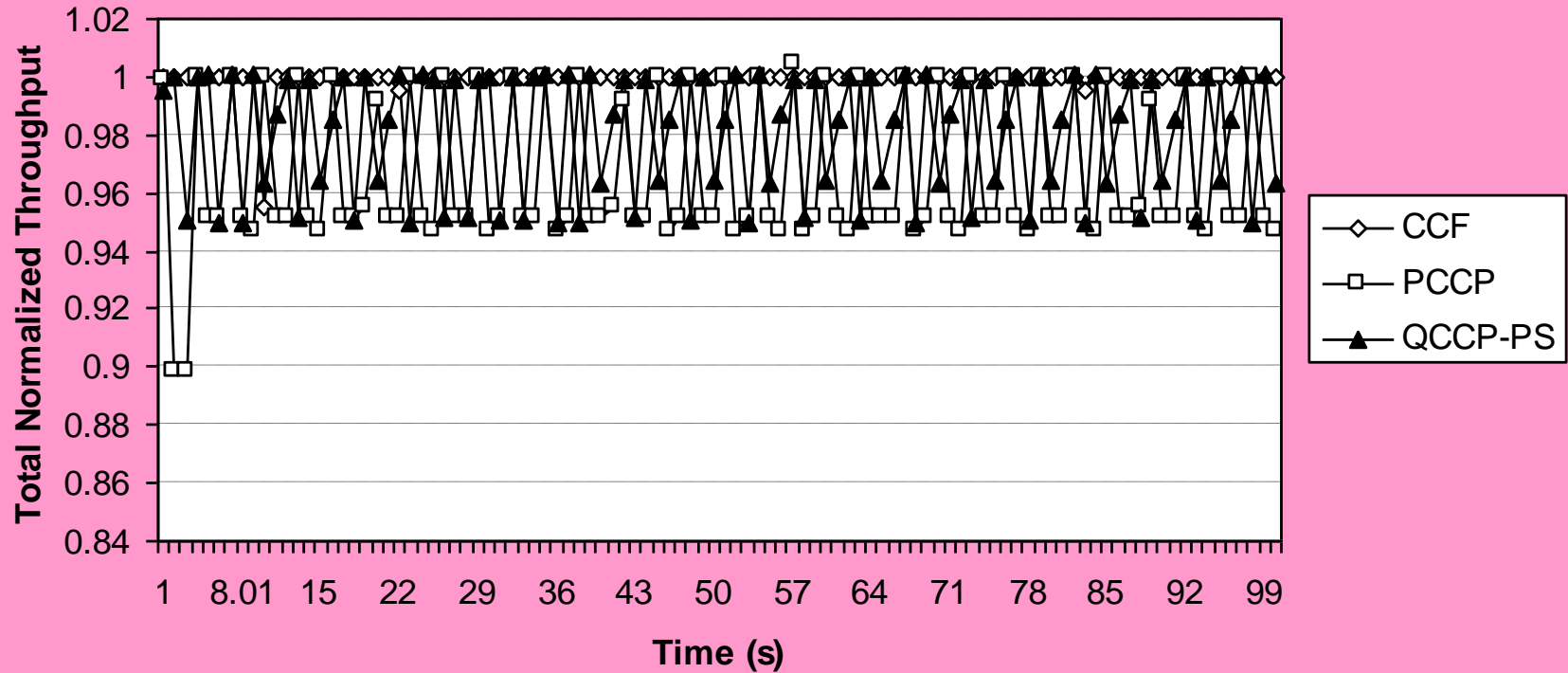
Simulation Results

- We compare the performance of QCCP-PS with that of PCCP and CCF protocols.
- A simulation software in C++ language were developed in UNIX environment.
- Three protocols were implemented:
 - CCF
 - PCCP
 - QCCP-PS



Simulation Topology

Utilization Performance



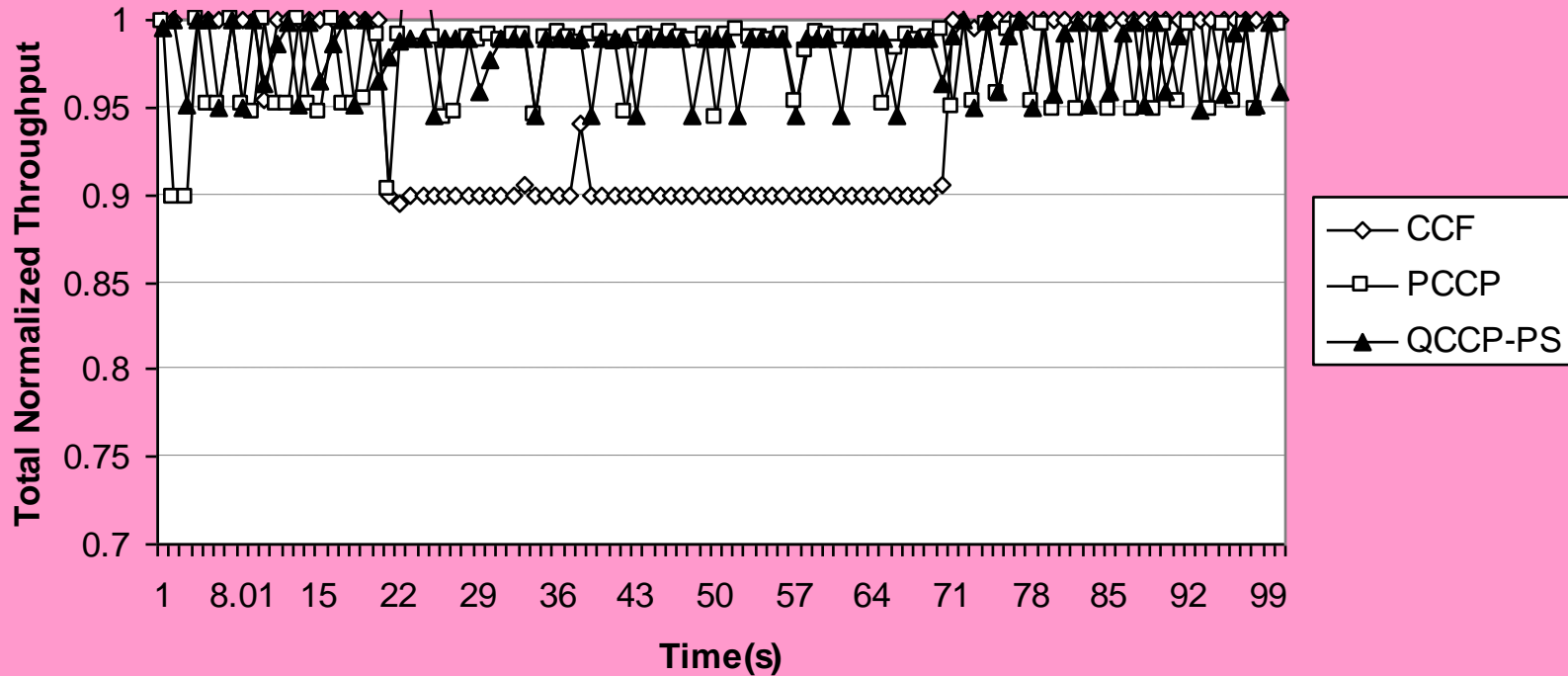
Average values

CCF: 0.99

QCCP-PS: 0.98

PCCP: 0.96

Dynamic Changes in Traffic Load

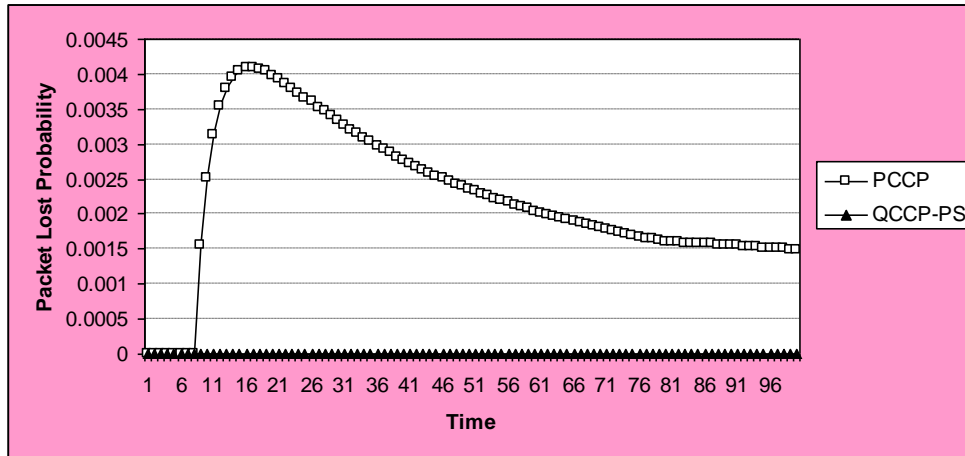


Sensor node 2, is off in time interval [20 sec,70 sec]

Average values

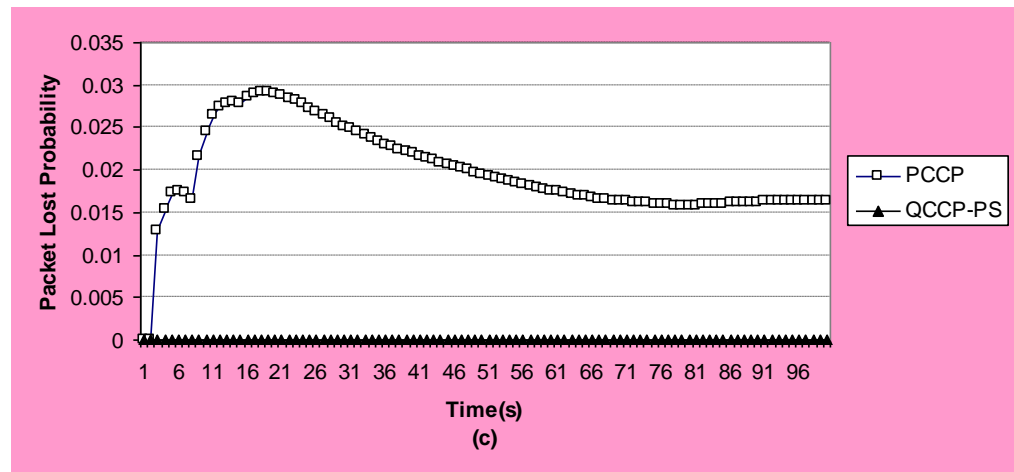
CCF: 0.95
QCCP-PS: 0.98
PCCP: 0.97

Packet Loss Performance



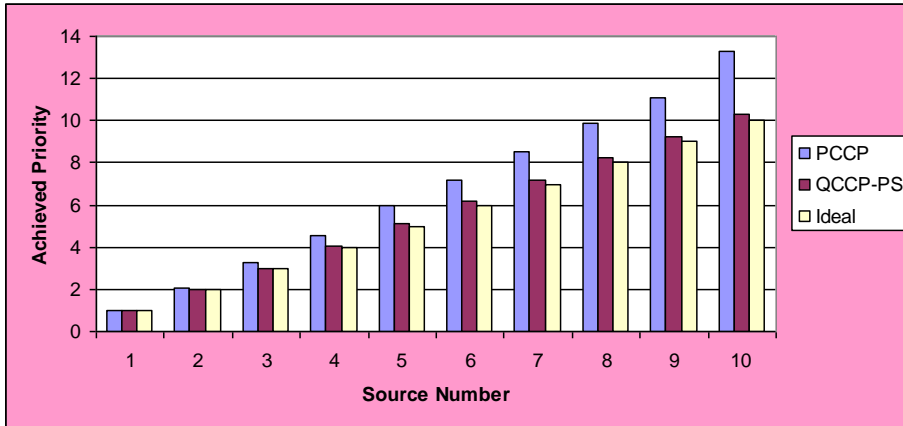
High Buffer Size (100 Packets)

Low Buffer Size (10 Packets)



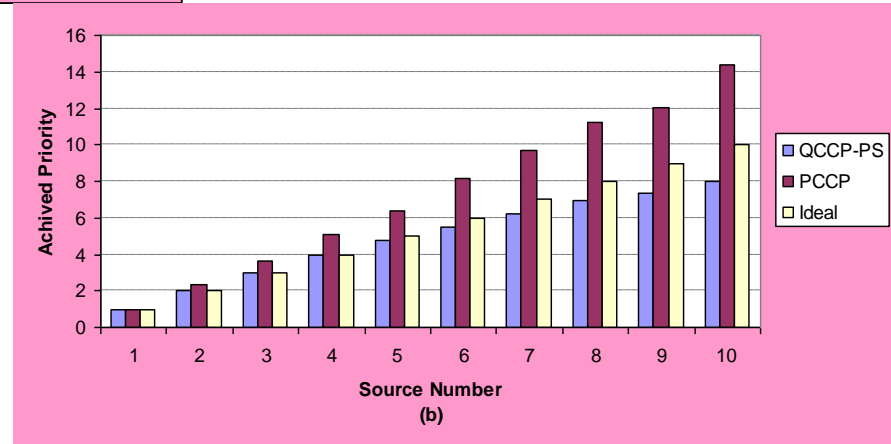
Achieved Priority

All sources have different priority



High Buffer Size (100 Packets)

Low Buffer Size (10 Packets)



Conclusion

- Priority based congestion control is an important issue in WMSNs.
- In this paper we proposed QCCP-PS which is a Queue based Congestion Control Protocol with Priority Support.
- QCCP-PS can provide achieve priority more better than PCCP.
- Simulation results show that the packet loss probability and queuing delay of QCCP-PS is less than PCCP which makes it suitable for multimedia applications.

RCRT: Rate-Controlled Reliable Transport for Wireless Sensor Networks

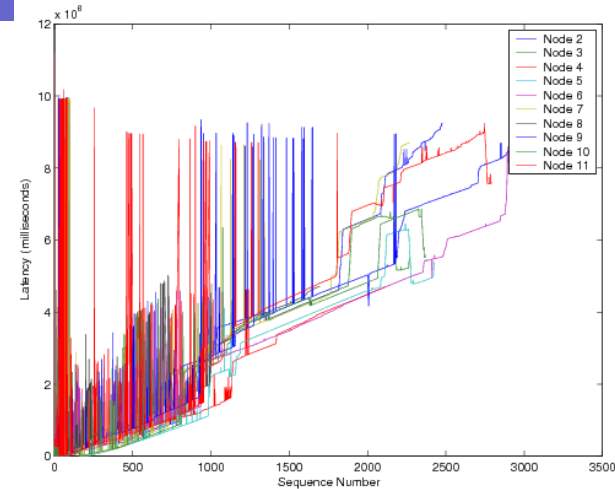
Jeongyeup Paek and Ramesh Govindan
University of Southern California

ACM SenSys'07

Outline

- Motivation
- Design goal
- RCRT Design
 - End-to-end Loss recovery
 - Congestion detection
 - Rate adaptation
 - Rate allocation
- Evaluation
- Conclusion

Motivation: A Wireless Sensor Network for Collecting Structural Vibrations



- Four seasons building deployment (Widén, 2004)
 - Nodes measured vibrations and transmitted it to a central node, over multiple hops
 - Preconfigured rates for each flow
- Led to congestion
 - More than an hour to receive 10 min of vibration data in a 15 node network

Question

Can we design a **protocol** that **reliably transports** sensor data from many sources to one or more sinks **without incurring congestion collapse**?

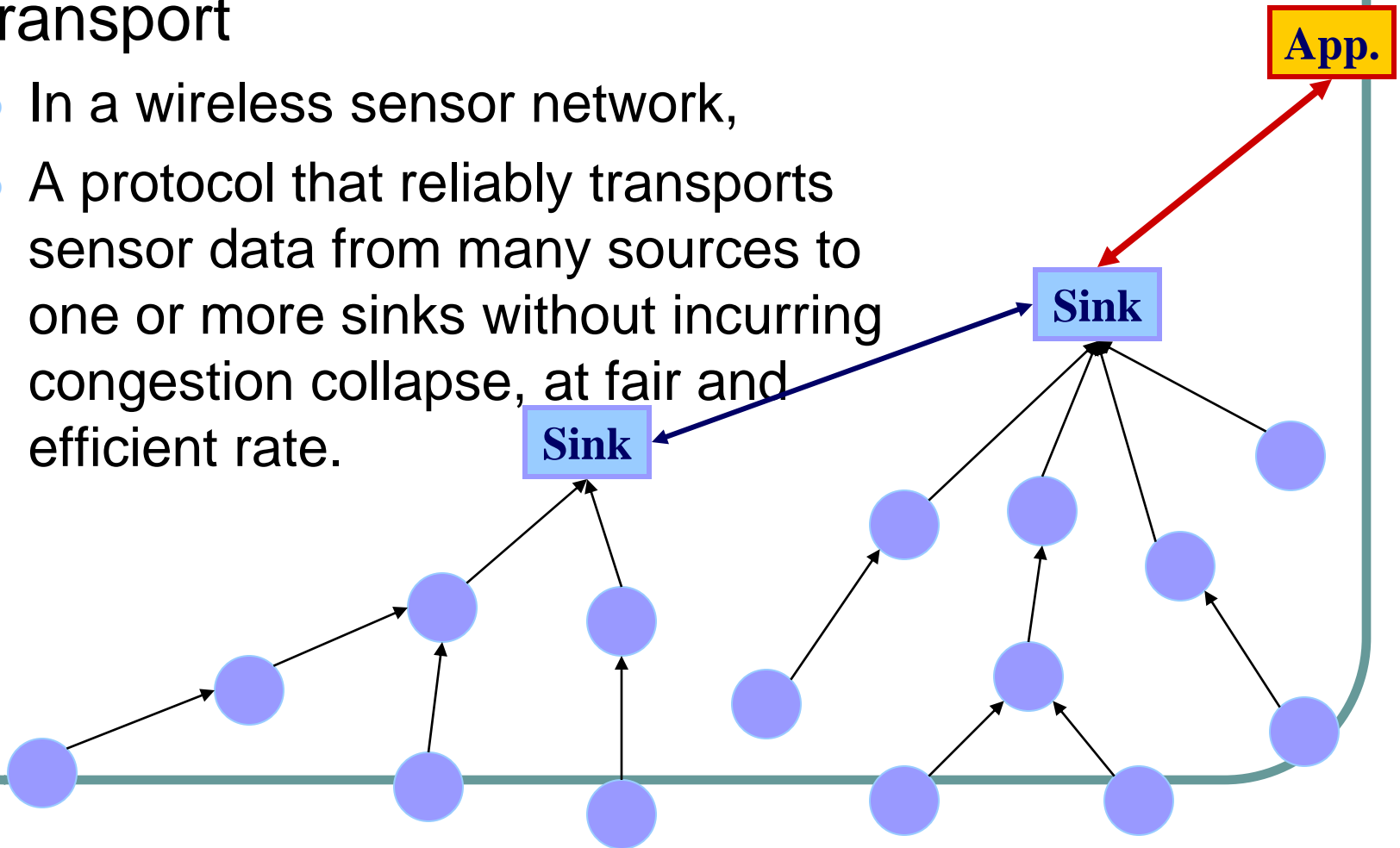
Design Goals

- Reliable end-to-end transmission
 - 100% packet delivery.
- Network efficiency
 - As high rate as possible without falling into congestion collapse
- Support for concurrent applications
- Flexibility
 - Allow different capacity allocation policies.
- Minimal Sensor functionality
- Robustness
 - To network dynamics

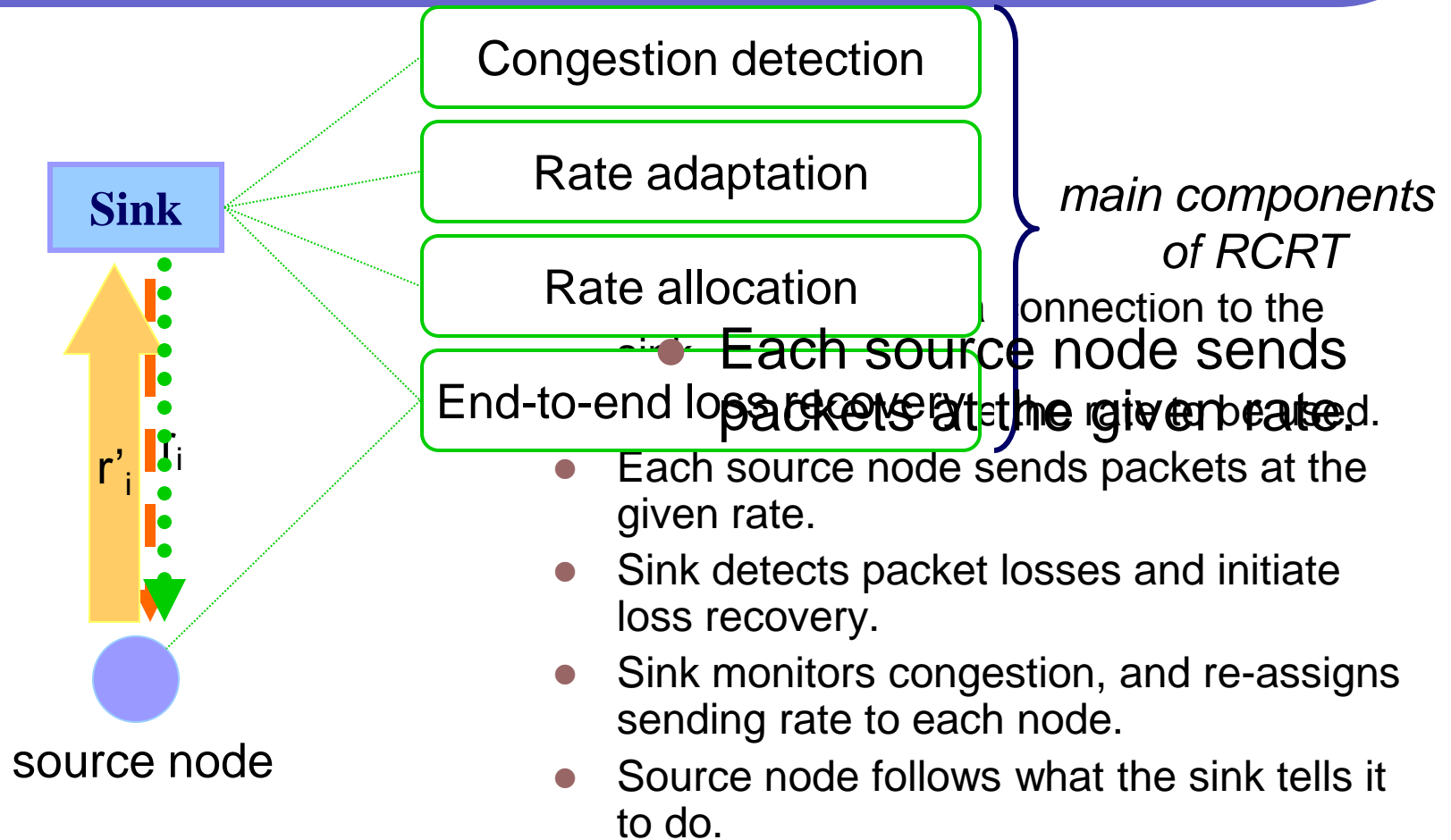
RCRT

- **Rate-Controlled Reliable Transport**

- In a wireless sensor network,
- A protocol that reliably transports sensor data from many sources to one or more sinks without incurring congestion collapse, at fair and efficient rate.

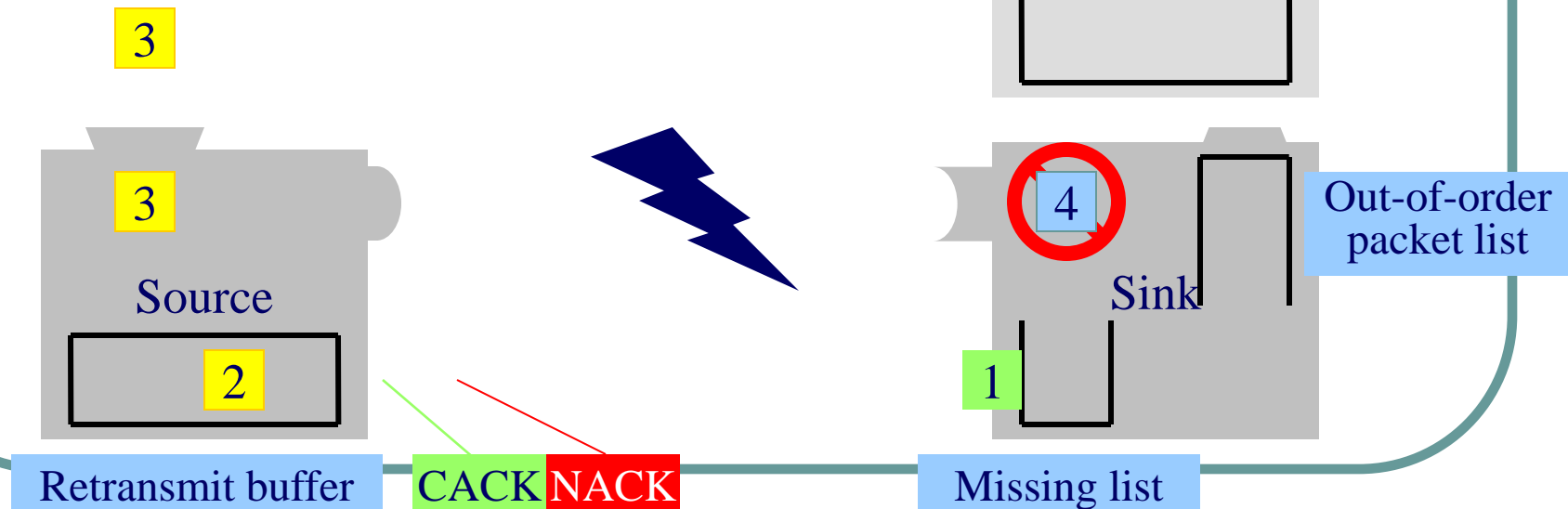


How it works...



