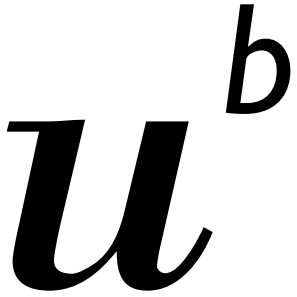


b
UNIVERSITÄT
BERN

UNIVERSITÄT BERN

Internet of Things

IX. Transport Protocols

Prof. Dr. Torsten Braun, Institut für Informatik

Bern, 26.04.2021 – 03.05.2021

UNIVERSITÄT BERN

Internet of Things: Medium Access Control

Table of Contents

1. Introduction

1. Reasons for Packet Loss

Packet error rate in low-power sensor network may be larger than 10 % because of

- Low signal strength
- High bit error rates (noisy channels)
- Simultaneous transmissions (contention for shared medium)
- Self-interference / reflection / multi-path propagation
- Queue overflow

1. Introduction

2. Transport Protocol Functions

- Transport protocols are used to
  - mitigate congestion
  - reduce and recover from packet loss
  - provide end-to-end reliability
- Congestion Control
  - Congestion detection
  - Congestion notification
  - Rate adjustment
- Error Control
  - Loss detection
  - Loss notification
  - Retransmission
- Congestion and error control are strongly interrelated:
  - Congestion might cause packet loss leading to (immediate) retransmission and additional network load as well as collisions

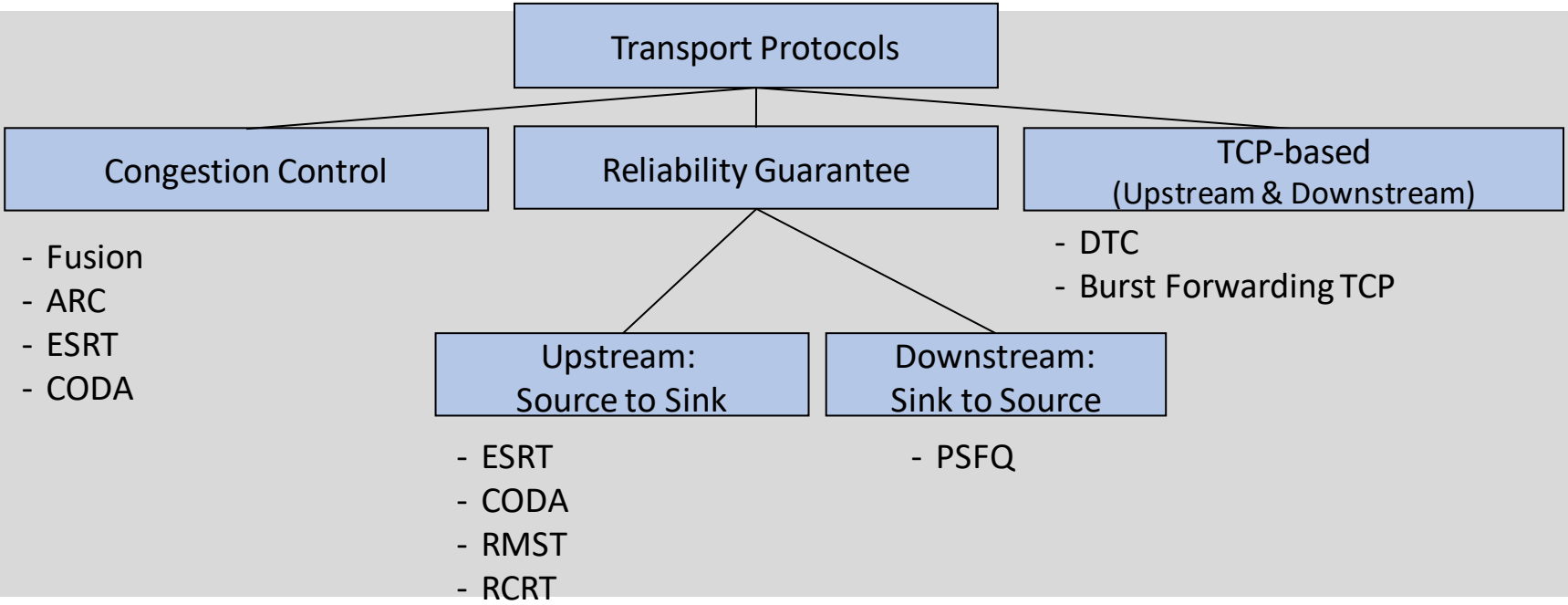
1. Introduction

3. Transport Protocol Challenges

- Reliability
- Congestion control
- Self-configuration: adaptivity to dynamic topologies caused by
  - Node mobility
  - (Temporary) Node failure
- Energy restrictions
- Complexity restrictions / memory
- Interoperability