End-to-end Loss Recovery

- Loss recovery mechanism
 - Negative ack. & cumulative ack.
 - End-to-end retransmission
- Data structures used for congestion control
 - Out-of-order packet list
 - Missing list



Congestion Detection

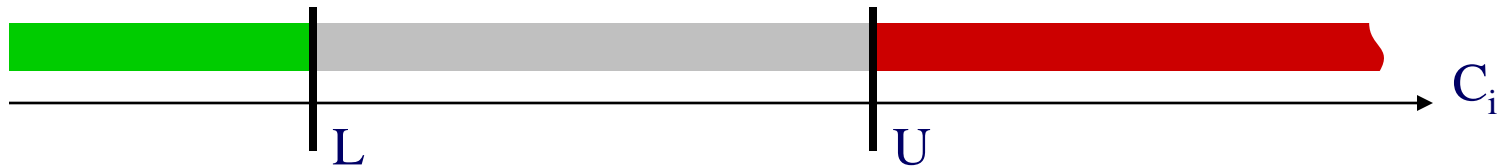
- Intuition:
 - "The network is not congested as long as end-to-end losses are repaired quickly enough"
- Use **'time to recover loss'** as congestion indicator

$$C_i = EWMA\left(\frac{L_i}{r_i RTT_i}\right)$$

Length of out-of-order packet list
+ (missing list length - 1)

Expected num. of packets in RTT

- Simple thresholding technique on C_i .



Under-utilized if $C_i \leq L, \forall i$

Congested if $C_i \geq U, \exists i$

Rate Adaptation

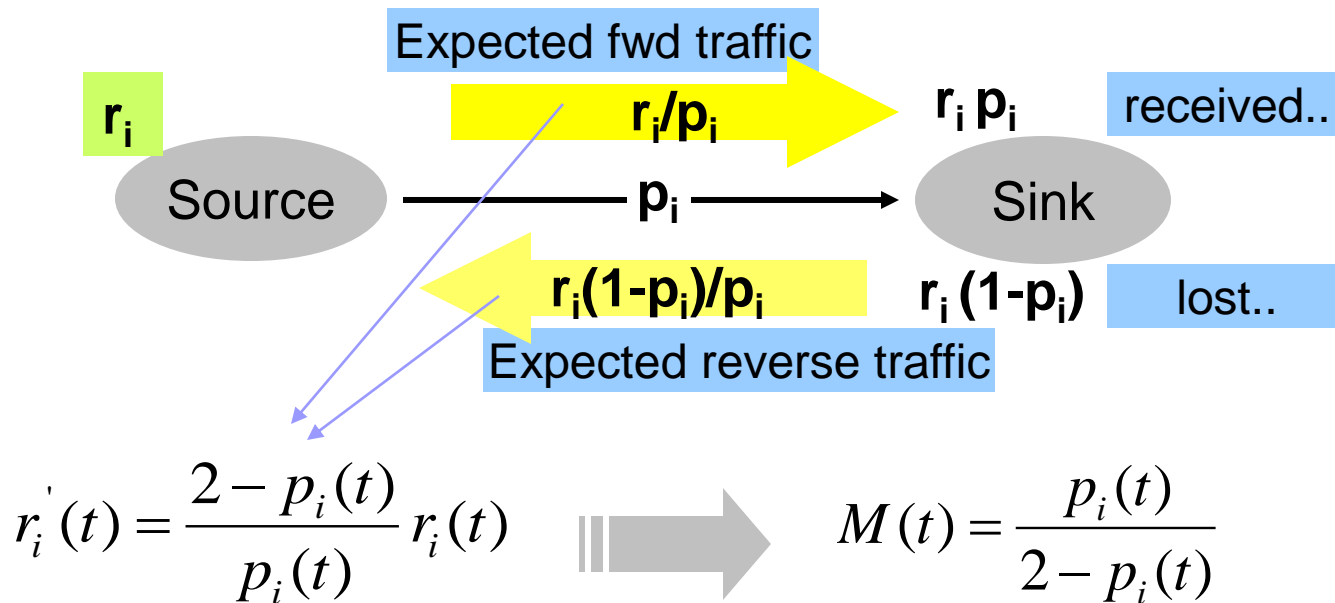
- **AIMD** on **total aggregate rate** of all the flows observed by sink:

$$R(t) = \sum r_i(t)$$

- Increase $R(t + 1) = R(t) + A$
 - Decrease $R(t + 1) = M(t)R(t)$
- When are the rate adaptation decisions made?
 - Only after when the previous decision has taken effect
 - How is $M(t)$ determined?
 - Can we be more efficient than always halving the rate?

Adaptive Multiplicative Decrease: $M(t)$

- Intuition:
 - When congested, actual amount of traffic is far greater than the source rate r_i , that was deemed sustainable.

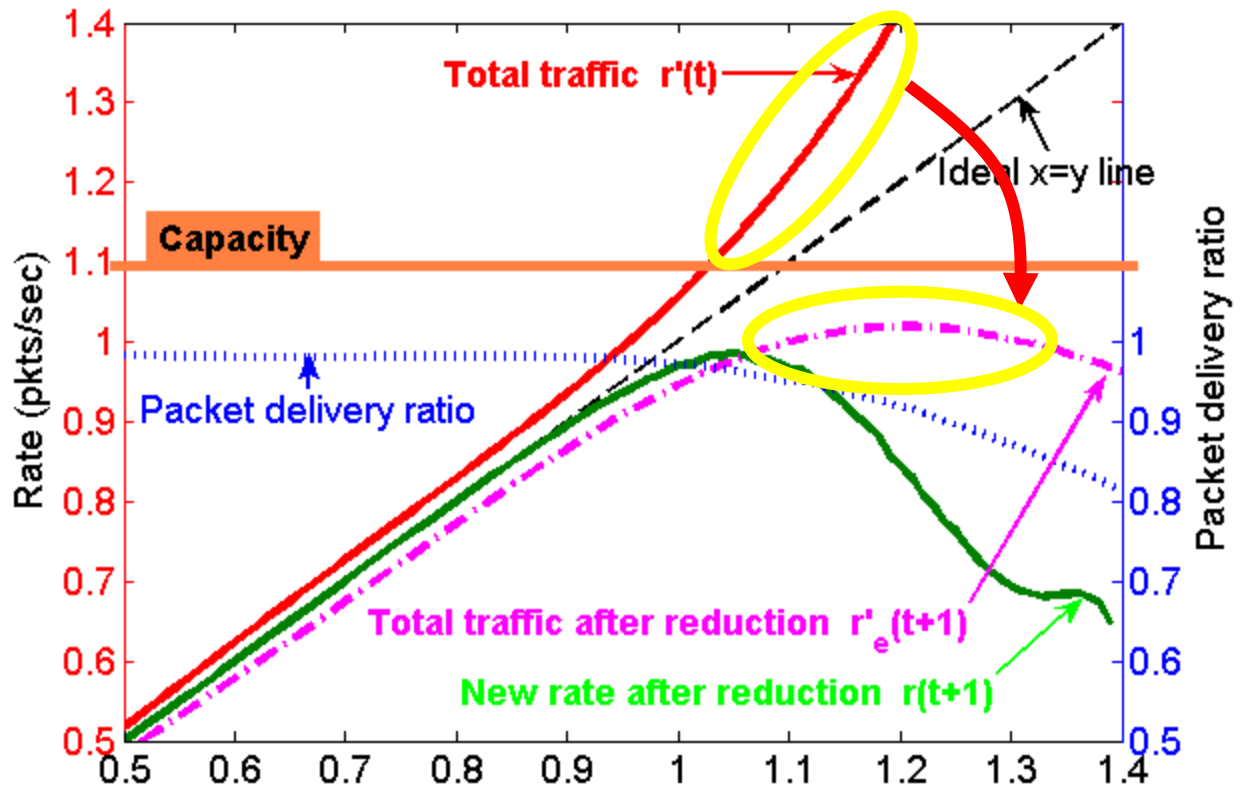


$M(t)$ is larger than **0.5** for $p_i \geq 0.67$

Does RCRT avoid congestion collapse?

Regardless of $r'_i(t)$, $r'_i(t+1)$ is a congestion w capacity.

CO



$M(t)$ is more aggressive when $r'_i(t)$ is higher

Rate Allocation

- Assign $r_i(t)$ to each flow based on the associated rate allocation policy P

$$R(t) = \sum r_i(t)$$

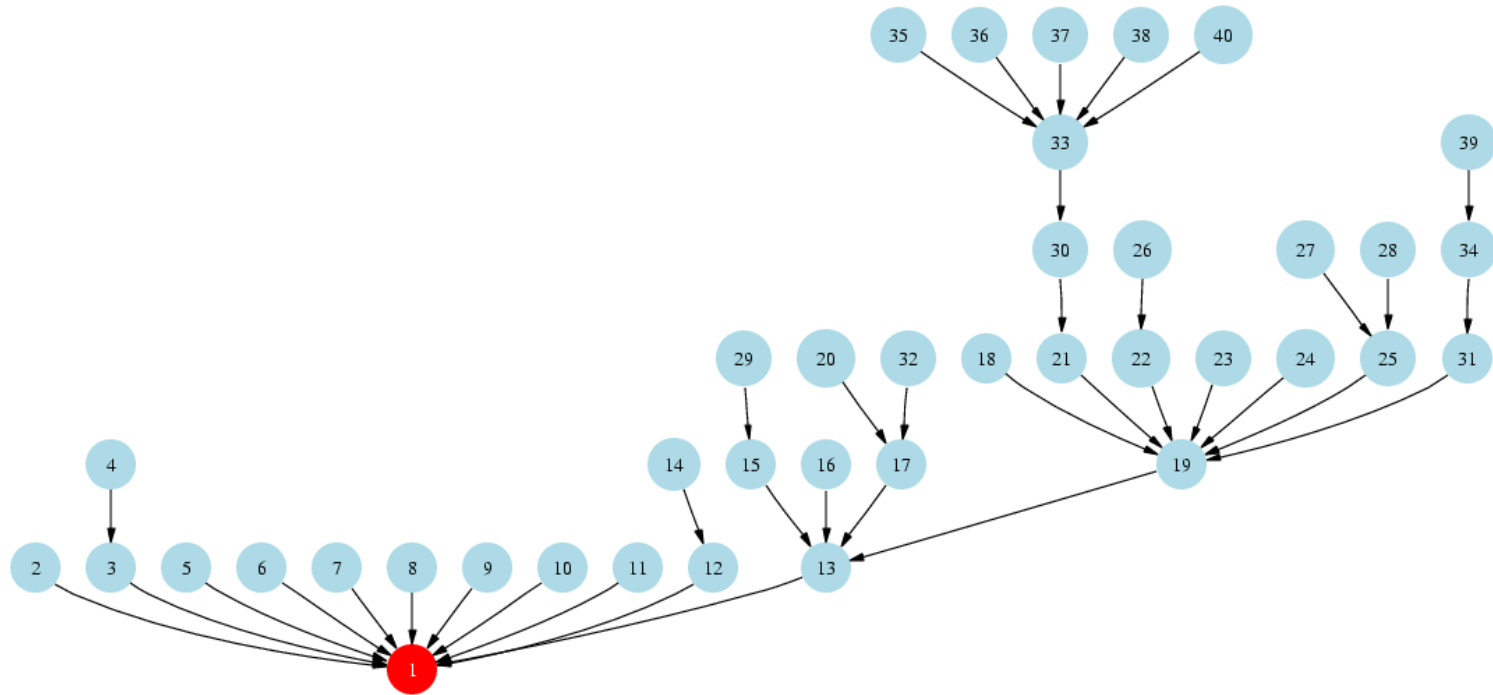
- Demand-proportional (Weighted)
- Demand-limited
- Fair

$$R(t)=4.5$$

$d_i = 2$	$d_j = 3$
$r_i = 1.8$	$r_j = 2.7$
$r_i = 2.0$	$r_j = 2.5$
$r_i = 2.25$	$r_j = 2.25$

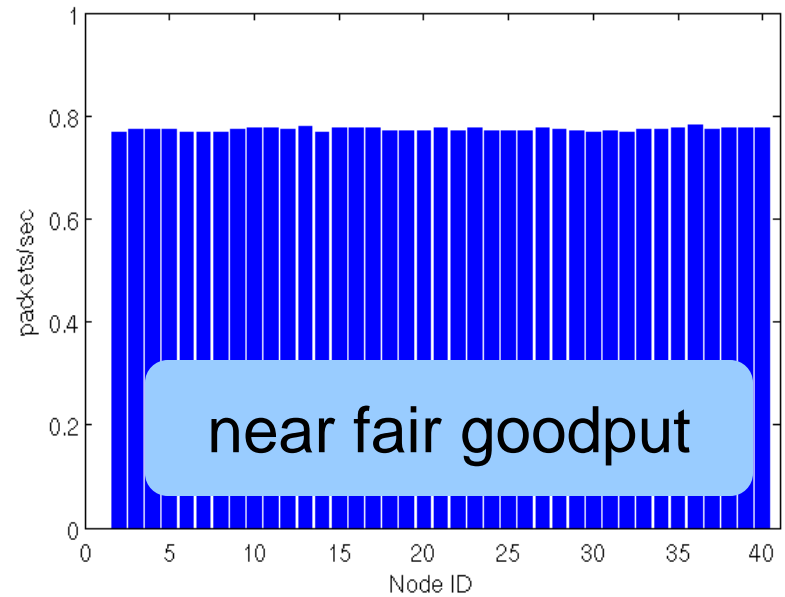
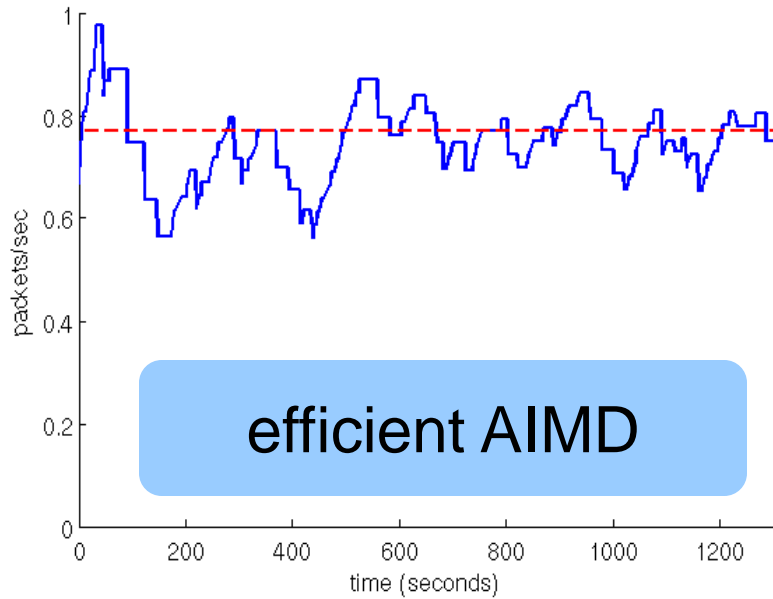
- Policy enforced at the sink
 - \rightarrow minimal sensor functionality
- Decouple adaptation from allocation
 - \rightarrow flexibility

Evaluation



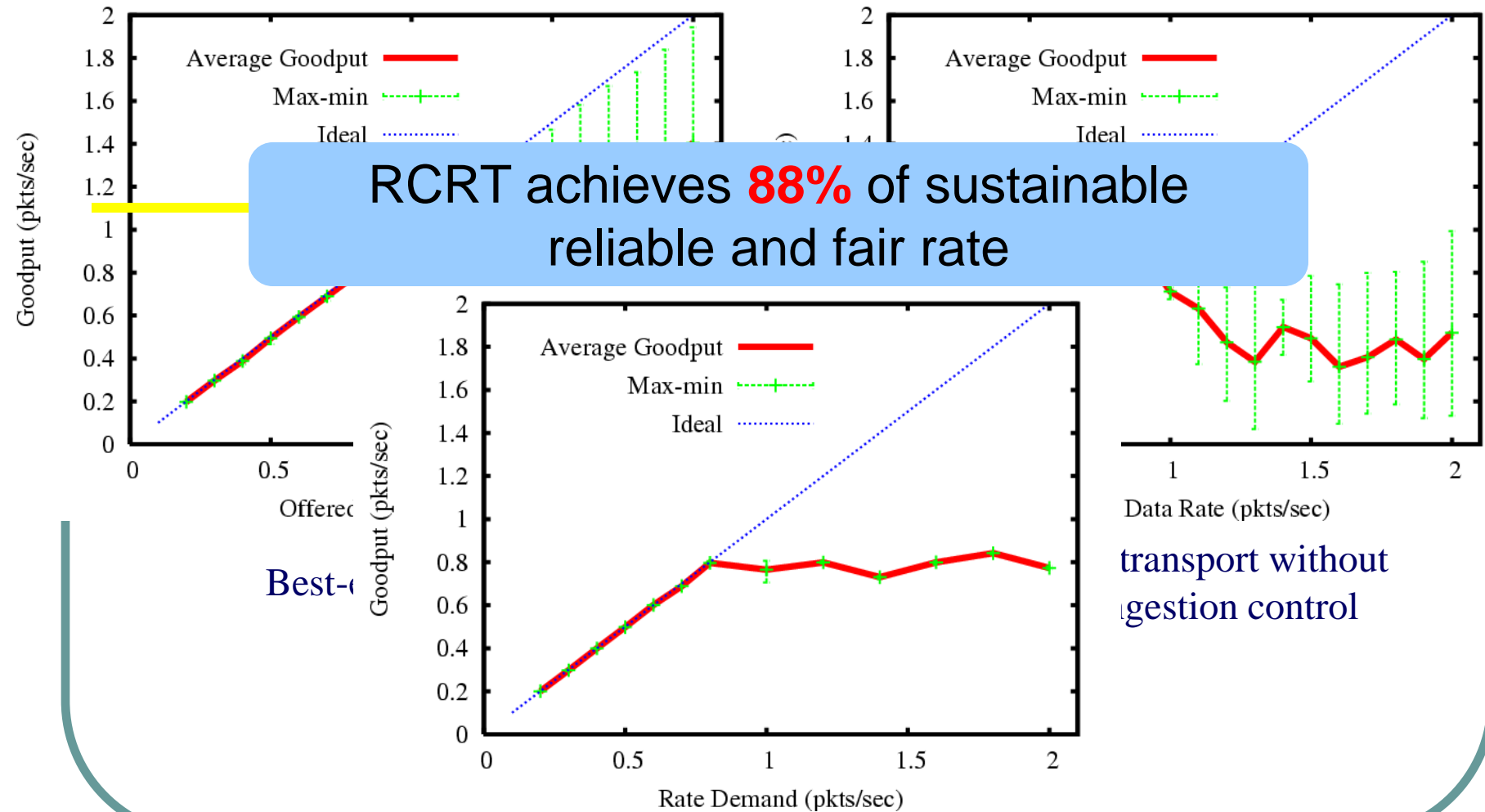
A snapshot of **routing tree**
during an experiment

RCRT Results



...and of course, **100%** reliable packet delivery

Optimality



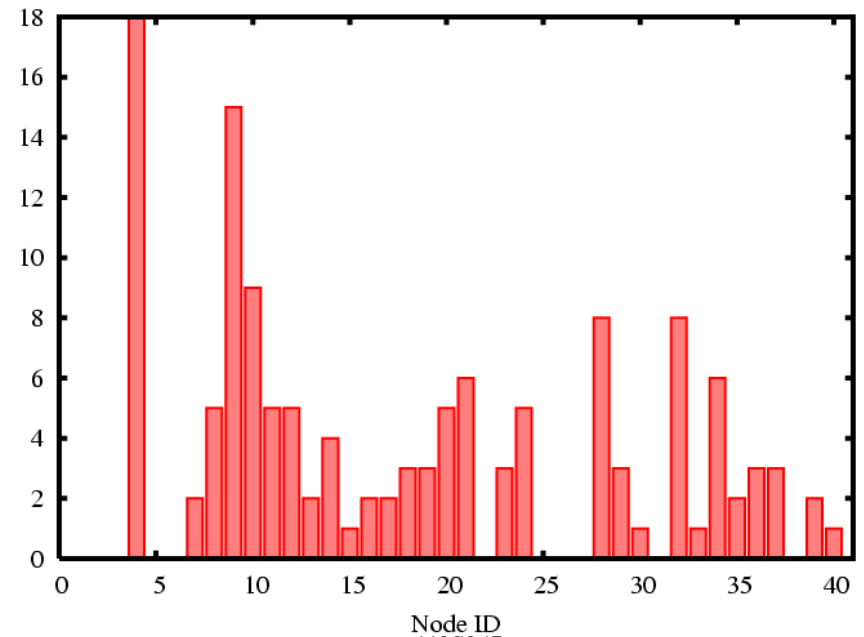
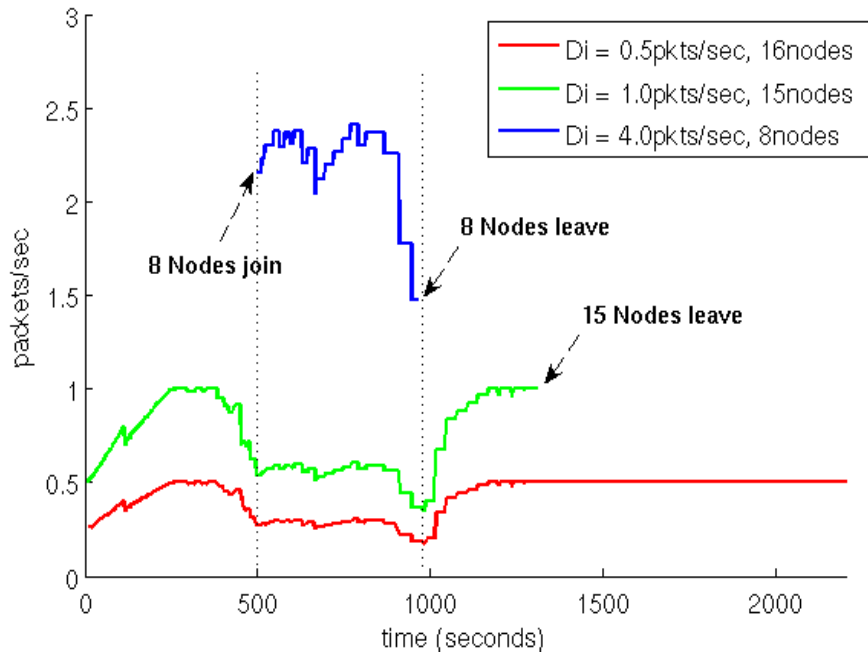
RCRT achieves **88%** of sustainable reliable and fair rate

Best-effort

transport without congestion control

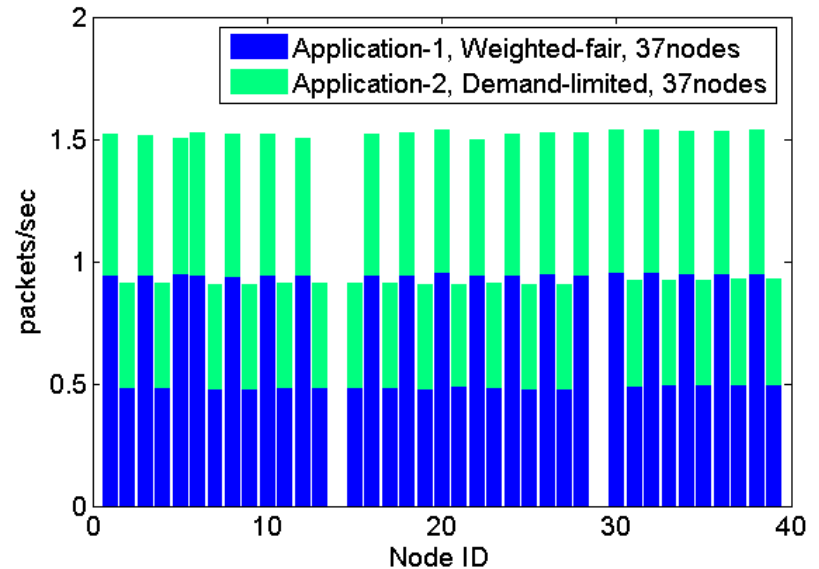
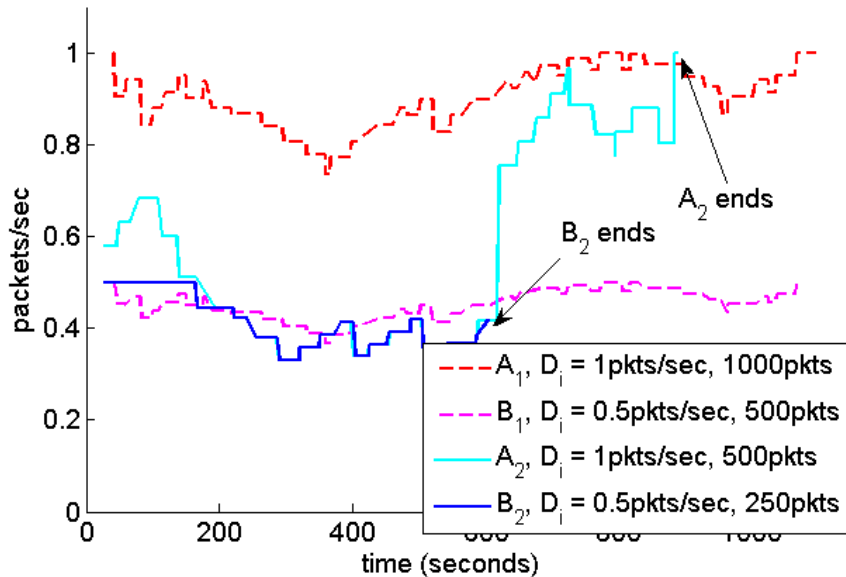
RCRT

Robustness to Network Dynamics



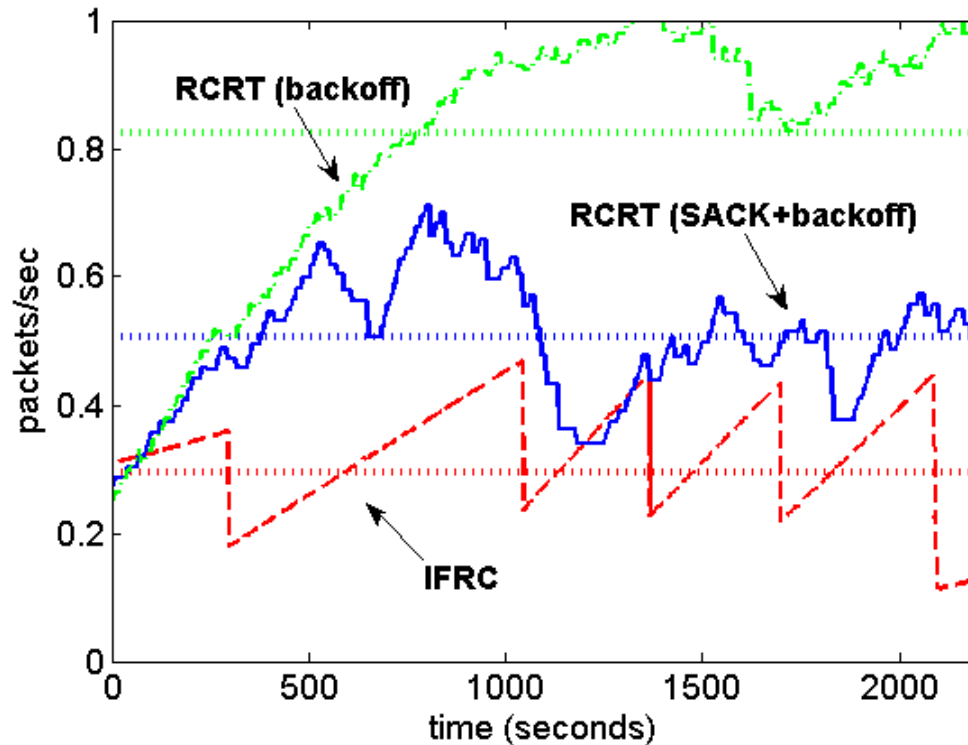
RCRT is **robust** to node joins & leaves,
and routing dynamics

Flexibility



Two **concurrent applications** with two different **rate allocation policies** ran successfully on a tiered multi-sink network.

Comparison with IFRC



RCRT achieves **x 1.7** the rate achieved by IFRC

Related Work

	Distributed Congestion Control	Centralized Congestion Control	No Congestion Control
Reliable	Flush, STCP	RCRT	Widren, Tenet, RMST
Unreliable	IFRC, Fusion, CODA	QCRA, ESRT	Surge, CentRoute, RBC

Conclusion

- RCRT is a reliable transport protocol for wireless sensor networks.
 - Centralized congestion control provides better perspective into the network, which enables better aggregate control of traffic and affords flexibility in rate allocation.

A Reliable Transport Protocol for Wireless Sensor Networks

**Mohammad Hossein Yaghmaee
and Donald Adjeroh**

IST 2008 Conference

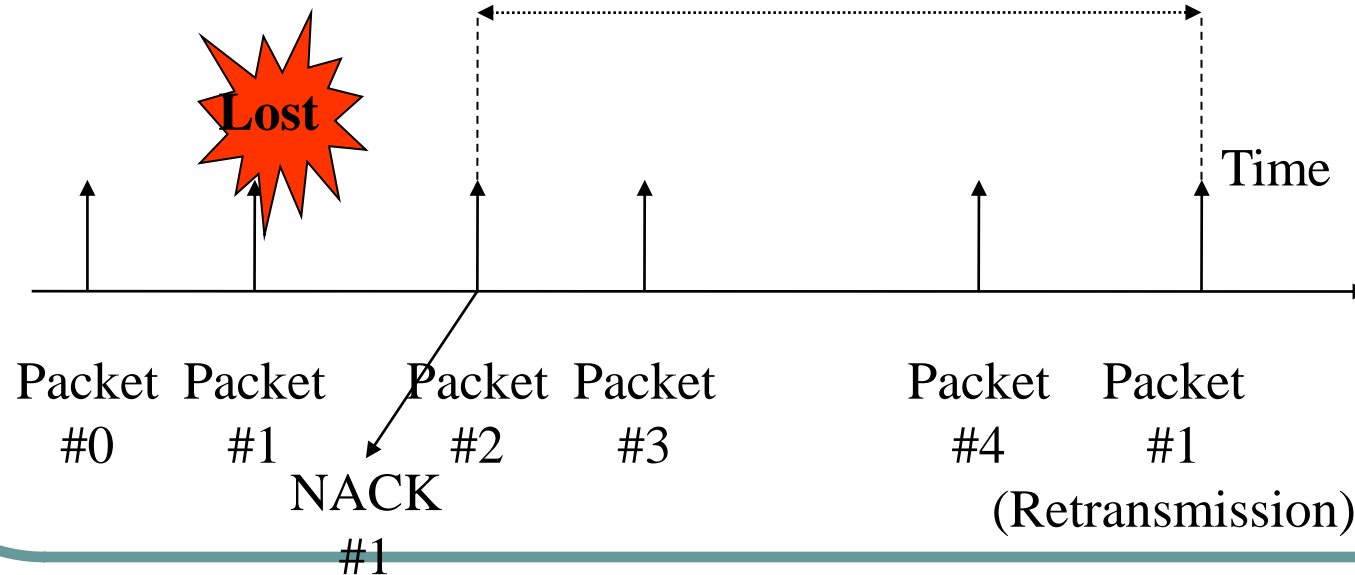
Proposed Model

- Hop-by-hop reliability guaranty
 - More energy efficiency
- Different types of buffer
 - Receive buffer
 - Packets which are received in order are placed in the receive buffer
 - Retransmission buffer
 - Packets which are received out of order are forwarded to the retransmission buffer.
 - Cache memory
 - A copy of each received packet is saved in a cache memory
 - When a node receives the ACK, it removes the packet from its local cache.

Congestion Detection

- **Time to recover packet loss** as a congestion detector
- Low congestion
 - The lost packet would be recovered very soon,
- High congestion
 - The packet lost recovery time is high

T_i



Congestion Degree

- D_i is the average delay between each node and its downstream node.
- CD_i is the Congestion Degree at node i

$$CD_i = (1 - \alpha)CD_i + \alpha \frac{T_i}{D_i}$$

- At each node, the value of congestion degree is forwarded to the sink node
- The sink node obtains the effective congestion degree :

$$CD_{eff} = \max \{CD_1, CD_2, \dots, CD_N\}$$

Congestion Detection

- The proposed model uses a simple threshold mechanism
- When $CD_{eff} \geq Max_{th}$
 - The network is congested.
 - The source rates of all network nodes should be decreased.
- When $CD_{eff} \leq Min_{th}$
 - There is no congestion in the network.
 - The source rates are increased
- When $Min_{th} < CD_{eff} < Max_{th}$
 - No changes in source rates

Rate Adjustment

- Additive Increase Multiplicative Decrease (AIMD) policy
- No congestion:

$$r_{Total}(t) = r_{Total}(t) + \lambda_I$$

- Congestion:

$$r_{Total}(t) = r_{Total}(t) \cdot \lambda_D(t)$$

- λ_I is a constant value and $\lambda_D(t)$ is a time-dependent multiplicative decrease factor

Packet Loss Estimation

- Using the Average Loss Interval (ALI) method.
- Suppose that S_k ($k = 1, \dots, 8$) be the number of packets in the k -th most recent loss interval

$$\hat{s}_0 = \frac{S_0 + S_1 + S_2 + S_3 + 0.8S_4 + 0.6S_5 + 0.4S_6 + 0.2S_7}{6}$$

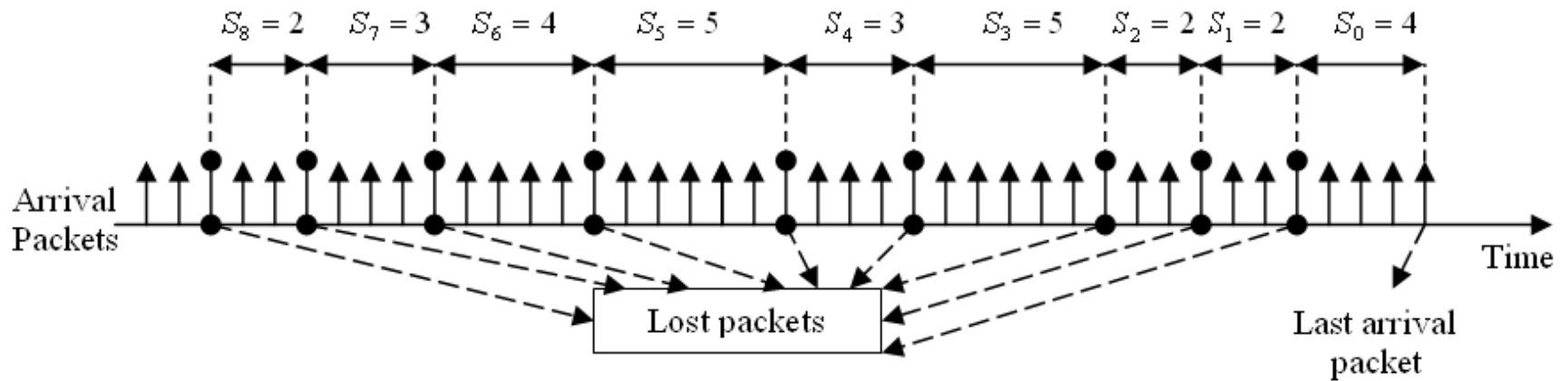
$$\hat{s}_1 = \frac{S_1 + S_2 + S_3 + S_4 + 0.8S_5 + 0.6S_6 + 0.4S_7 + 0.2S_8}{6}$$

- Packet loss:

$$P_{loss}^i = \frac{1}{\max(\hat{s}_0, \hat{s}_1)}$$

- Multiplicative decrease factor $\lambda_D(t) = \min_{i=1}^N \frac{1 - P_{loss}^i}{1 + P_{loss}^i}$

Example



$$\hat{s}_0 = \frac{20.6}{6} \approx 3.4$$

$$\hat{s}_1 = \frac{20}{6} \approx 3.3$$

$$P_{loss} = \frac{1}{3.4} = 0.294$$

Loss Recovery

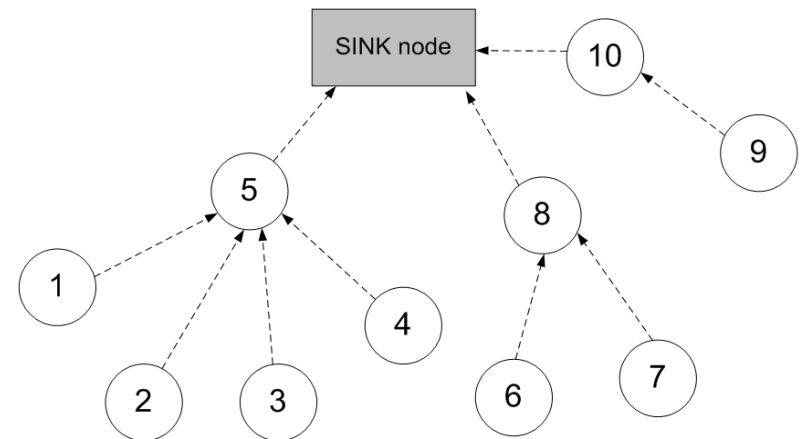
- When a node detects a lost packet, an NACK message is sent to the next hop on the reverse path toward the source.
- If it is found in the local cache, a copy of the lost packet is retransmitted.
- If not, the NACK message is forwarded to the next hop toward the source.
- Each packet must contain a sequence number.
- Each node uses a timer based loss detection mechanism.
- NACK based
 - Upon receiving a NACK, the node retransmits the requested packets to repair the losses.

Simulation Results

- A simulation software was developed in C++ language on LINUX OS.
- All sensor nodes have a random service time.

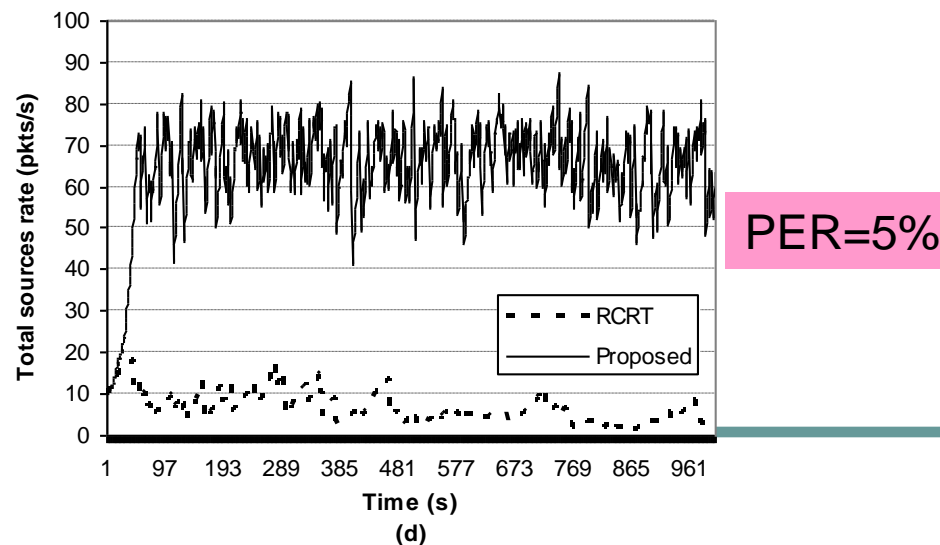
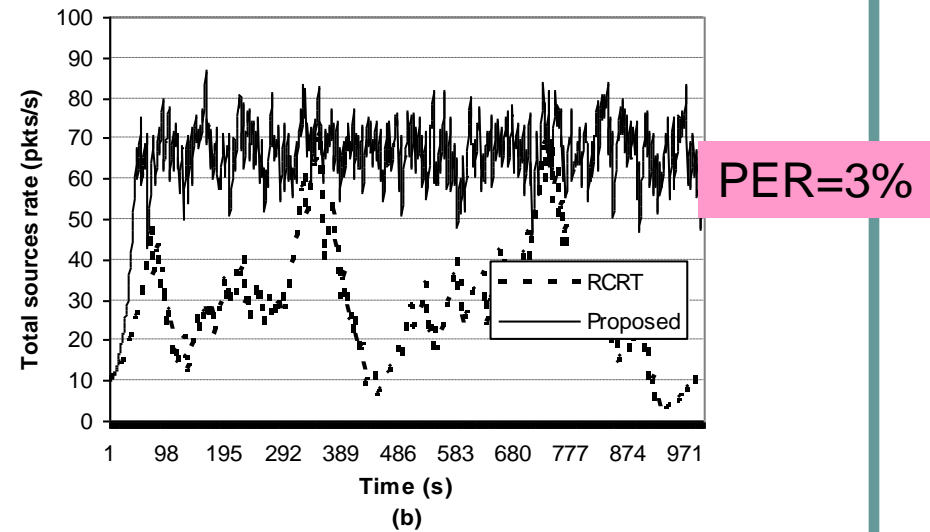
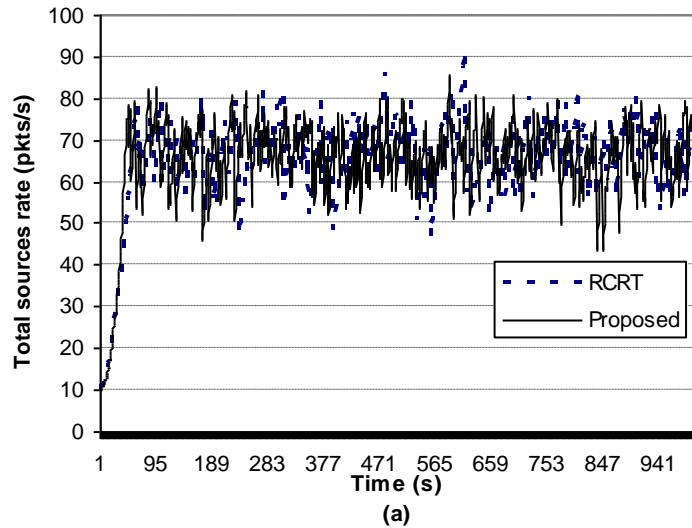
Parameter	Value
Buffer size	100 Pkts
Mean service time	0.01 s
Min_{th}	1
Max_{th}	4
r^i_{max}	9.5 pkts/s
λ_I	0.05 pkts/s
Simulation time	1000 s

Simulation Parameters

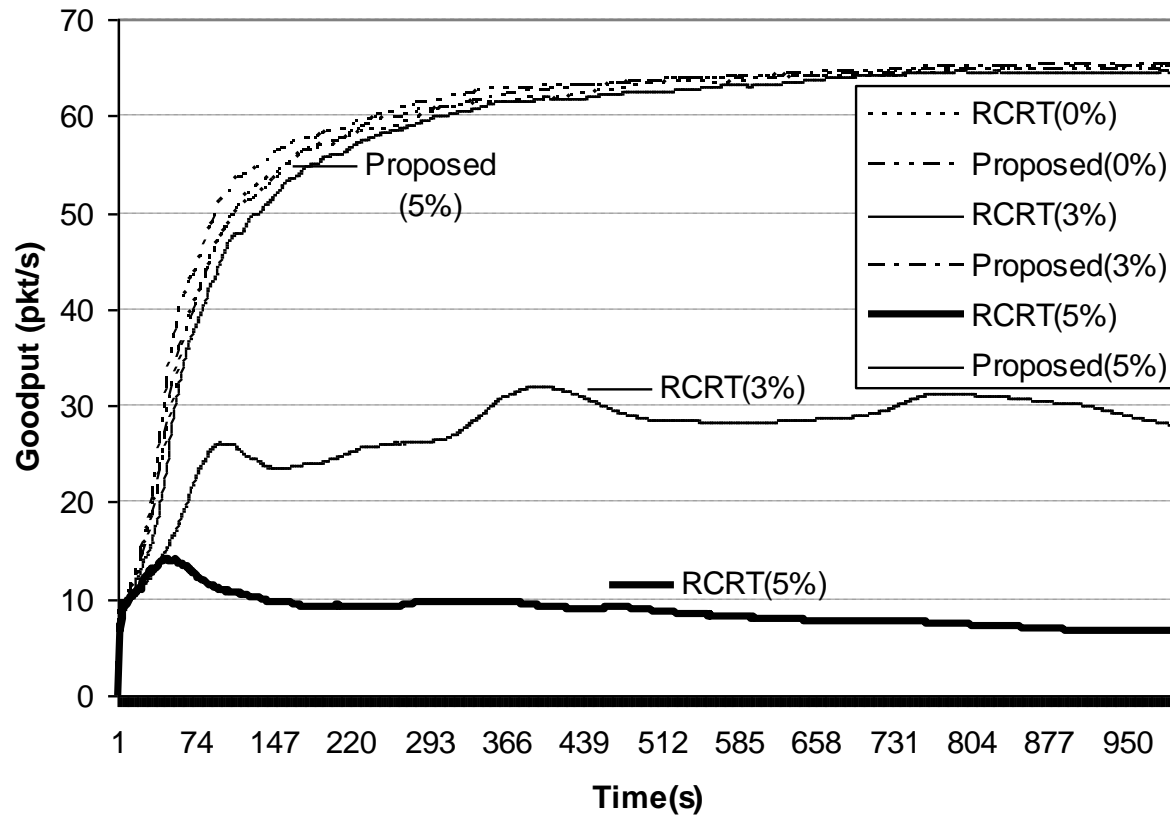


Network Topology

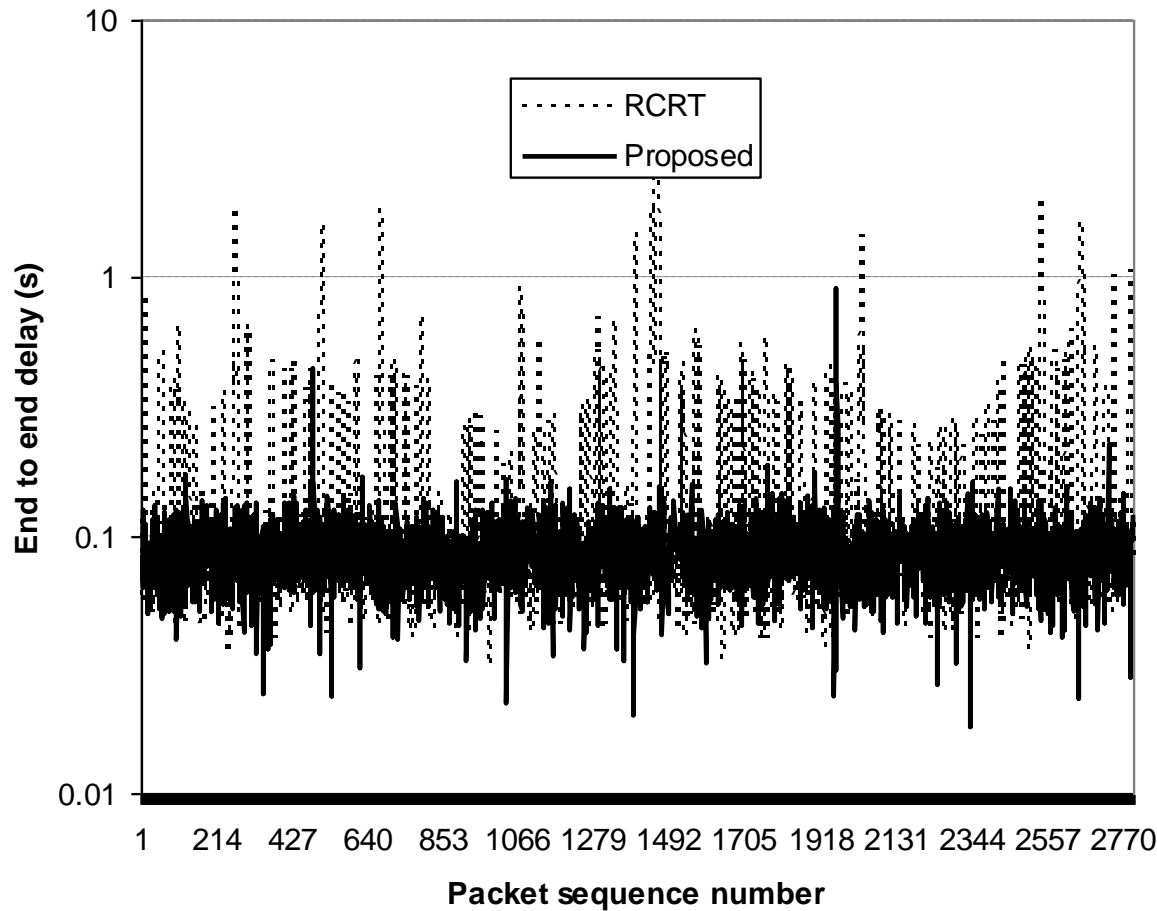
Total Source Rate Versus Simulation Time at Different Value of PER



Total Goodput Versus Simulation Time at Different Values of PER

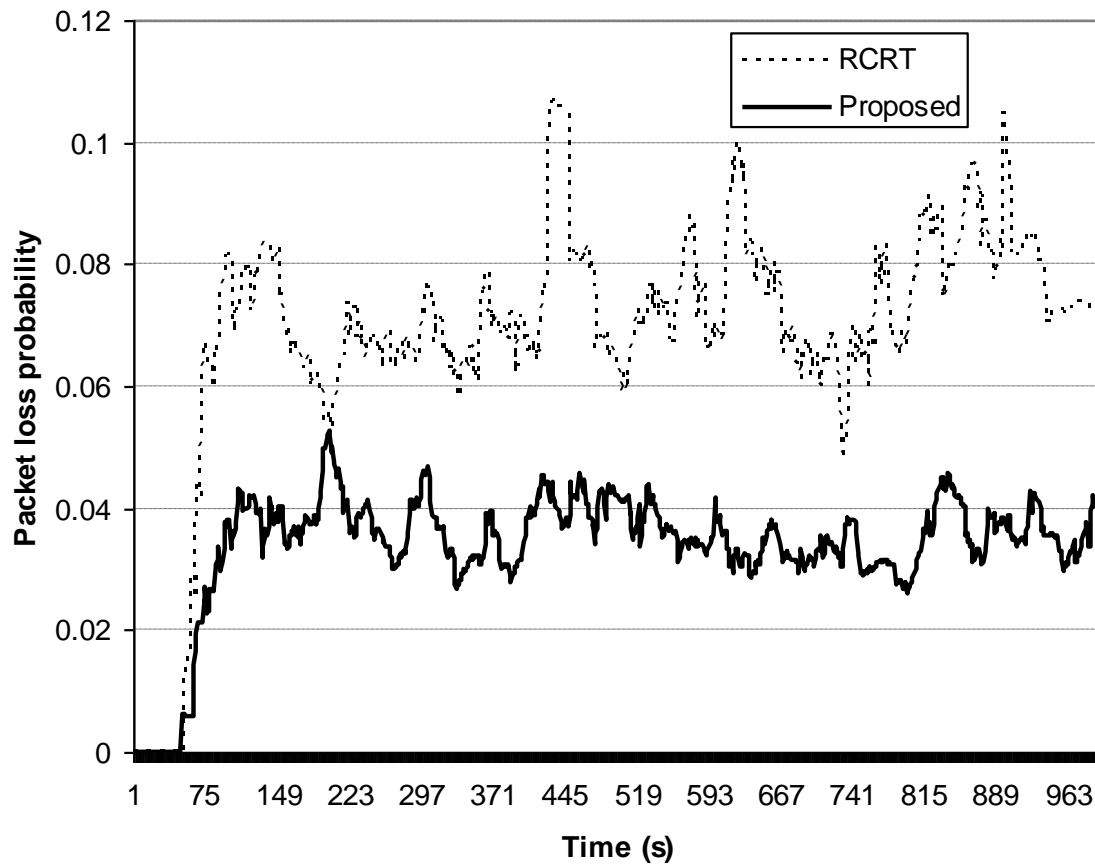


End-to-end Delay



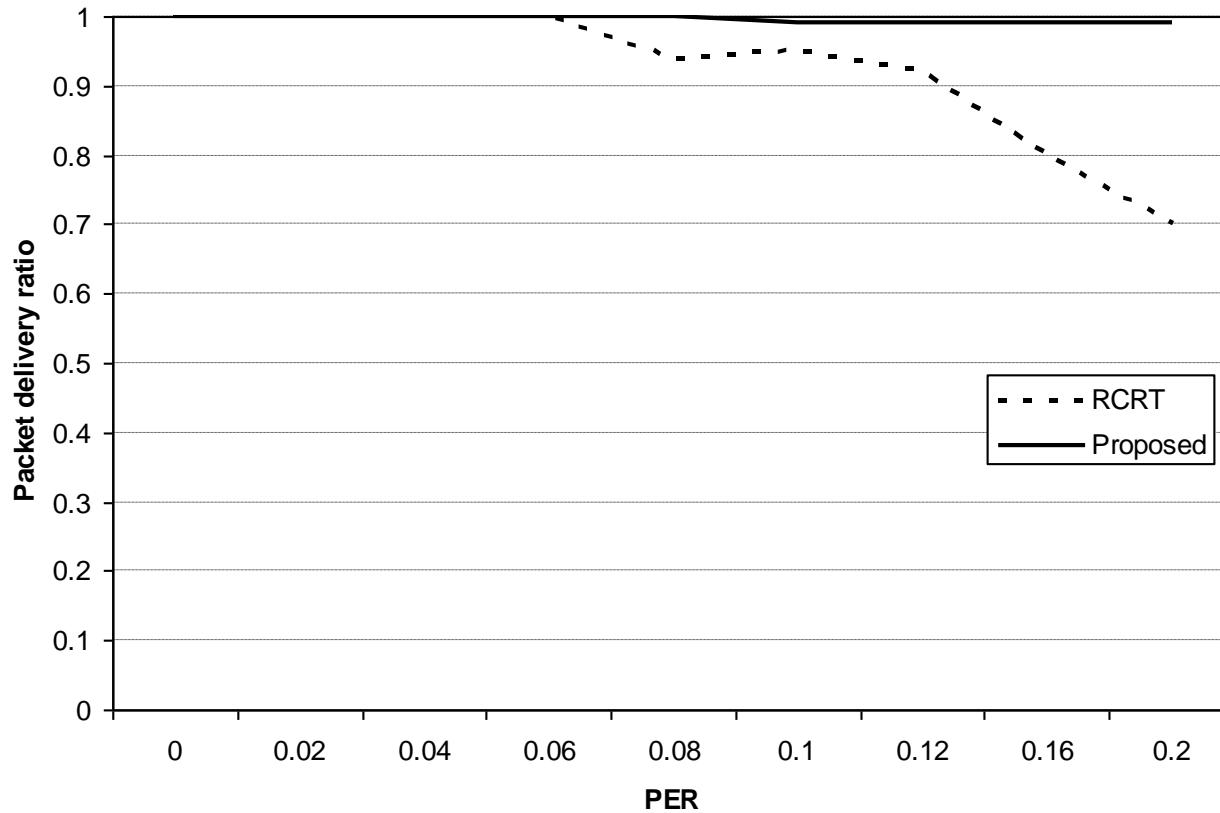
Average
RCRT: 0.142s
Proposed: 0.078s

Packet Loss Probability Versus Simulation Time

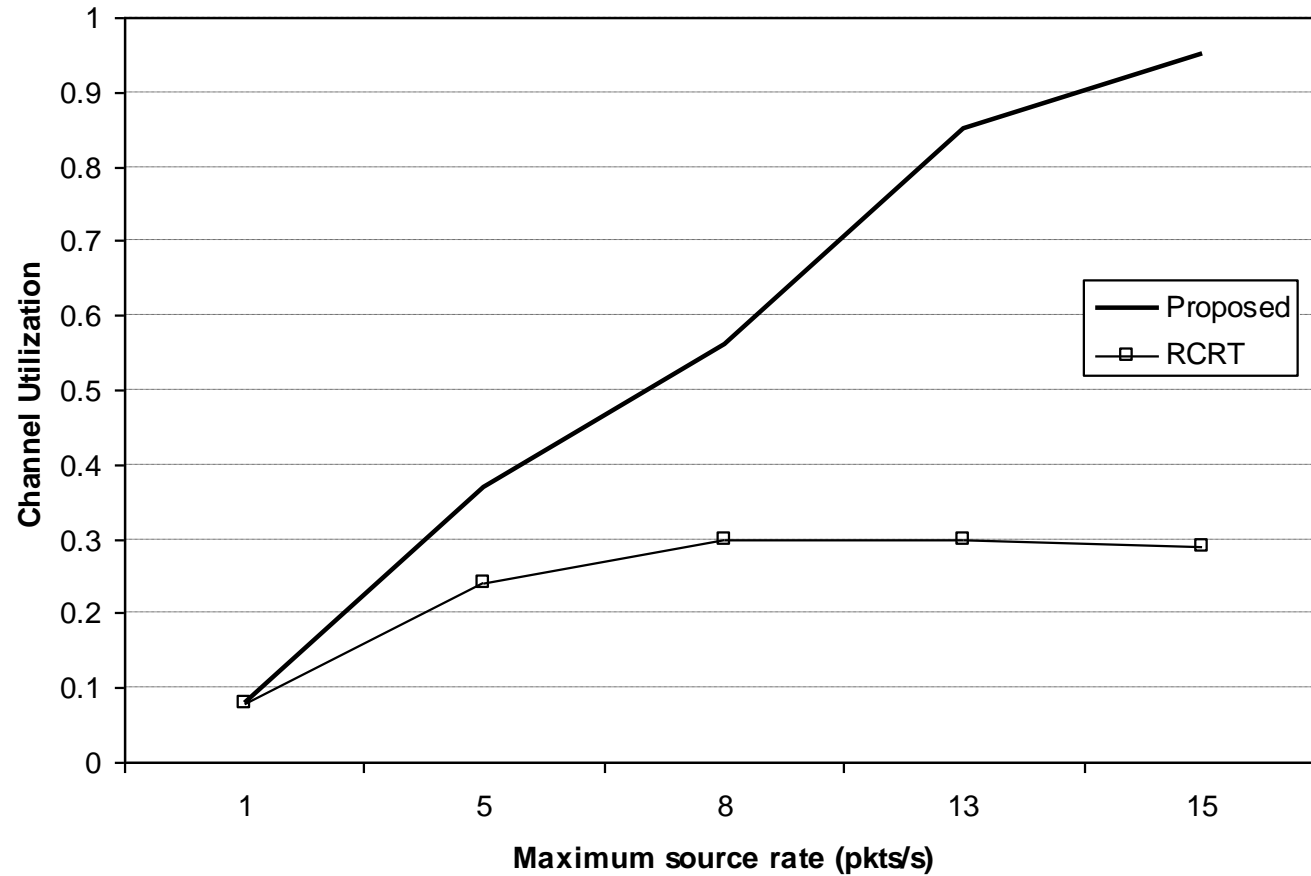


Average
RCRT: 0.068
Proposed: 0.033

Packet Delivery Ratio Versus PER



Channel Utilization Versus PER



Conclusion

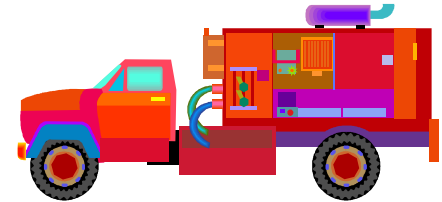
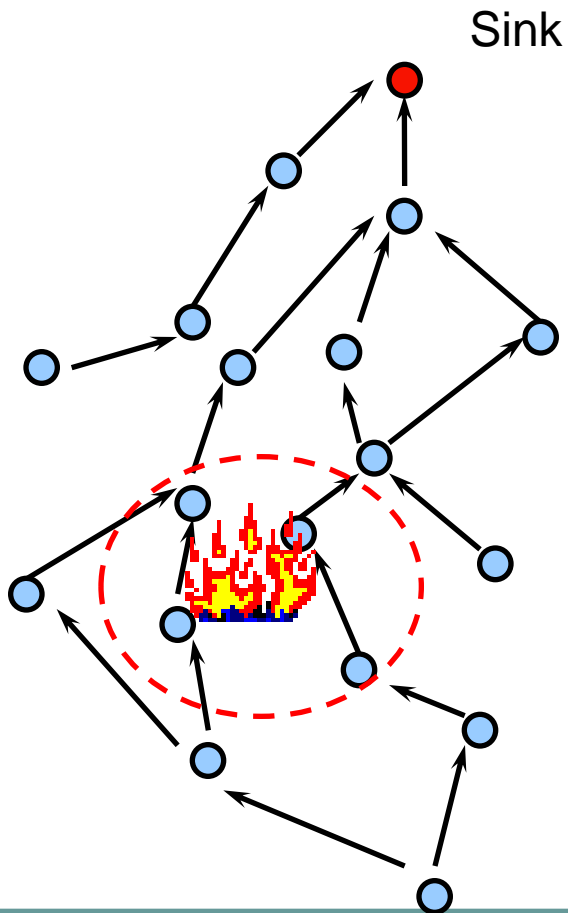
- Congestion control and loss recovery are two important issues in transport layer.
- We presented a new protocol for congestion control and loss recovery
 - Time to recover loss as congestion indicator
 - Hop by hop loss recovery
 - AIMD rate adjustment
 - End to end rate assignment
- Simulation results confirms that the proposed protocol has better performance than the RCRT protocol.

PORT: A Price-Oriented Reliable Transport Protocol for Wireless Sensor Networks

**Yangfan Zhou, Michael. R. Lyu,
Jiangchuan Liu and Hui Wang**

The Chinese University of Hong Kong
ISSRE 2005

Introduction



Introduction

- Reliable sensor-to-sink data transport for WSN
 - It is important as it ensures the mission of the networks
 - Objective
 - To ensure that the sink can receive desired information
 - The work presented here is to address this problem.

Introduction

● WSN Challenges

- WSN suffers from energy constraints
- WSN conditions
 - Unreliable wireless link

● High and dynamic packet loss rate

● Network Dynamics

- Node failures
- Link failures
- Dynamic traffic load

What Is Addressed

- What should a reliable sensor-to-sink data transport protocol do?
 - Ensure that the sink can collect enough information
 - Minimize energy consumption of data transport
- How should it be designed to achieve the goals?
 - With cooperation of the application layer
 - Adjust the reporting rates of sources
 - Adapting to wireless communication conditions

Presentation Outlines

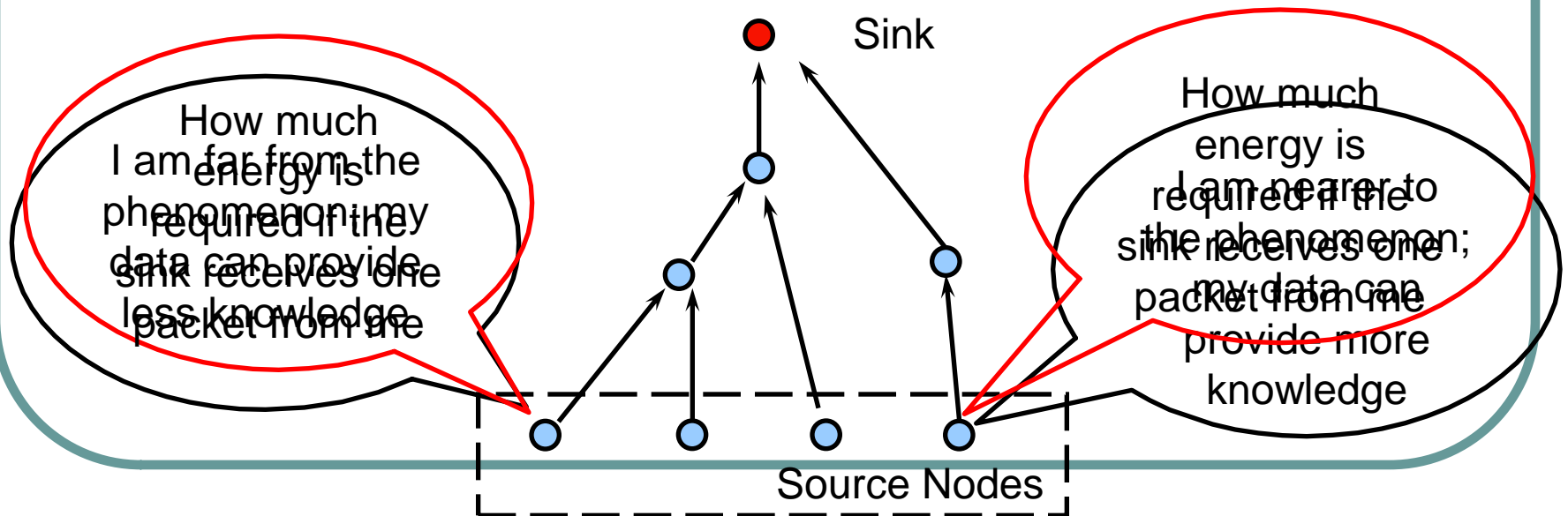
- 1. Introduction
- 2. **Design Considerations**
- 3. Protocol Implementation
- 4. Simulation Results
- 5. Conclusion

Reliable Sensor-to-Sink Data Transport

- Ensure that the sink can obtain enough fidelity of the knowledge on the phenomena of interest
 - 100% packet delivery is not necessary.
 - The key is that the desired information can be obtained.
 - Only the application that utilizes the packets knows whether the data transport is reliable or not.

Observations

- Different sources have different contributions to improve the sink's knowledge on the phenomena of interest (known by the application)
- Different energy is required for communications between different sources to the sink (known by the transport protocol)



Presentation Outlines

- 1. Introduction
- 2. Design Considerations
- 3. **Protocol Implementation**
- 4. Simulation Results
- 5. Conclusion

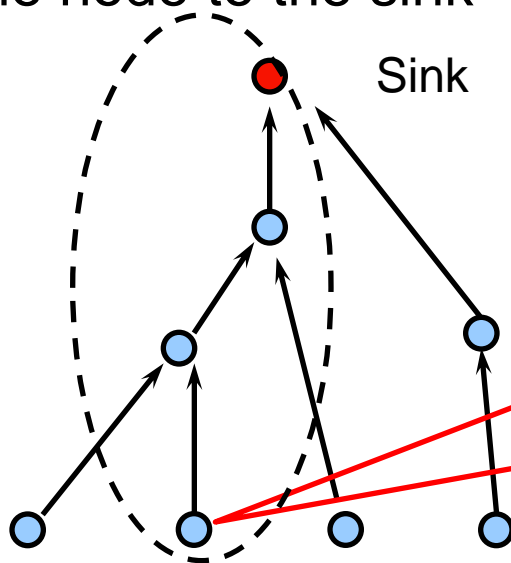
Protocol Requirements

- A good sensor-to-sink communication cost estimation mechanism
- A good routing scheme to achieve energy efficiency as well as in-network congestion avoidance.
- A feedback mechanism to adjust each source's reporting rate

Communication Cost Estimation

- Node Price (NP)

- A node's node price is the energy consumed by all the in-network nodes for each packet successfully delivered from the node to the sink



If the average energy consumed by the network to successfully deliver a packet from me to the sink is N , then my node price is N .

Node Price Calculation

- Calculated in a backward propagating way
 - The node prices of a node's possible downstream neighbors
 - Obtained by the feedbacks of its downstream neighbors
 - The energy consumed to send packet to each downstream neighbor
 - Calculated with link loss rate to each downstream neighbor
 - The proportion of traffic the node sends to each downstream neighbor
 - Determined by its routing scheme

Node Price Calculation

- Link loss rate
 - Mainly caused by three factors
 - Congestion
 - Signal Interference
 - Fading.
 - Packet loss rate will exhibit graceful increasing behavior as the communication load increases (IEEE 802.11 MAC)
 - Reasonable to estimate the packet loss rate based on an EWMA (Exponential Weighted Moving Average) approach.

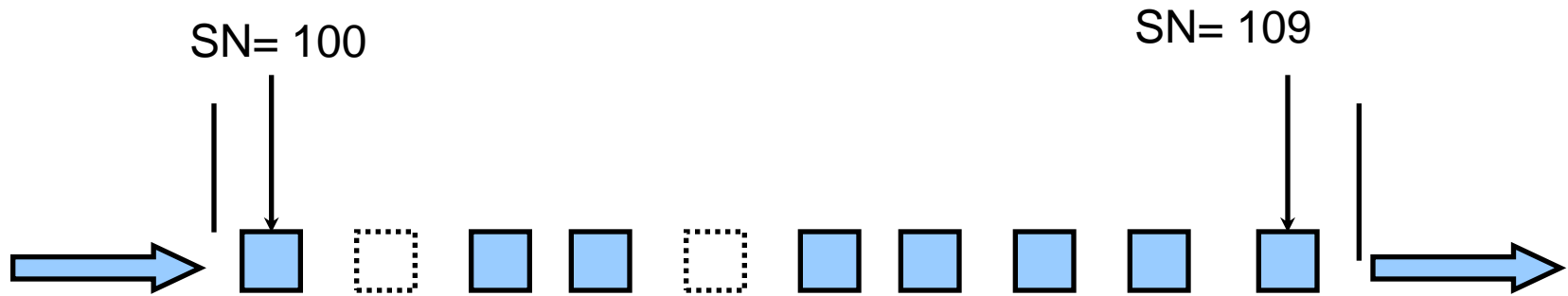
$$\text{Current Estimation} = \frac{(1 - a) \times \text{Current Loss} + a \times \text{Previous Estimation}}{1}$$

Node Price Calculation

- Estimate the link loss rate to each downstream neighbor
 - Accurate and current link loss rate estimation
 - Well indicates the congestion condition
 - Well indicates the weak link
 - Node Price: based on loss rate estimation
 - well indicates the dynamic wireless communication condition from the node to the sink
 - can help to determine the reporting rates
 - can help to determine the routing scheme

Node Price Calculation

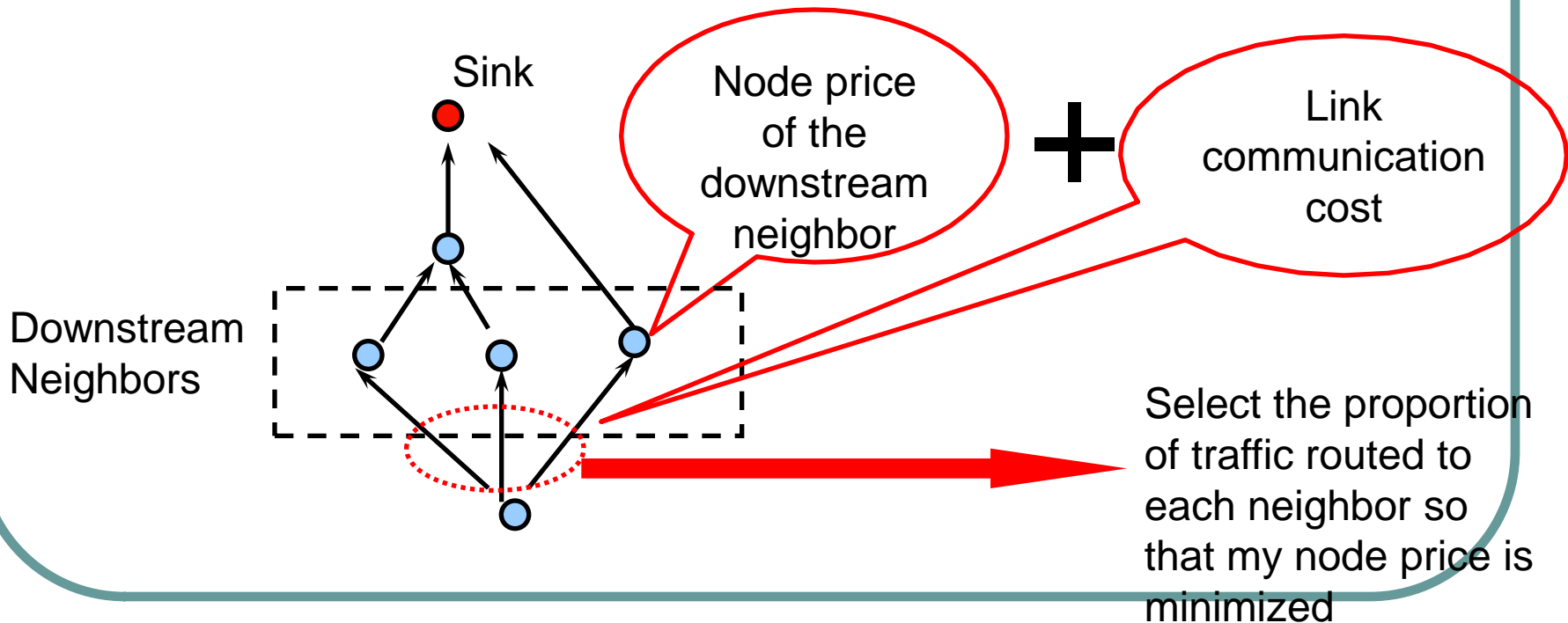
- Link loss rate estimation
 - Measured according to packet serial numbers holes
 - Estimated with an EWMA approach.



$$\text{Measured Loss Rate} = 2 / (109 - 100 + 1) = 20\%$$

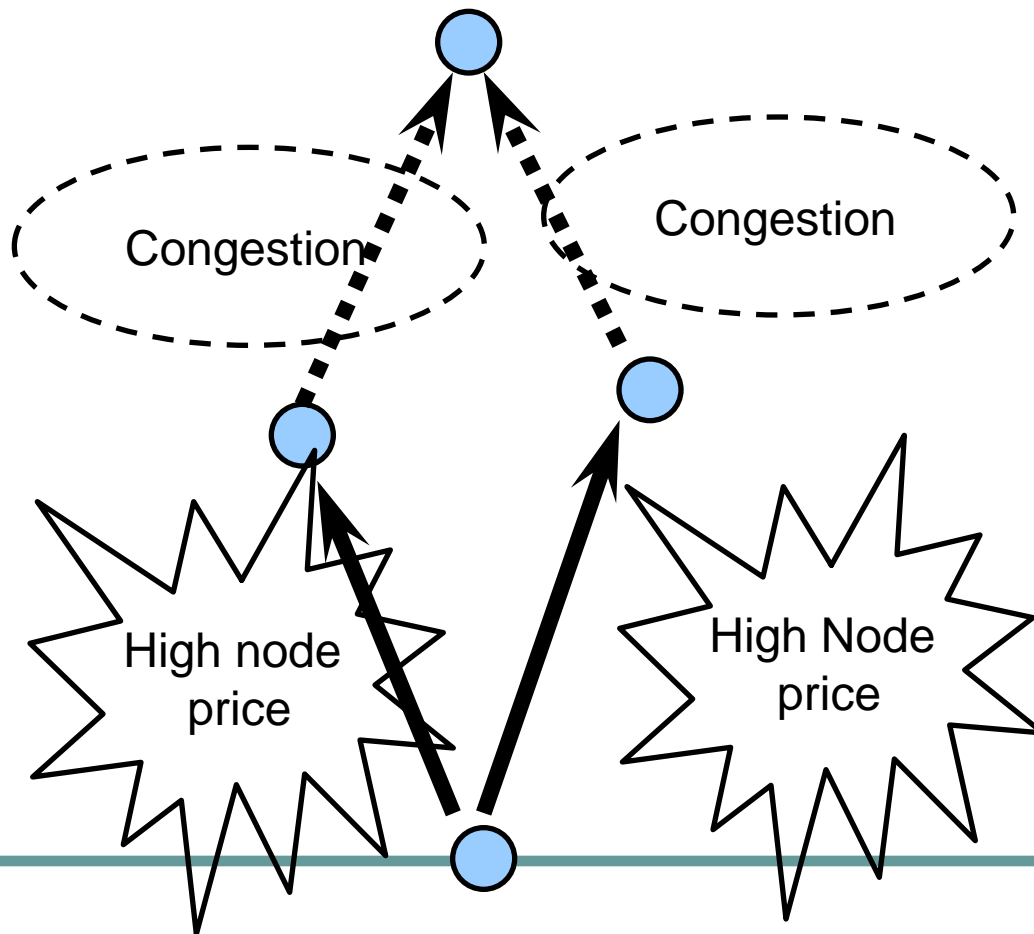
Routing Schemes

- Minimizing local node price.
 - A node should minimize the energy consumed for the network to successfully deliver a packet to the sink from the node



Routing Schemes

- Oscillation Avoidance



Routing Schemes

- Oscillation Avoidance

- Gradually shift traffic to best path
- Adaptive to downstream dynamics

Traffic proportion shifted to a better node (with lower NP) at each time is:

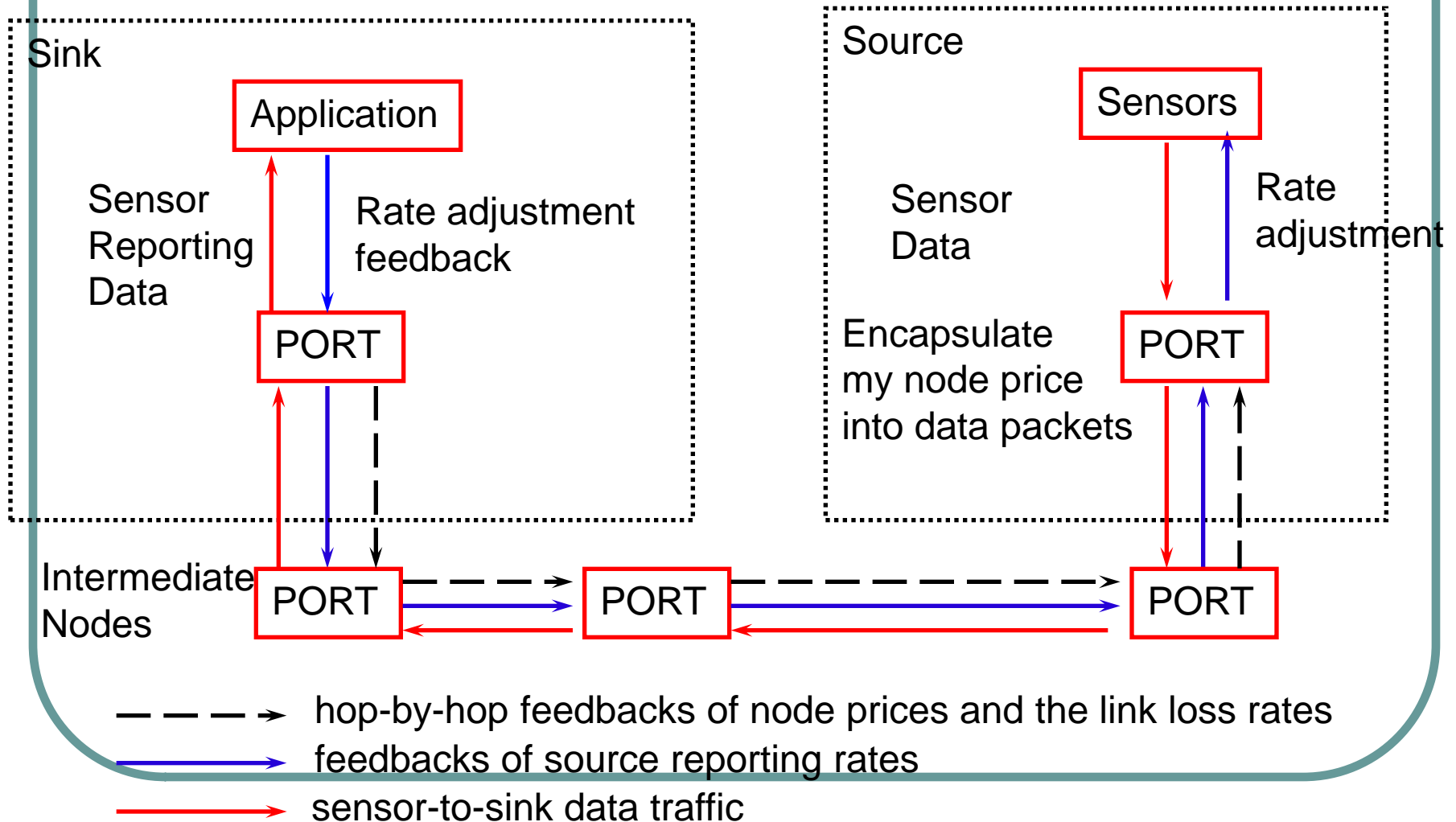
$$Perc_{higher} \times \frac{NP_{higher} - NP_{lowest}}{NP_{higher}}$$

Node price differences of the better neighbor and each of the worse neighbor

Current traffic proportion that is sent to the worse neighbor

Node price of the worse neighbor

Diagram of PORT



Presentation Outlines

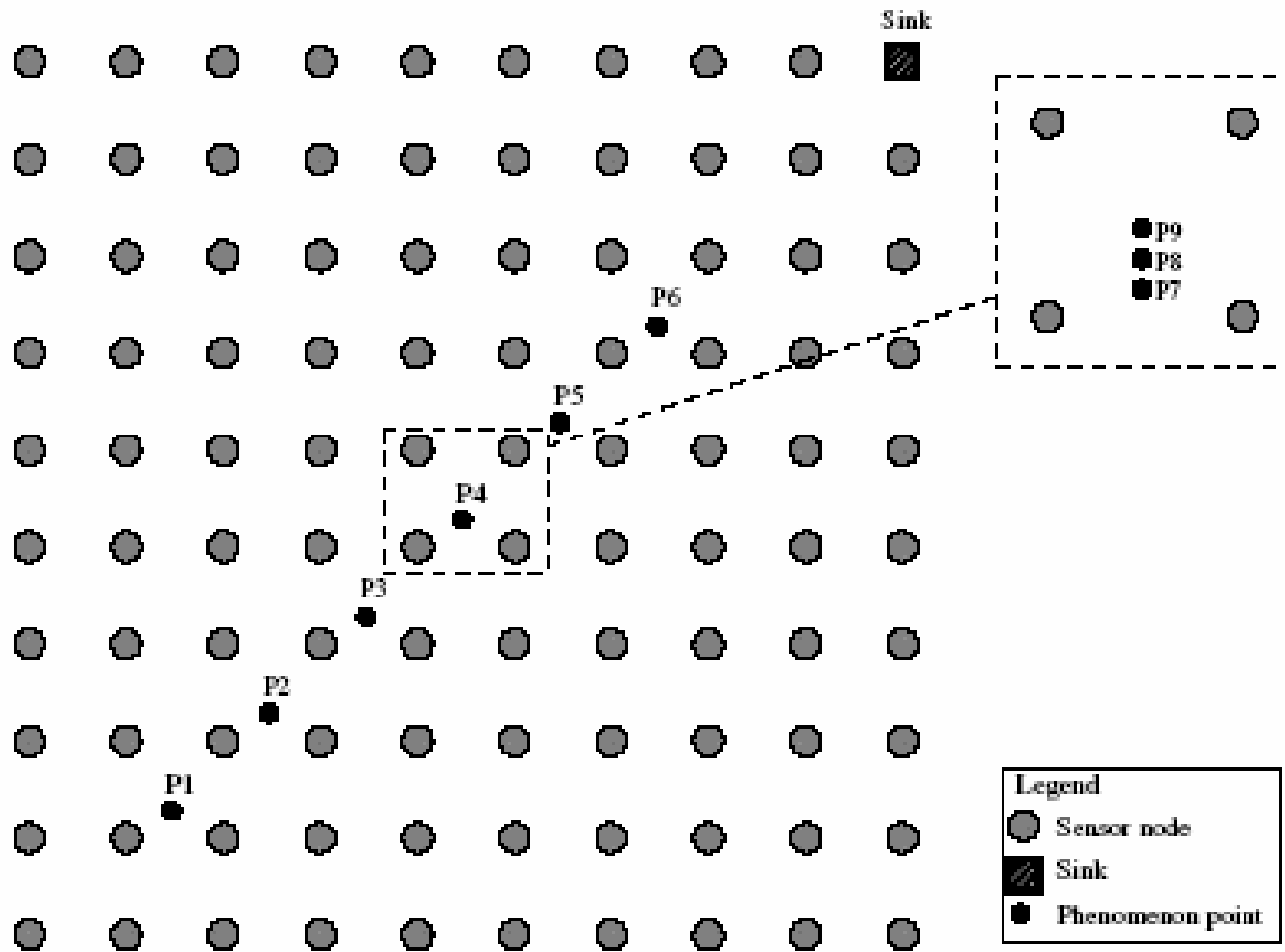
- 1. Introduction
- 2. Motivations and Design Considerations
- 3. Protocol Implementation
- 4. **Simulation Results**
- 5. Conclusion

Simulation Settings

- Coding PORT over NS-2
- Simulation Settings

Area of sensor field	1500m*1500m
Number of sensor nodes	100
MAC	IEEE 802.11 without CTS/RTS and ACK
Radio power	0.2818
Packet length	36 bytes
Transmit Power	0.660 W
Receive Power	0.395 W
Feedback interval	1 second
IFQ length	50 packets
Simulation Time	1000 seconds

Simulation Networks

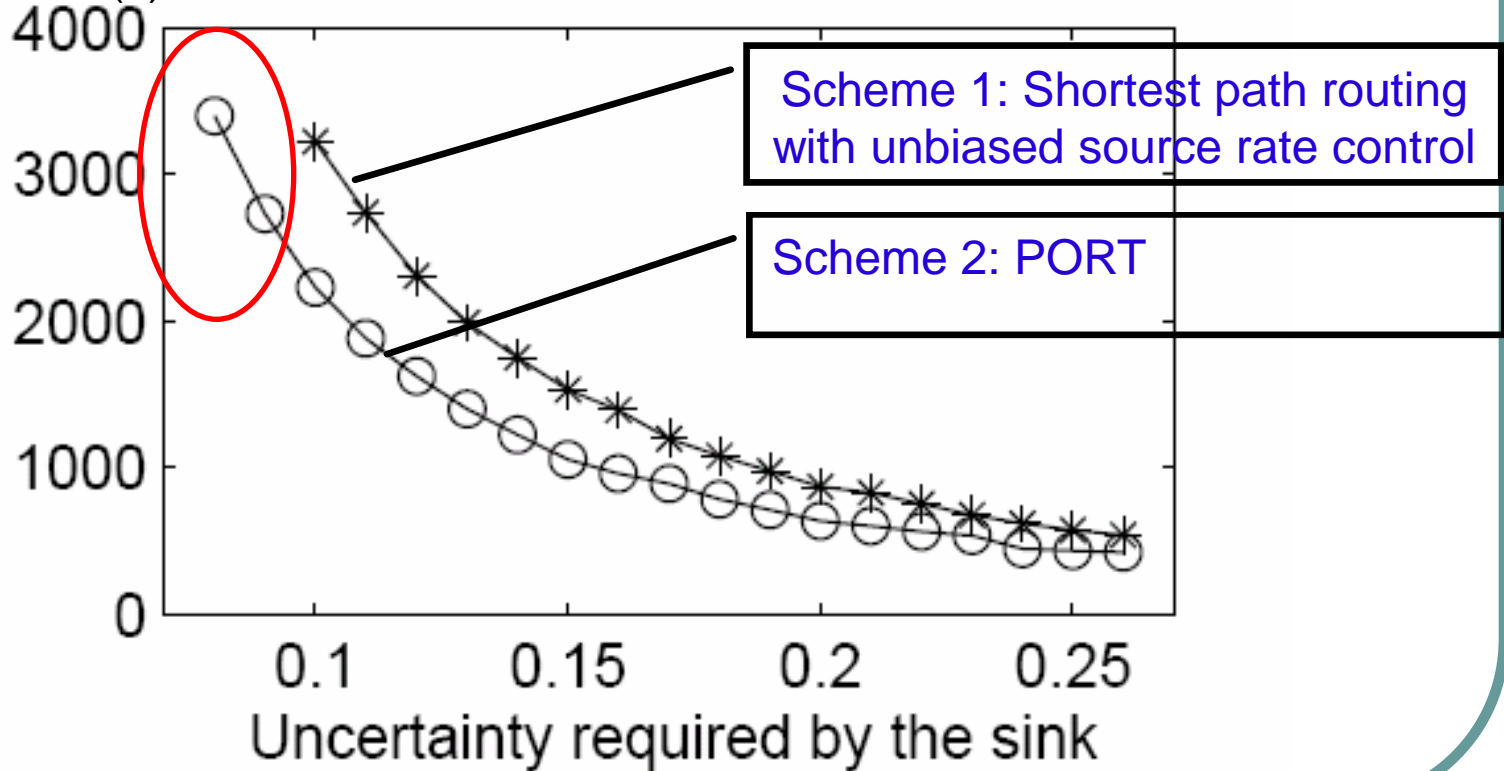


Simulation Results

● Results

Total Energy Consumption (J)

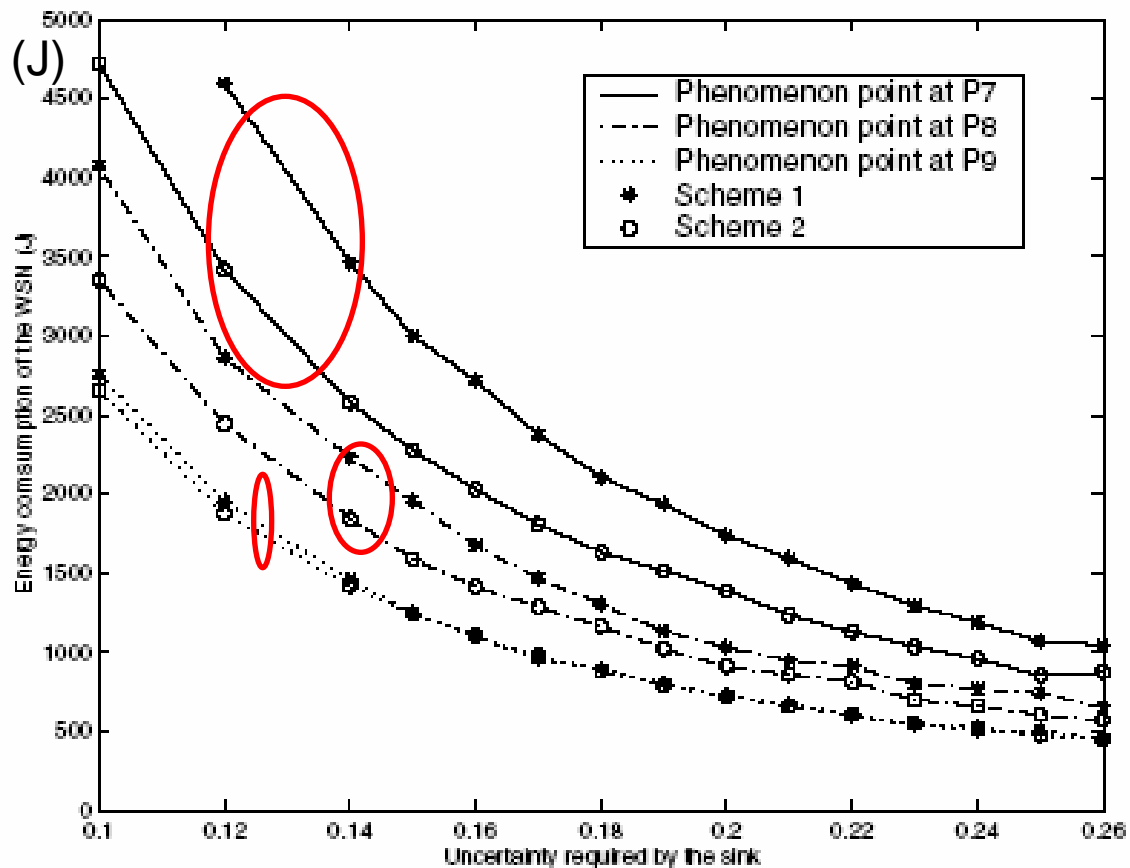
Phenomenon Point at P6



Simulation Results

● Results

Total Energy Consumption (J)



Presentation Outlines

- 1. Introduction
- 2. Motivations and Design Considerations
- 3. Protocol Implementation
- 4. Simulation Results
- 5. **Conclusion**

Conclusion

- PORT optimizes the energy consumptions with two schemes.
 - The sink's optimization scheme that feeds back the optimal reporting rate of each source.
 - A routing scheme for in-network nodes according to the feedback of downstream communication conditions to achieve energy-efficiency and avoid congestion.

PSFQ: A Reliable Transport Protocol for Wireless Sensor Networks

**Chieh-Yih Wan, Andrew Campbell, and
Lakshman Krishnamurthy**

Columbia University, New York

IEEE JOURNAL ON SELECTED AREAS IN
COMMUNICATIONS, VOL. 23, NO. 4, APRIL 2005

What is PSFQ?

- PSFQ =
 - Pump slowly
 - Fetch quickly
- A transport protocol for wireless sensor networks



Purpose of Research

- Closer to a general purpose transport layer for sensor networks
- Not application specific
- Increase data reliability in sensor networks
- Allows for sensor networks that require reliability

Example Problem

- Consider a sensor network in a disaster zone
- Nodes must be re-tasked
- An EEPROM image is sent to all sensor nodes
- No packets can be lost



Protocol Basics

- Pump slowly: Distribute data at a slow speed
- Fetch Quickly: When packet loss occurs, fetch packets from neighbors aggressively
- Similar to a negative acknowledgment system

End to End Recovery

- End to end recovery
 - Error rates increase exponentially
 - If the probability of successfully sending a message across 1 hop is $(1 - p)$, then n hops would have a probability of success of $(1 - p)^n$

Success Rate Versus Network Size

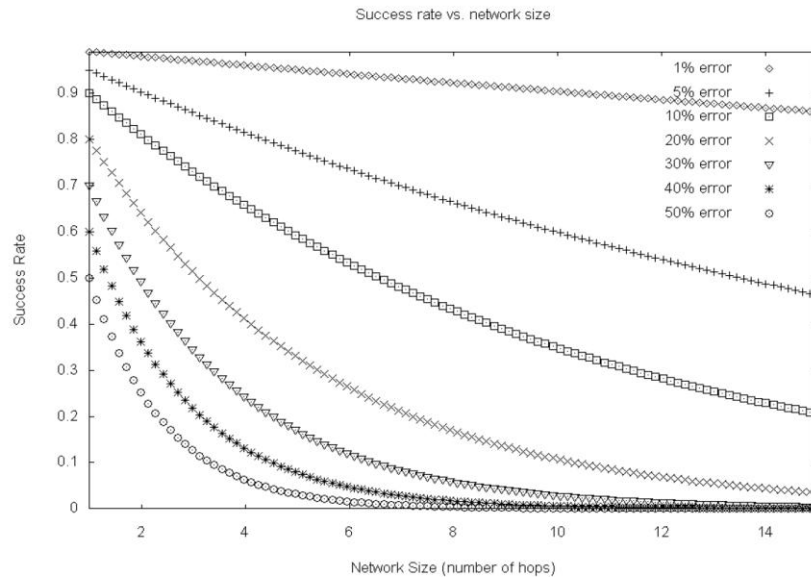


Figure 1. Probability of successful delivery of a message using an end-to-end model across a multi-hop network.

- Error rates can reach five to ten percent (Second and third line)
- With 14 hops, the probability of successful packet being sent is between 30 and 50%!

Hop by Hop

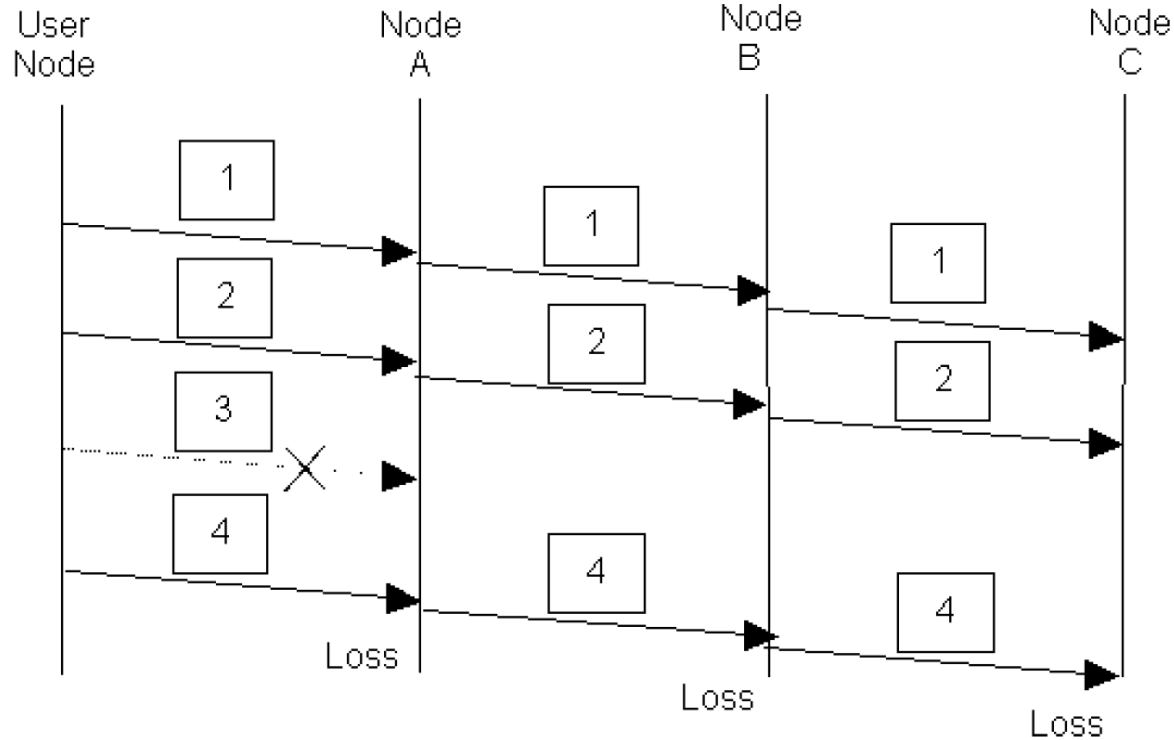
- Probability of packet delivery is $(1 - p)$, regardless of network size
- Non-trivial to determine the “optimal” number of allowable retransmissions

Protocol Specifics

- Three primary functions:
 - Pump operation: Message relaying
 - Fetch operation: Relay-initiated error recovery
 - Report operation: Selective status reporting


Problem of End To End Model: Loss Propagation


- Nodes should only send packets, ordered by sequence number



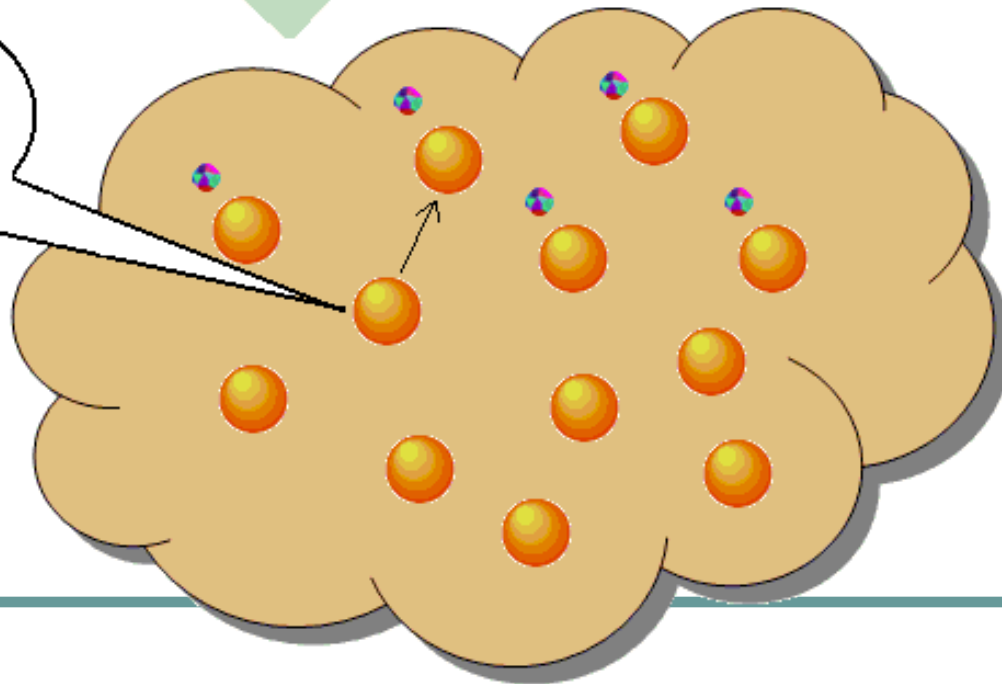
Protocol Idea



 = Packet

 = Sensor node

I did not get
the packet,
fetch!



Pump Operation

- Inject message header
 - File ID
 - File length
 - Sequence number
 - Time to Live (TTL)
- Scheduling accomplished by using 2 timers, T_{\min} and T_{\max} .



Pump Timers

- User nodes (Nodes sending packets) broadcast to others every T_{\min} time units
- Nodes decrease TTL field in header by 1
- If the TTL field is not zero, and packets are in sequence, packets are forwarded
- Packets are sent to neighbours at a random time between T_{\min} and T_{\max}

Rebroadcasts

- Minimize rebroadcasts
- Cut off rebroadcasts after four times
 - Ni, Tseng, Chen, and Sheu show that after 4 rebroadcasts the coverage area increases by at most 0.05%, a minimal benefit

Fetch

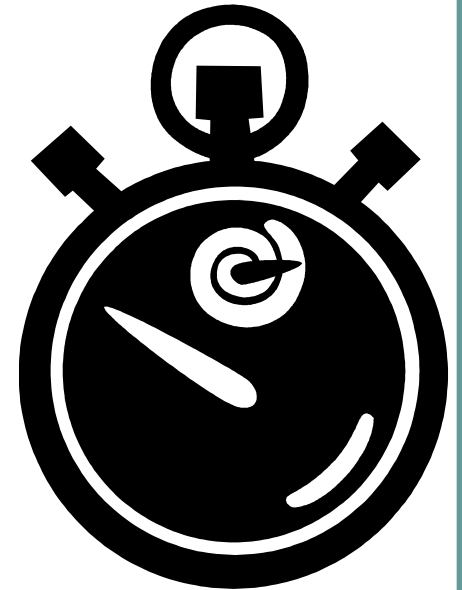
- Fetch tailored to poor link quality, not congestion
- Deal with loss aggregation
 - Concerned with “windows” of lost packets
 - Allow nodes to receive different lost packets from different neighbours

Fetch Nack Messages

- Requests retransmission of a packet
- Three header fields
 - File ID
 - File Length
 - Loss window
- Loss window represents a pair of sequence numbers (t_begin , t_end)
- Example loss window: Sequence (3,5,6,9,11) computes to (4,4), (7,8), and (10,10)

Fetch Timer for Nack messages

- Nodes with missing packets send messages every T_r intervals
- Packets sent every T_r either until:
 - Packet is received, or
 - Threshold is reached
- T_r includes randomization
- Nack messages can go one extra hop(Note: This causes a loss event!!!)
- Authors claim the “extra hop” is rarely needed, but give no probability data.



Proactive Fetch: Last Segments

- With Fetch, nodes are only re-requested (Via Nack) if another node is received with a higher sequence number
- What about the last segment?
- Solution: Check if the last segment has not been received after some T_{pro} time

Computing T_pro

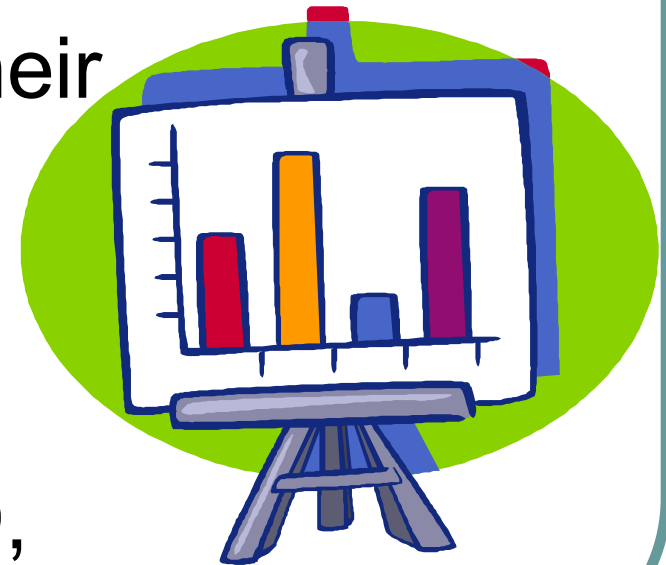
- Consider the length of T_pro
 - Too Big? Causes network delays
 - Too Small? Packet may still be en route.
- Authors suggest the following equation:
- $T_{pro} = a * (S_{max} - S_{last}) * T_{max}$
- $a \geq 1$
- S_{max} = largest sequence number
- S_{last} = last sequence number received
- T_{max} = the time between sent messages from the original sending node

More on T_pro

- Suppose that a node has a small data cache
- In this case, the authors suggest this equation for T_pro
- $T_{pro} = a * n * T_{max}$ ($a \geq 1$)
- n = number of segments kept in the data cache
- This equation is to allow nodes to proactively fetch every n segments

Report operation

- Cost of every node sending short reporting messages is high
- Nodes can “piggy back” their own messages
- Header contains only the destination ID
- Report payload: <Node ID, Sequence Number> pairs



Report Process

- Only the last hop, identified by TTL packet = 0, initially responds to the report call
- Last hop sends a message to its parent, source of request for a report, with the report payload
- Time to wait before sending a report message is $T_{\text{report}} = T_{\text{max}} * \text{TTL} + \text{delta}$
- Delta = random time between 0 some real number

Report Process (cont..)

- Suppose a node receives a “full” report.
- A full report is one where no more data can be appended to the report
- Nodes do not add to a report if their unique ID is included in the message.
- **My own analysis:**
 - This implies nodes MUST search for their ID
 - If the data is sorted, this is a $O(\log n)$ search
 - If the data is unsorted, this implies a $O(n)$ search for each sensor node, yikes!

Performance Evaluation

- Compare PSFQ to Scalable Reliable Multicast (SRM), which is traditionally for IP networks
- Idealized version of SRM is compared, with “omniscient” routing
- Metrics used are:
 - Average delivery ratio – basically error tolerance
 - Average latency - delays
 - Average delivery overhead – sent vs. received messages, essentially communication cost

Performance Evaluation (cont..)

- Metrics studied vs. Channel error rate and network size
- Simulation of a disaster situation in building, nodes have 2Mbps bandwidth.
- 50 Packets sent
- $T_{\max} = 100$ milliseconds(ms)
- $T_{\min} = 50$ ms
- $T_r = 20$ ms(recall that T_r is the fetch time)
- Packets transmitted from user node every 10 ms

Study Idea

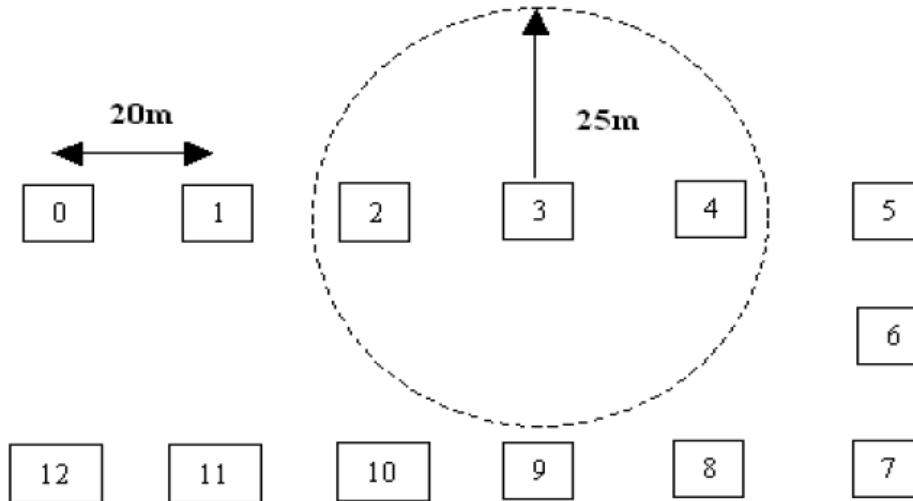
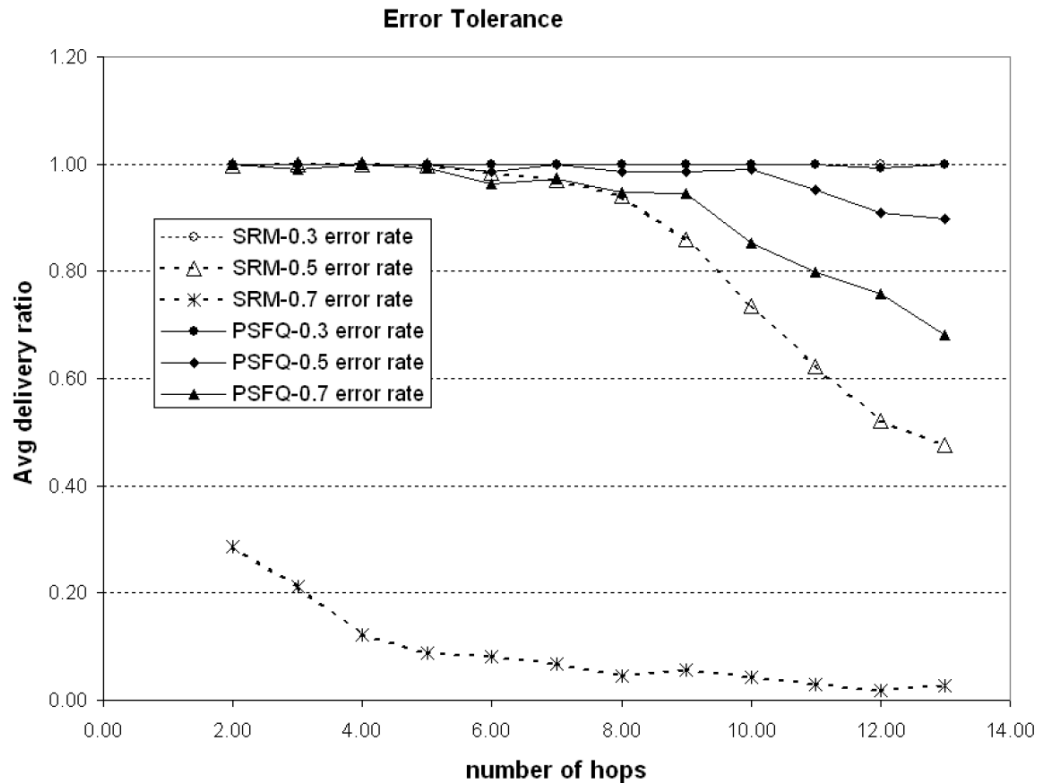


Figure 4. Sensor network in a building. A user node at location 0 injects 50 packets into the network within 0.5 seconds.

Simulation Results



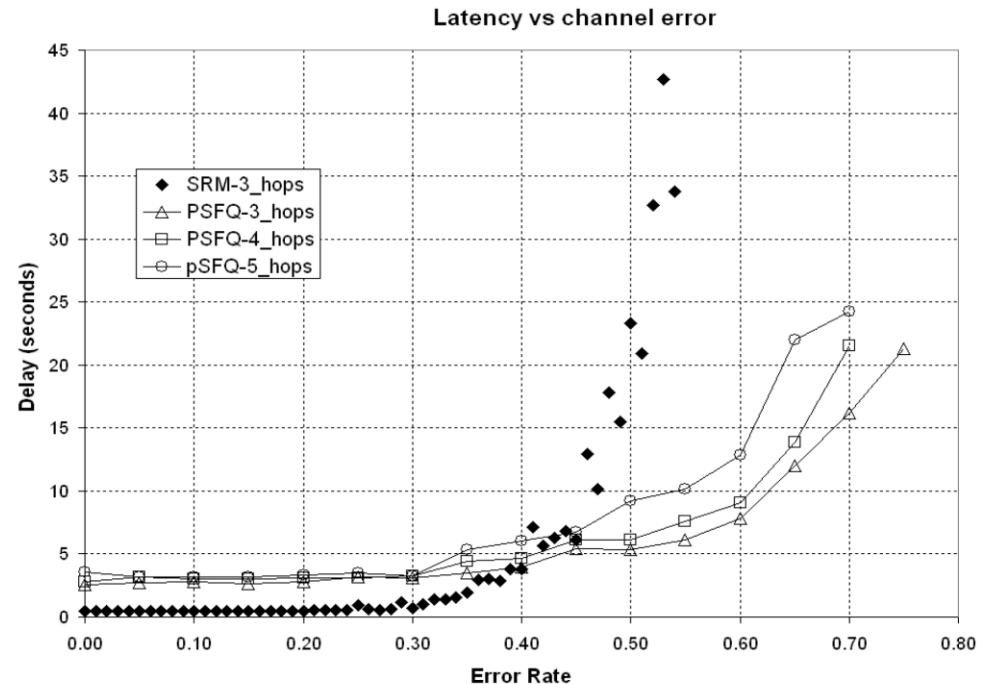
- Error rate is roughly 30%
- The authors suggest even in military applications sensor networks have only 5-10% error rate, yet no data is shown for this class.

Simulation Results (cont..)

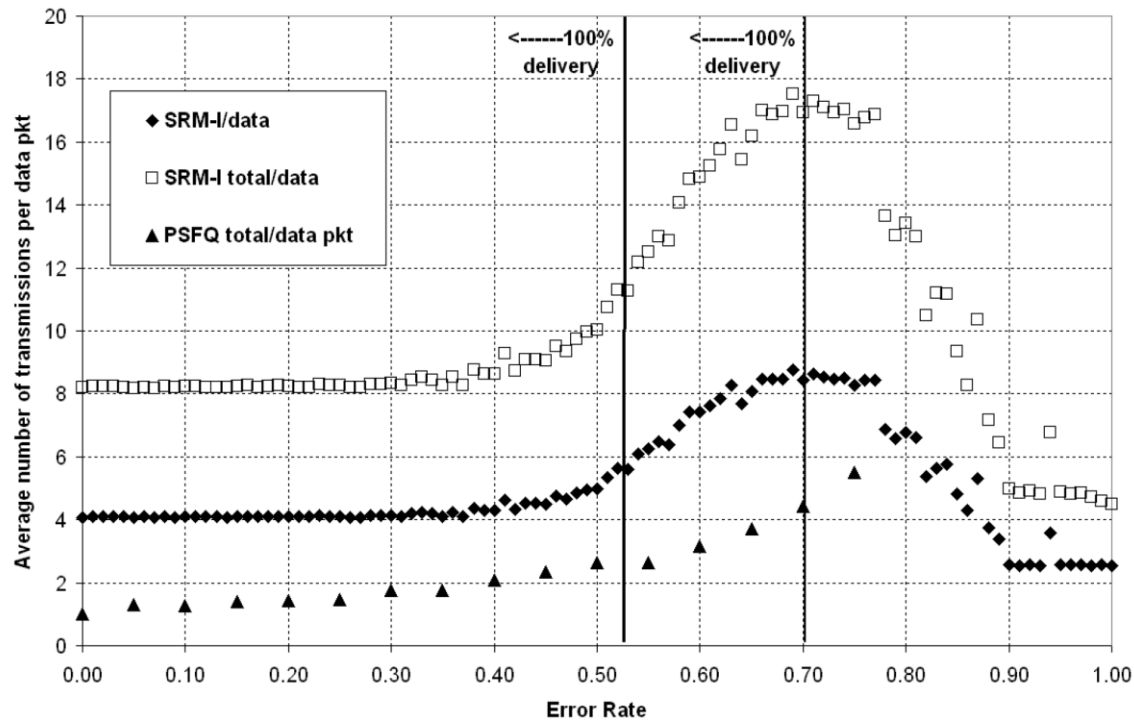
- Note that SRM-I (Idealized SRM) performs better for error conditions below approximately 40 % but increases exponentially after 40%.

- If error rates are typically 5-10%, this would indicate that for most cases PSFQ is actually “less” effective than SRM-I, which the authors “forget” to mention

- Thus, only use PSFQ if error rates are extremely poor

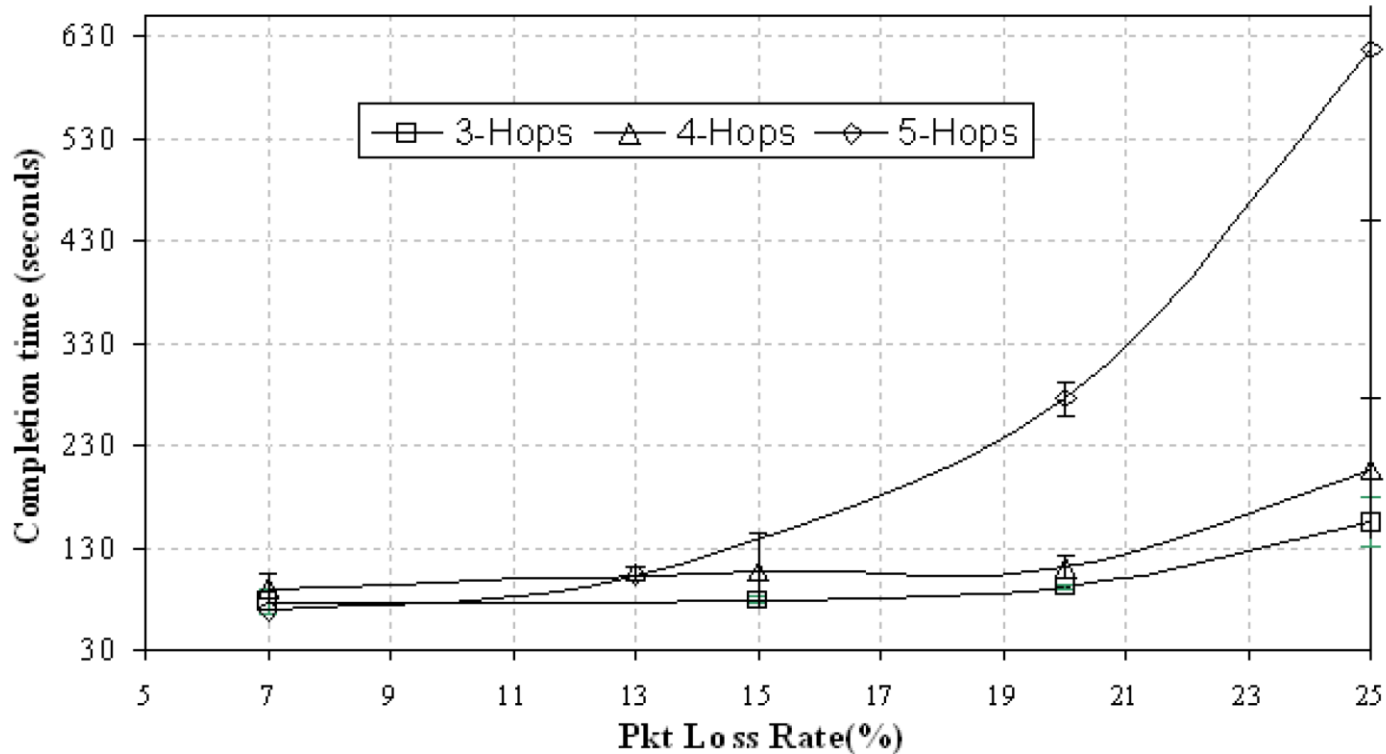


Communication Overhead Simulation



- In this experiment, however, note that PSFQ has a significantly lower communication overhead
- No obvious confounds were found in this simulation
- Even for small error rates, the communication cost is lower for PSFQ

Final Experiment



- Incorporate PSFQ into RENE motes (A sensor platform) using TinyOS.
- Results essentially indicated that real world results do not match the simulations exactly

Conclusion

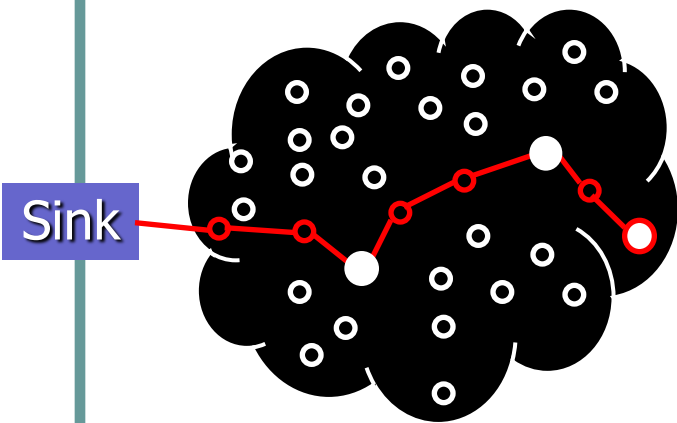
- Pump slowly Fetch Quickly(PSFQ) is a transport protocol for sensor networks
- PSFQ seems to work, in simulations, for extremely high error rate systems
- PSFQ seems comparable, or slightly worse, than Scalable Reliable Multicast
 - For low error rate systems
 - For some simulations
- Problems with internal/external validity

Reliable Multi-Segment Transport (RMST)

Fred Stann, John Heidemann
USC/Information Sciences Institute

IEEE International Workshop on Sensor Net Protocols and Applications
(SNPA) 2003

Reliable Multi-Segment Transport (RMST)



- End-to-end data-packet transfer reliability
- Each RMST node caches the packets
- When a packet is not received before the so-called WATCHDOG timer expires, a NAK is sent backward
- The first RMST node that has the required packet along the path retransmits the packet
- In-network caching brings significant overhead in terms of power and processing
- Relies on Directed Diffusion Scheme

RMST Node

Source Node



RMST

Placement of reliability for data transport

- RMST considers 3 layers
 - MAC
 - Transport
 - Application
- Focus is on MAC and Transport

RMST

MAC Layer Choices

- No ARQ
 - All transmissions are broadcast
 - No RTS/CTS or ACK
 - Reliability deferred to upper layers
 - Benefits: no control overhead, no erroneous path selection
- ARQ always
 - All transmissions are unicast
 - RTS/CTS and ACKs used
 - One-to-many communication done via multiple unicasts
 - Benefits: packets traveling on established paths have high probability of delivery
- Selective ARQ
 - Use broadcast for one-to-many and unicast for one-to-one
 - Data and control packets traveling on established paths are unicast
 - Route discovery uses broadcast

RMST

Transport Layer Choices

- **End-to-End Selective Request NACK**
 - Loss detection happens only at sinks (endpoints)
 - Repair requests travel on reverse (multihop) path from sinks to sources
- **Hop-by-Hop Selective Request NACK**
 - Each node along the path caches data
 - Loss detection happens at each node along the path
 - Repair requests sent to immediate neighbors
 - If data isn't found in the caches, NACKs are forwarded to next hop towards source

RMST

Application Layer Choices

- **End-to-End Positive ACK**
 - Sink requests a large data entity
 - Source fragments data
 - Sink keeps sending interests until all fragments have been received
 - Used only as a baseline

Motivation

- Need for Reliability
- Reliability at MAC, Transport Layer, Application Layer
- Hop-by-Hop Recovery OR End-to-End Recovery

MAC Layer Design Choices

- ❑ Automatic Repeat Request (ARQ) for hop-by-hop recovery
 - ❑ RTS/CTS, ACK is used to support ARQ
 - ❑ 802.11 uses ARQ for unicast and for BCast it does NOT use ARQ
- ❑ No-ARQ
 - ❑ No Reliability
 - ❑ Higher Layers responsible for Reliability
- ❑ Selective ARQ
 - ❑ Use ARQ for unicast on established paths
 - ❑ No-ARQ for broadcast messages like route discovery

Transport Layer Design Issues

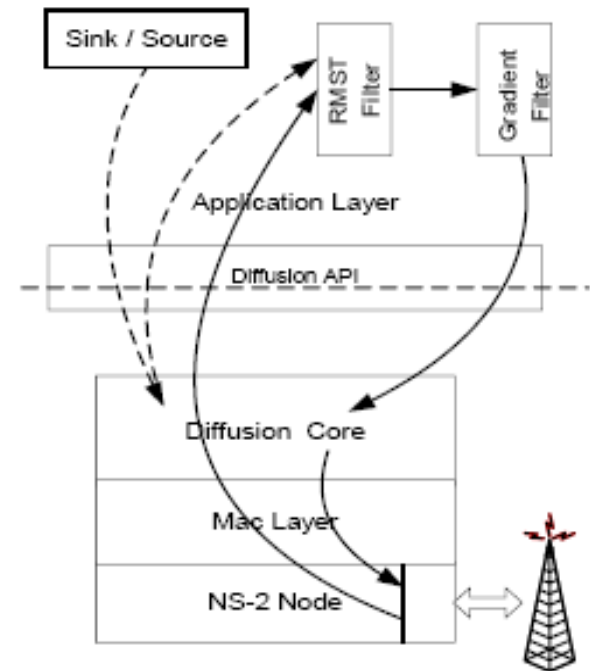
- End-to-End Selective Request NACK
- Hop-by-Hop Selective Request NACK
- Cache/Non-Cache Mode

Application Layer Design for Reliability

- End-to-End Positive ACK
 - Tracks arrived fragments at the sink node

RMST: Reliable Multi Segment Transport

- To run in conjunction with Directed Diffusion
 - Reinforced paths for data transfer
 - In case of node failures/link quality degradation new path is reinforced
 - RMST is a transport layer over directed diffusion routing protocol



RMST Overview

- Reliability: Eventual delivery of all fragments to all interested sinks
- No real-time guarantee
- Two Key goals
 - Effective management of fragmentation/reassembly
 - Guaranteed delivery

Mechanism

- Caching Mode
 - Intermediate node can detect a hole and make request (NACK) to upstream node on the reinforced path
 - Request propagates till missing fragment is found in the cache
- Non Caching Mode
 - Sink detects losses

Multiple holes are aggregated in NACK

Each cache node maintains fragment map and hole map

Flow id is used to track fragments by flow id, fragment id

MAC Layer Retries - Analysis

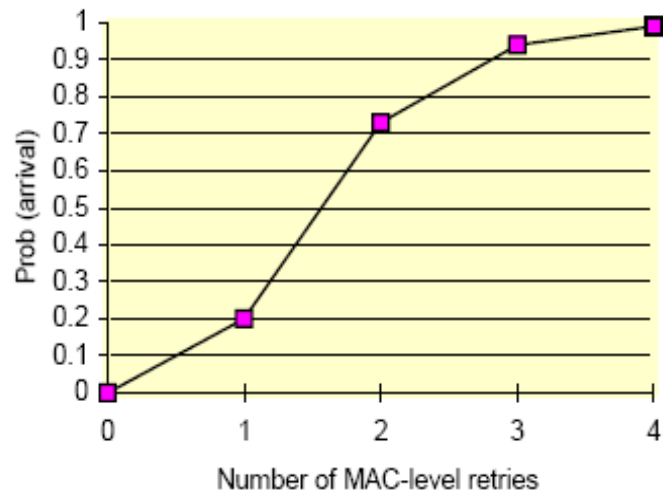


Figure 1: Probability of arrival across 40 hops with an average error rate of .10 per hop, given R retries per hop.

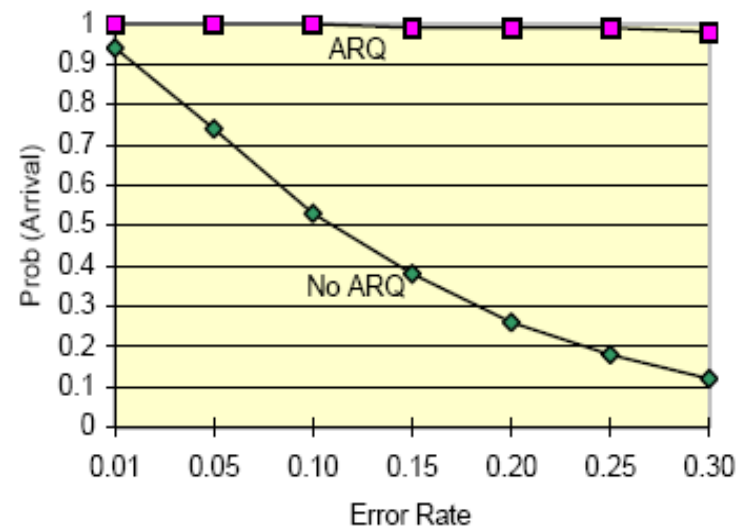


Figure 2: Probability of arrival across 6 hops

Transport Layer: Hop-by-Hop Vs End-to-End Analysis

# Fragments	5 Hops	10 Hops
5	27.77 / 42.33	55.55 / 143.39
10	55.55 / 84.67	111.11 / 286.79
20	111.11 / 169.35	222.22 / 573.59

Table 1: Number of total transmissions required to send M fragments across N hops (with cache/without cache)

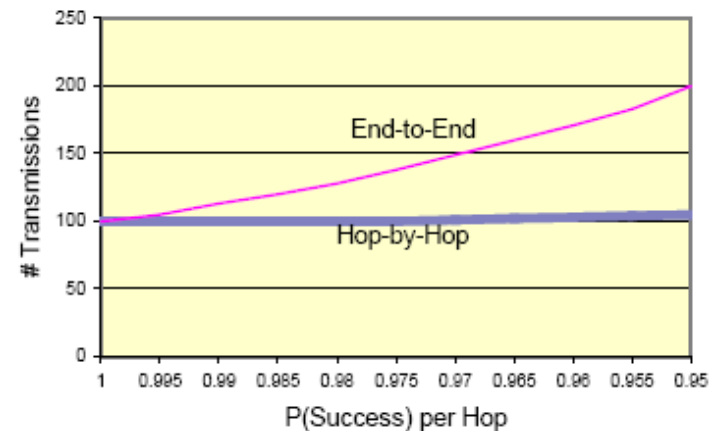


Figure 3: Number of transmissions required to send 10 fragments across 10 hops. Hop-by-hop vs. End-to-End repair.

RMST Evaluation

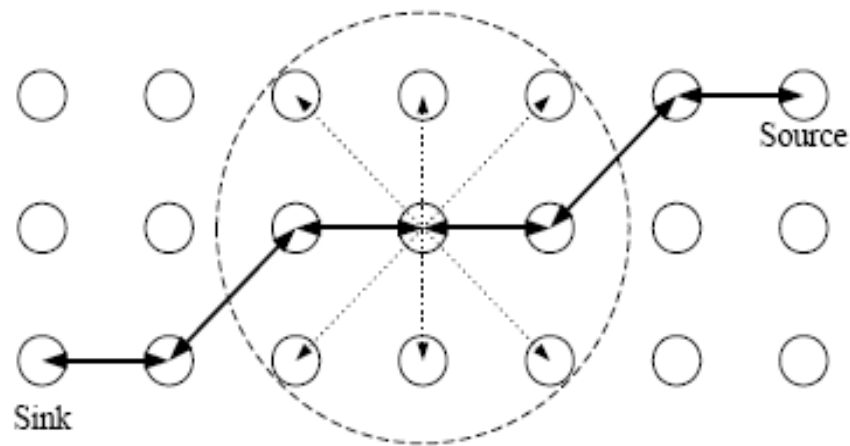


Figure 4

Layout of test grid showing radius of central node,
reinforced path from source to sink and back-channel

Baseline Case: No Transport but only standard directed diffusion

PHY Error Rate	No ARQ	ARQ All	Selective ARQ
0	.93 (.07)	.57 (.03)	.65 (.03)
.01	.51 (.04)	.56 (.03)	.61 (.05)
.10	.21 (.05)	.47 (.09)	.54 (.06)

Table 2: End-to-End Positive ACK
Normalized byte transmissions required for diffusion to transfer 50 fragments of 100 bytes across 6 hops without any transport layer

RMST: Hop-by-Hop Recovery and Caching

PHY Error Rate	No ARQ	ARQ All	Selective ARQ
0	.99 (.05)	.60 (.06)	.68 (.06)
.01	.95 (.06)	.57 (.06)	.67 (.07)
.10	.76 (.07)	.48 (.07)	.61 (.07)

Table 3: Hop-by-Hop Selective NACK and Caching
Normalized byte transmissions required for diffusion to
transfer 50 fragments of 100 bytes across 6 hops
with hop-by-hop caching and repair

RMST: End-to-End Recovery – Non Caching Mode

PHY Error Rate	No ARQ	ARQ All	Selective ARQ
0	1.0 (.05)	.61 (.08)	.67 (.07)
.01	.90 (.06)	.60 (.10)	.66 (.07)
.10	n/c	.49 (.09)	.61 (.07)

Table 4: End-to-End Selective NACK

Total byte transmissions required for diffusion to transfer 50 fragments of 100 bytes across 6 hops with end-to-end repair.

Performance Under High Loss Condition

PHY Error Rate	Hop by Hop RMST NoARQ	Hop by Hop RMST Sel ARQ	End to End RMST Sel ARQ
.20	.48 (.19)*	.40 (.18)	.40 (.17)
.30	n/c	.24 (.23)	.27 (.25)

Table 5: High Error Rate Test

Total byte transmissions required for diffusion to transfer 50 fragments of 100 bytes across 6 hops with high error rates.

Conclusions

- Reliability support at Transport and MAC Layer
- Selective ARQ is recommend at MAC Layer
- NACK based Transport layer in tandem with Selective ARQ support at MAC layer

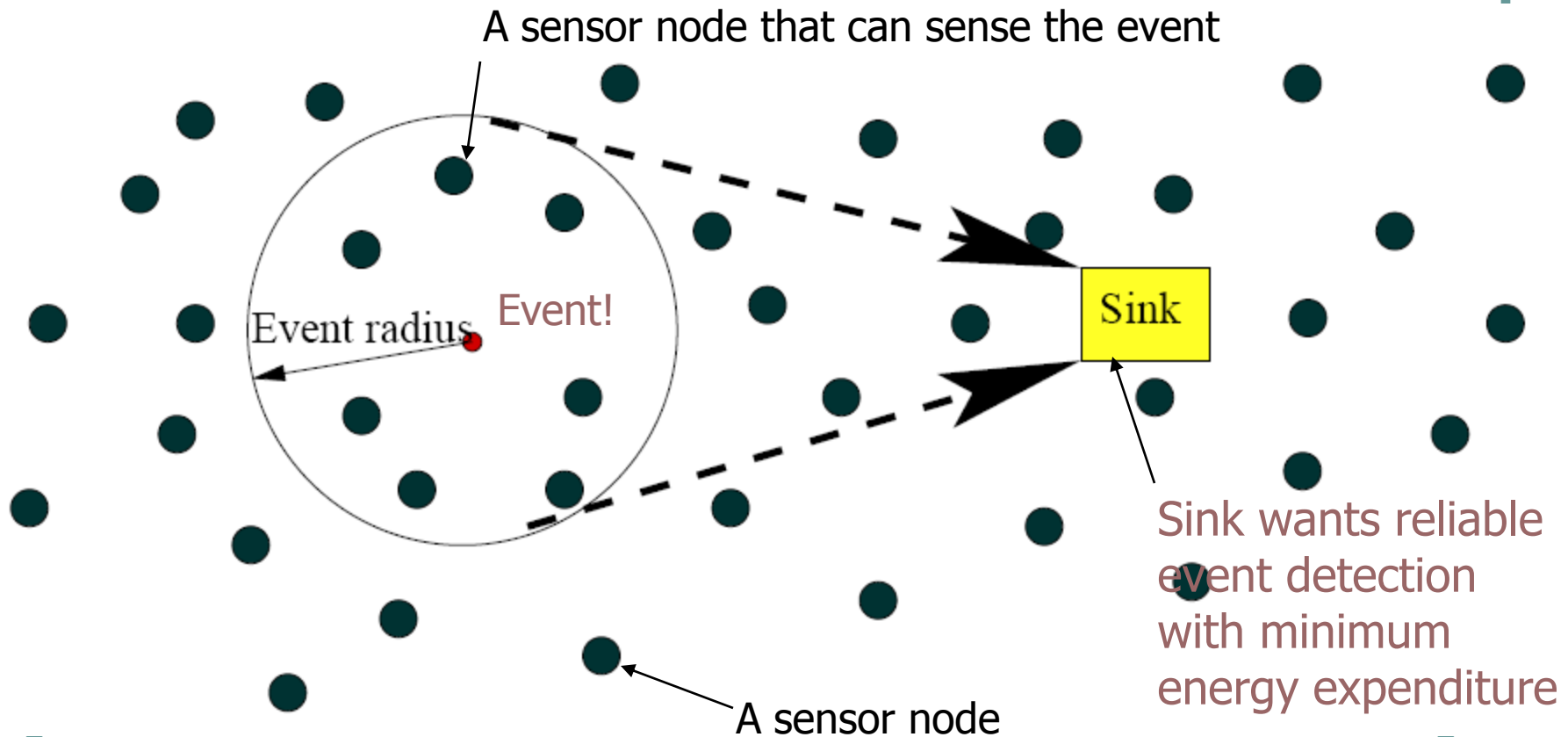
ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks

Özgür B. Akan, and Ian F. Akyildiz,

Georgia Institute of Technology,

IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 13, NO.
5, OCTOBER 2005

Event Detection in a WSN



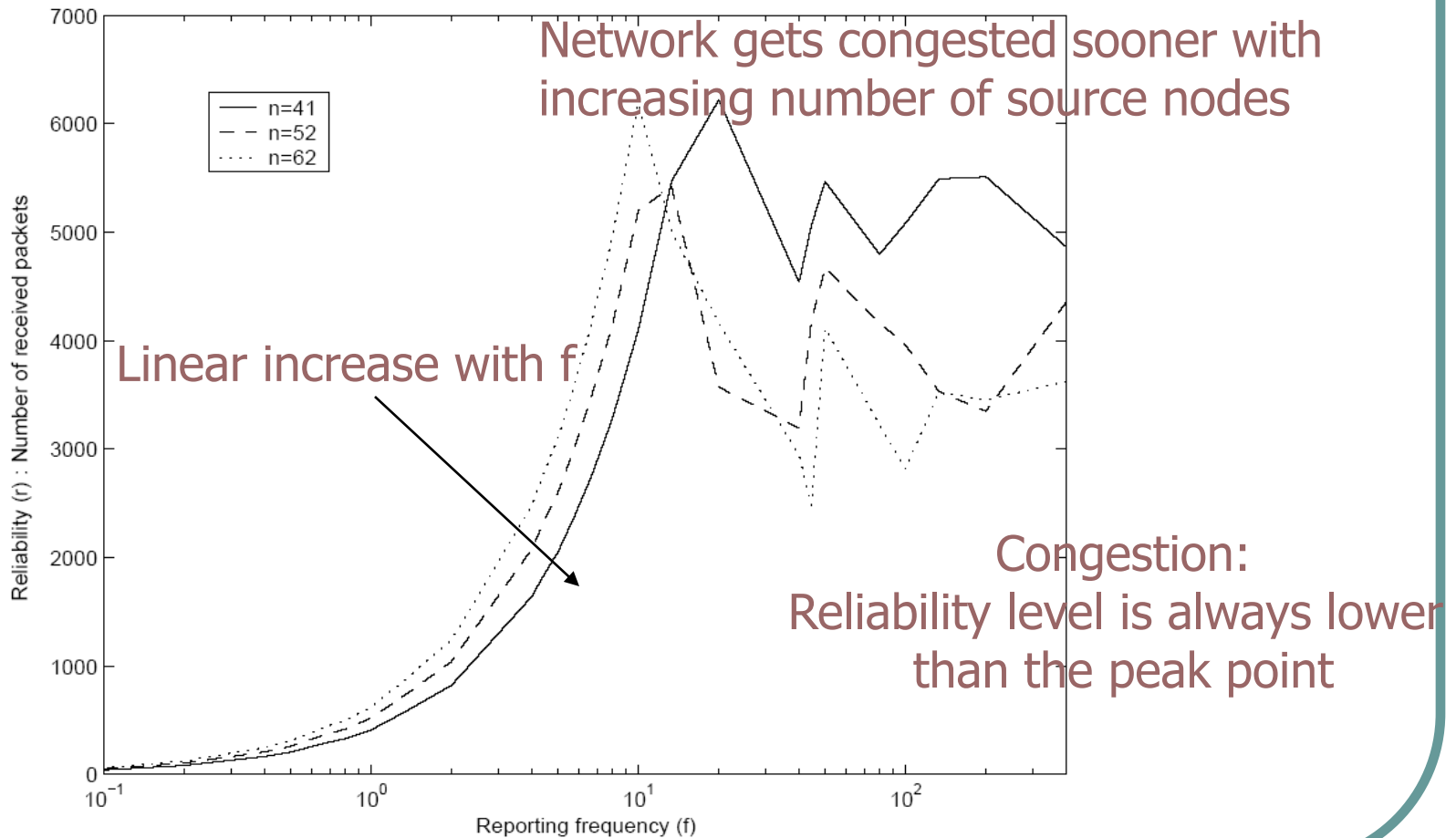
Motivation

- A sink is only interested in the **collective information** from a number of source nodes and not in individual sensor reports
- **Event-to-sink communication**
 - Different from traditional notion of end-to-end communication
- Energy-efficient
- Congestion resolution

Problem Statement

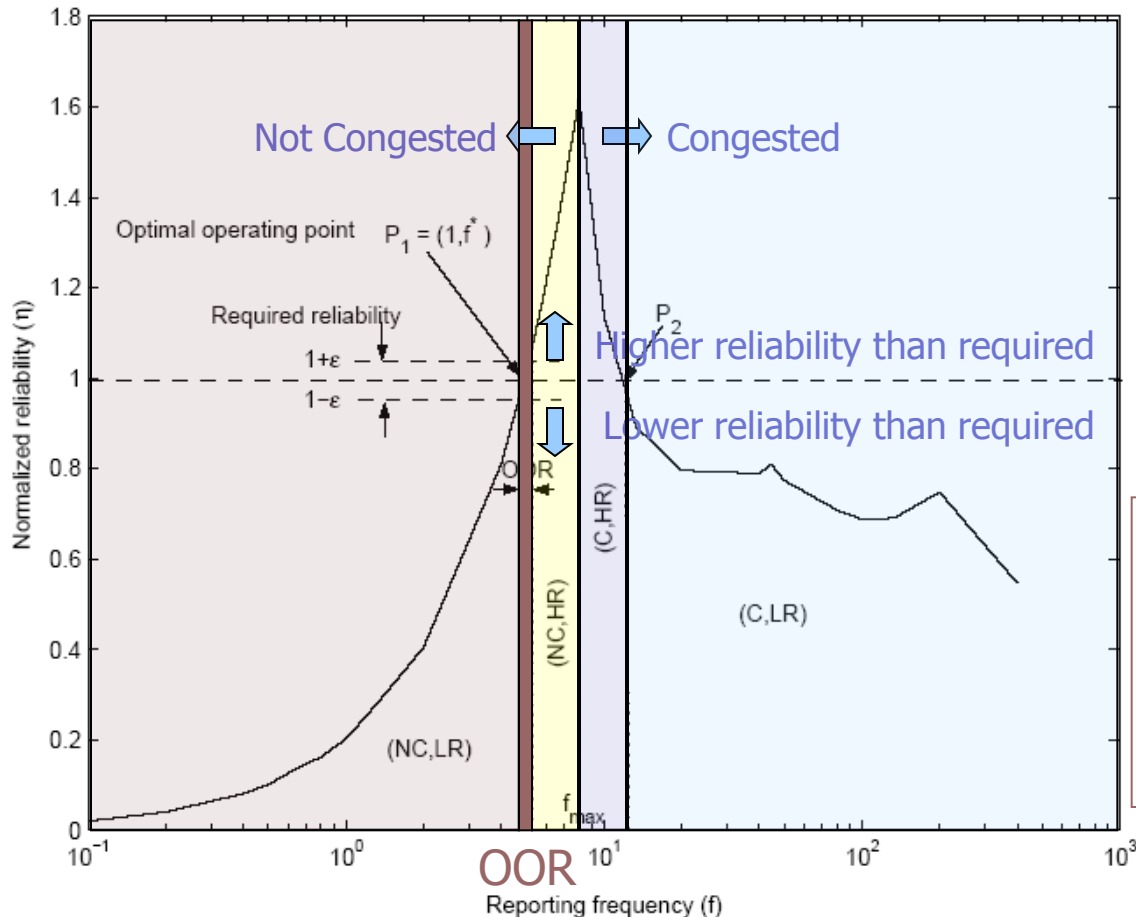
- To configure the **reporting rate** of source nodes so as to achieve the required event detection **reliability** at the sink with **minimum resource utilization**
- Also resolve congestion

Typical Behavior at a Sink



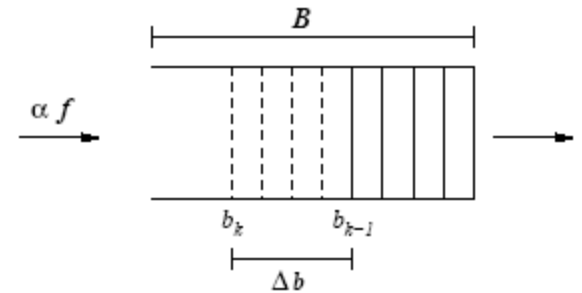
Five characteristic regions

$$\eta = \frac{r}{R}$$



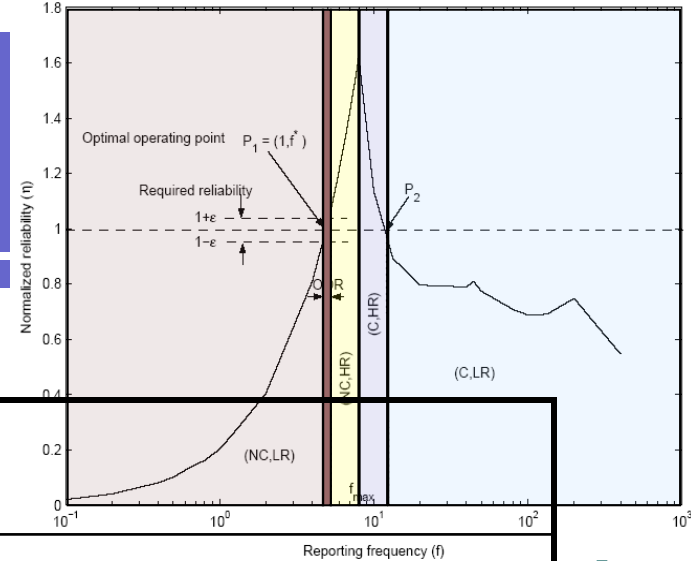
Goal:
To stay in OOR
where energy
expenditure is
optimal

Congestion Detection



- Congestion status is required at the sink to determine the network state
- Based on expectation of buffer overflow at sensor nodes
 - During a single interval, f and n do not change much
- If pending congestion is detected CN bit is set in event reports

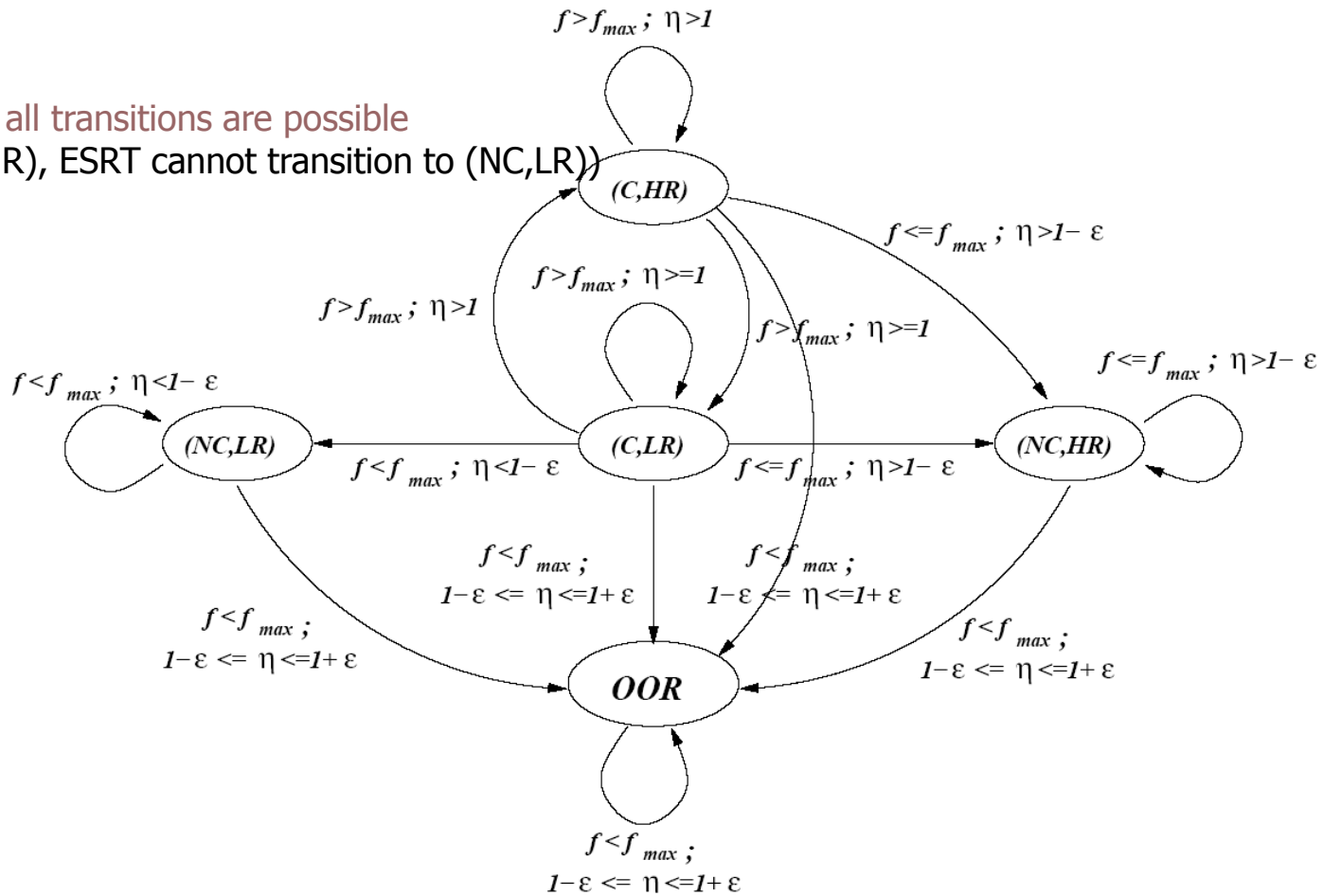
ESRT Actions



Network State	Action
(NC,LR)	Multiplicatively increase f Achieve required reliability ASAP
OOR	Stay
(NC,HR)	Decrease f conservatively Cautiously reduce energy consumption while not compromising reliability
(C,HR)	Decrease f carefully but aggressively to (NC,HR) to relieve congestion Then, follow (NC,HR) behavior
(C,LR)	Decrease f exponentially to relieve congestion ASAP

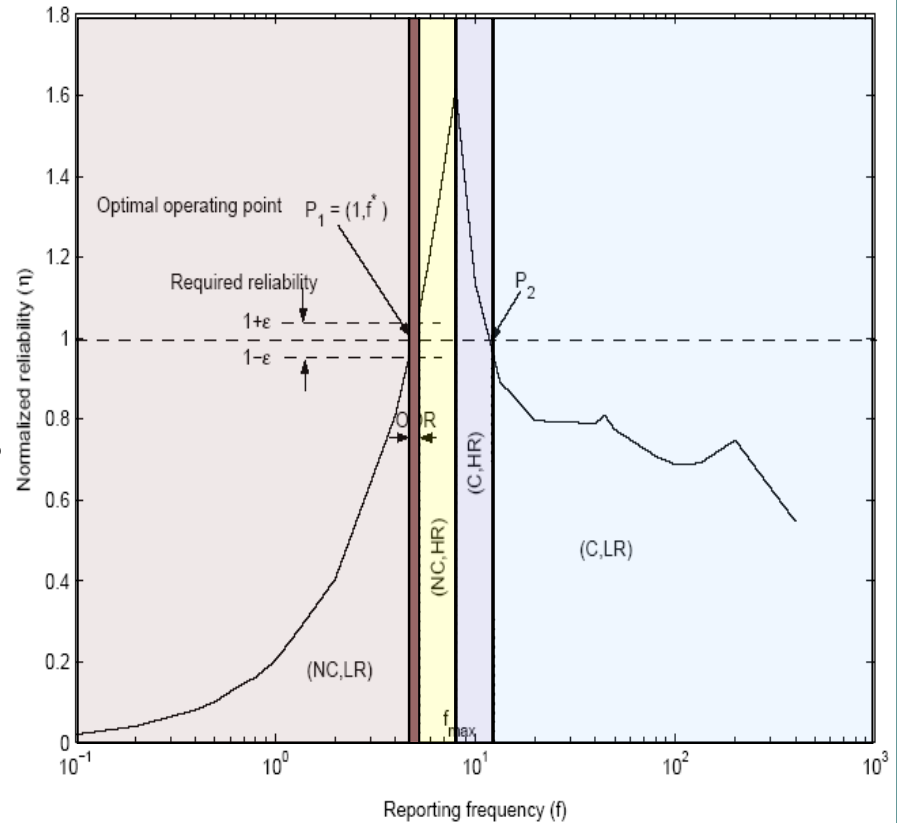
ESRT State Diagram

Not all transitions are possible
 (e.g. From (C,HR), ESRT cannot transition to (NC,LR))



Stability of ESRT

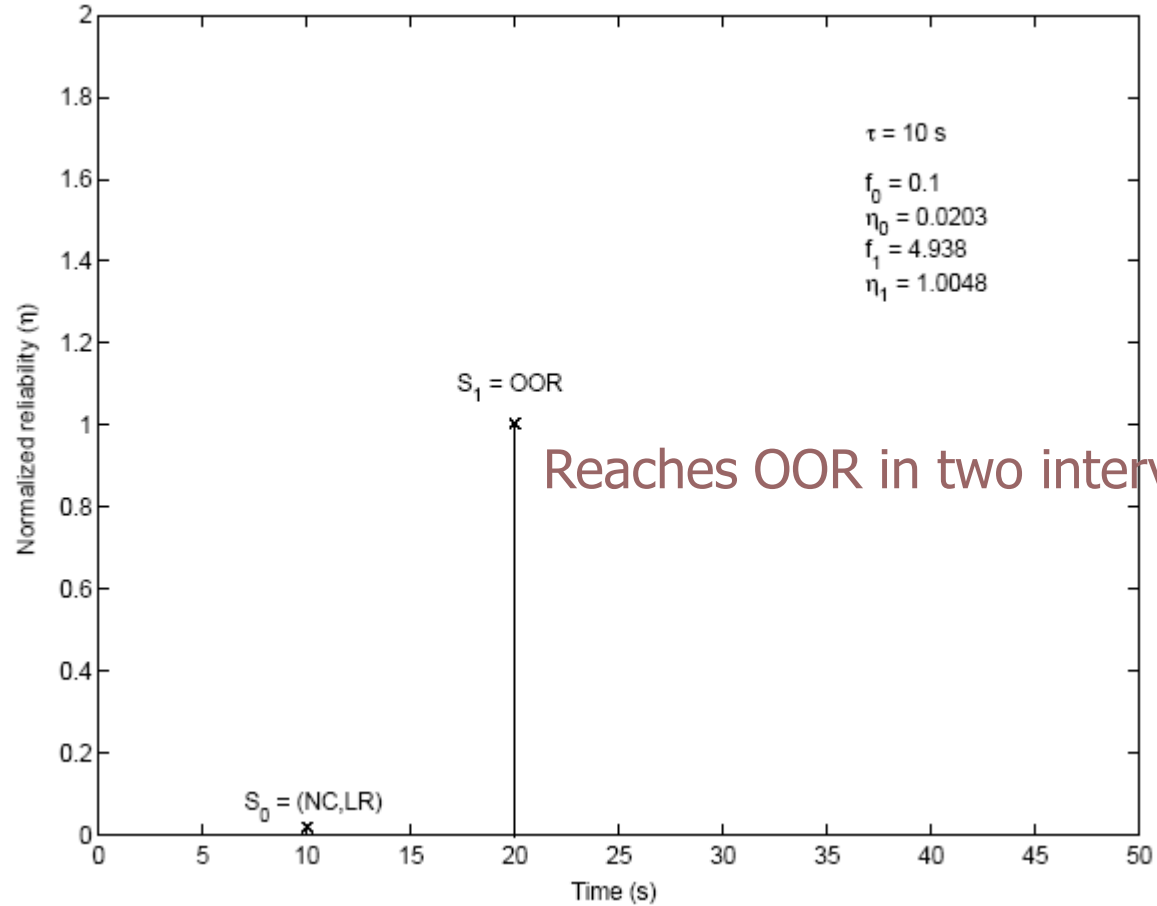
- ESRT converges to OOR from any of four initial states $\{(NC,LR), (NC,HR), (C,HR), (C,LR)\}$
- From (NC,HR) , ESRT stays in the state until converges to OOR
 - Convergence time depends on ε – smaller ε causes longer convergence time



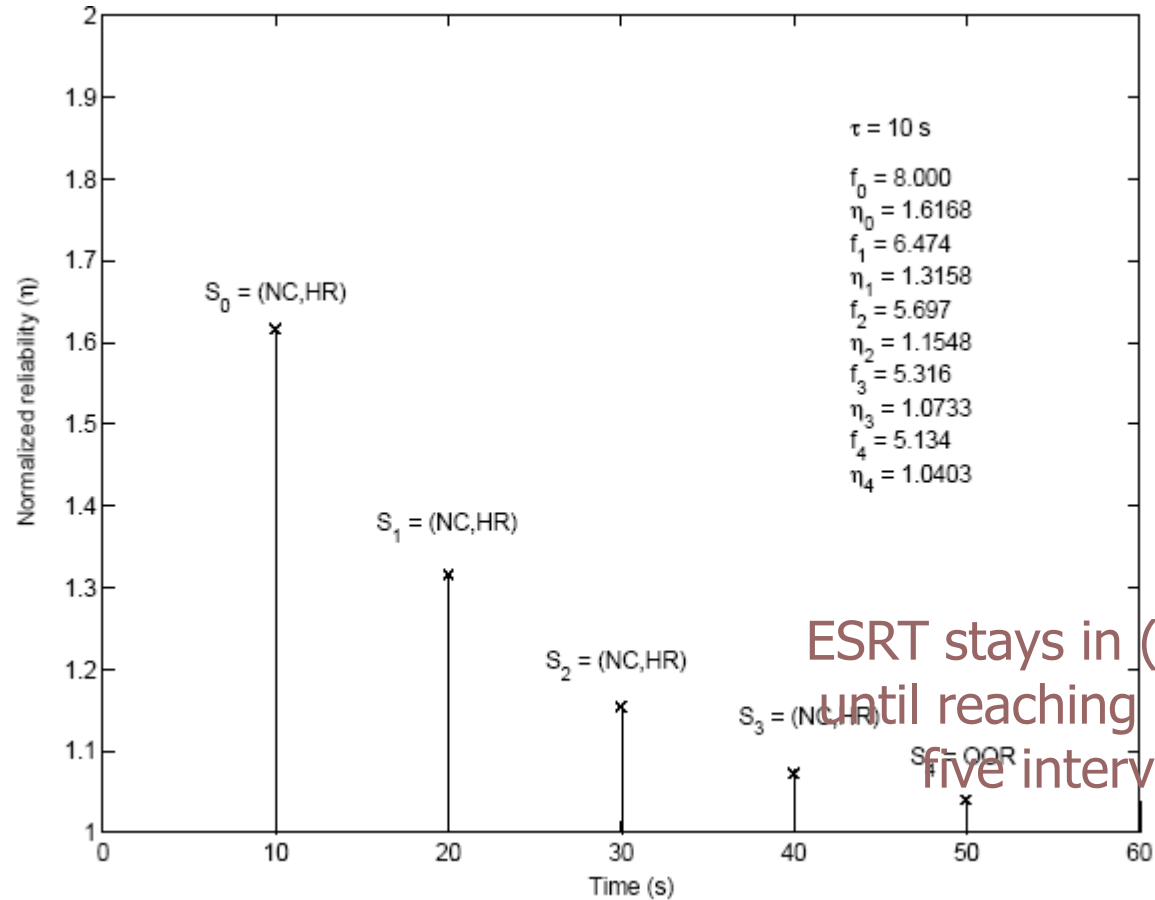
Simulation Setup

- Ns-2 simulator
- 200 sensor nodes
- 100m x 100m area
- 40m transmission range
- 30 byte packets
- 65 packets IFQ
- 10 sec decision interval (τ)

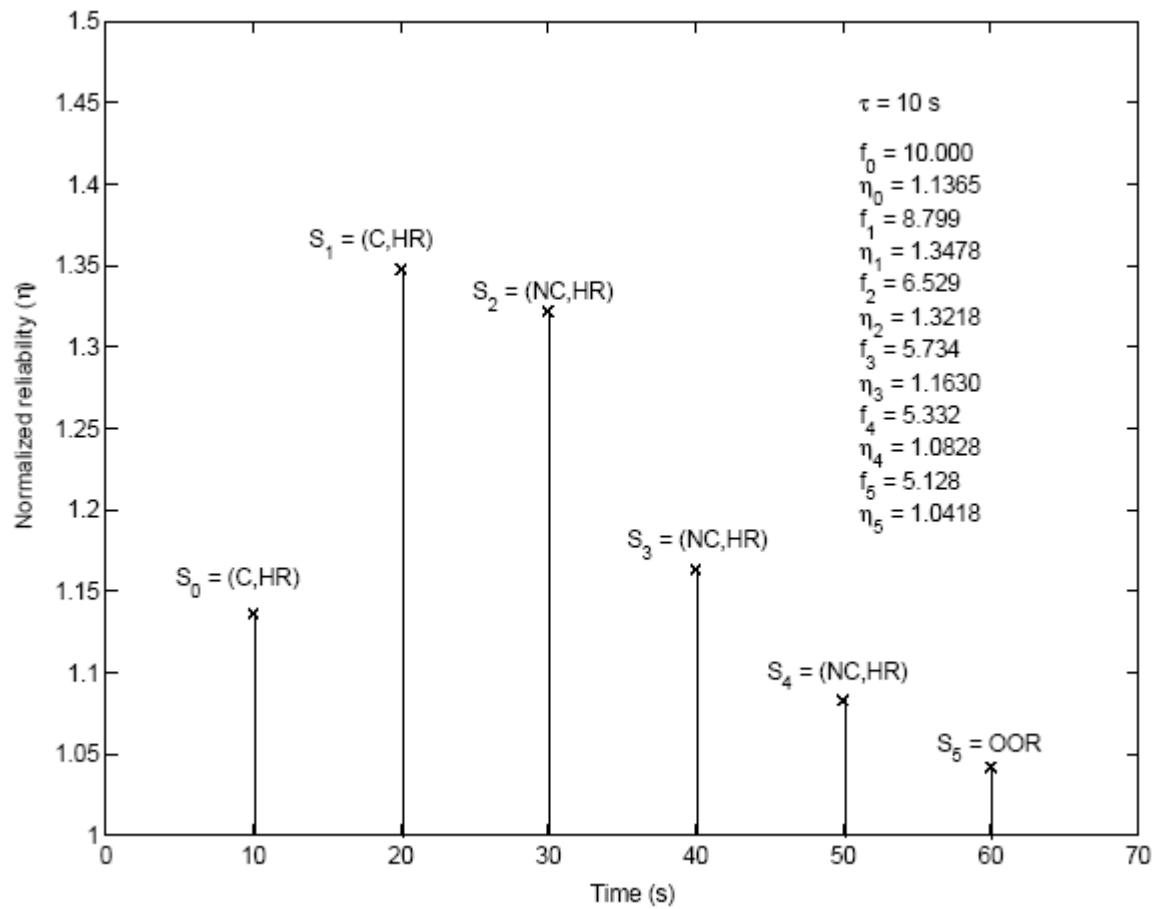
From (NC,LR)



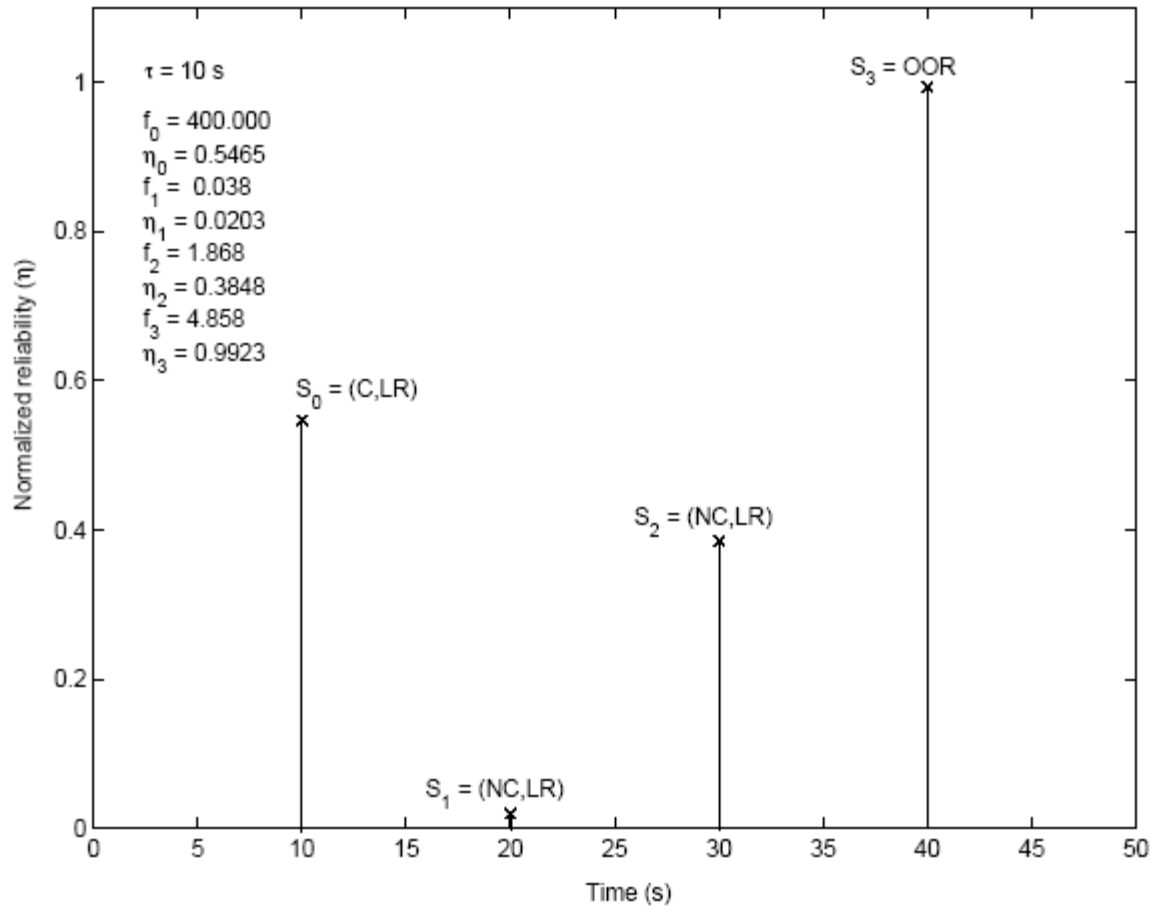
From (NC,HR)



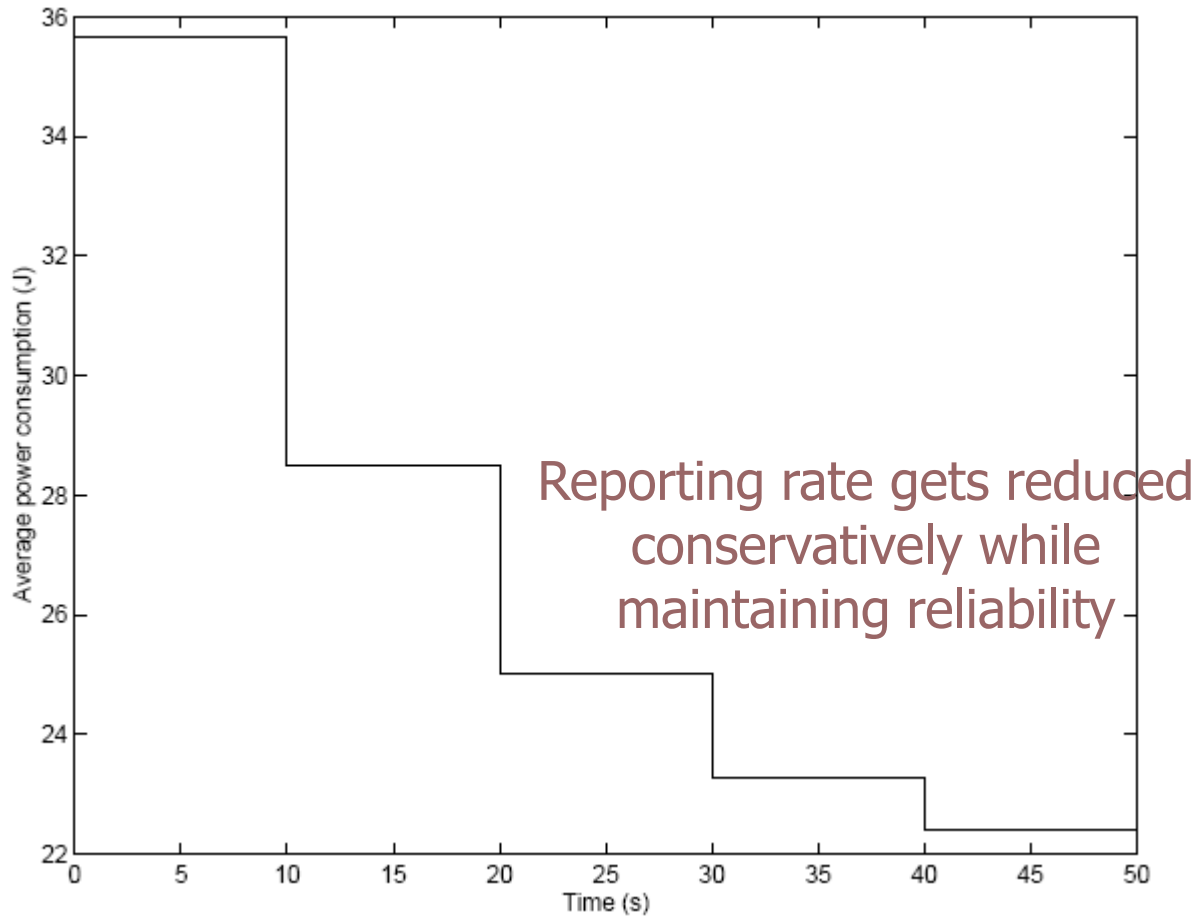
(C,HR) to (NC,HR) then OOR



(C,LR) to (NC,LR) then OOR



Power savings from (NC,HR)



Conclusion

- ESRT provides a reliable event-to-sink communication
 - Self-configuration
 - Energy awareness
 - Uses minimum energy while achieving required reliability
 - Congestion control
 - Collective identification
 - Individual sensor ID is not necessary
 - Biased implementation
 - Almost entirely in sink