1. Introduction

4. Transport Protocol Classification



1. Introduction

5. Congestion Control

- Congestion detection by observing different parameters
  - Queue length: Exceeding a certain queue fill level indicates congested node
  - (Packet inter-arrival time / service time) at a node
  - Channel utilization
  - Packet loss
- Congestion notification
  - Explicit notification by special control messages to notify other involved nodes in close neighbourhood or along dissemination tree
  - Implicit notifications by piggy-backing and overhearing
- Rate adjustment
  - Fair or priority-dependent adjustment of transmission rate
  - Often: Additive Increase Multiplicative Decrease

1. Introduction

6. Reliable Data Transport

Data Flows

- Source to sink:
  Sensor data should be delivered to sink with a certain reliability.
- Sink to source:
  Queries, configuration and management information, or code to be transferred without any error.

Error Control

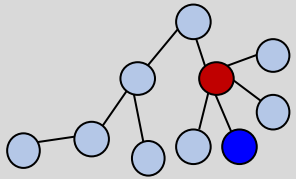
- Loss detection
  - sender-based:
    timer mechanism / overhearing
  - receiver-based:
    out-of-sequence packet arrivals
- Loss notification
  - (positive / implicit / negative) acknowledgments
- Retransmission
  - end-to-end:
    causes delay and requires more energy
  - hop-by-hop: requires buffering

2. Congestion Control Protocols

1. Fusion

Combination of

- Hop-by-hop flow control
  - Setting of congestion bit in outgoing packets dependent on queue level
  - Sensors overhearing congestion bits set by parent (node closer to sink) stop forwarding data. One more packet might be needed to indicate congestion to other nodes.
- Rate limiting source traffic when transit traffic is present
  - Node estimates number of sources routing through parent N, e.g., N = 4
  - 1 token is generated when parent has forwarded N packets.
  - 1 token is required to generate an own packet



- Prioritized MAC protocol
  - Smaller back-off window for congested nodes in order to empty queues

2. Congestion Control Protocols

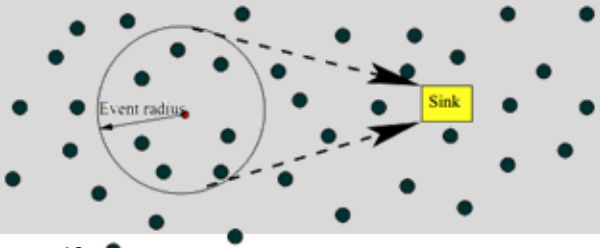
2. Adaptive Rate Control

- Implicit congestion detection
  - Forwarding an injected packet by parent indicates capacity.
  - Packet loss indicates congestion.
- Transmission rate of a node: $p \cdot S$
  - $p \in [0,1]$, S: application transmission rate
- Capacity: linear increase of p by α, $p := \max(1, p + \alpha)$, $(0 < \alpha < 1)$
- Congestion: multiplicative decrease by β: $p := p \cdot \beta$ $(0 < \beta < 1)$
- Different values for β and α in order to penalize routed traffic less in case of congestion and be fair in case of sufficient capacity:
  - $\beta_{route} = 1.5 \cdot \beta_{originate}$
  - $\alpha_{originate} = \alpha_{route} / (N + 1)$, N = number of other nodes routing through a node

2. Congestion Control Protocols

3. Event to Sink Reliable Transport

- Events in a sensor network need to be tracked with a certain accuracy at the sink.
- **Event reporting rate f**: frequency of packet transmissions at a sensor node
- Sink decides about f every i time units (decision intervals).



- Application specific parameters known at sink
  - Observed event reliability r_i
    - Number of data packets received in decision interval i at sink
  - Desired event reliability R
    - Number of packets required for reliable event detection
- Event is reliably detected if $r_i > R$. Otherwise, appropriate actions must be taken to achieve R.
- Normalized reliability: $\eta_i = r_i / R$

2. Congestion Control Protocols

3.1 ESRT Protocol Operation

Problem: Configuration of event reporting rate f at sensor nodes in order to achieve desired event reliability at sink with minimum resource utilization.

Sources (sensor nodes)

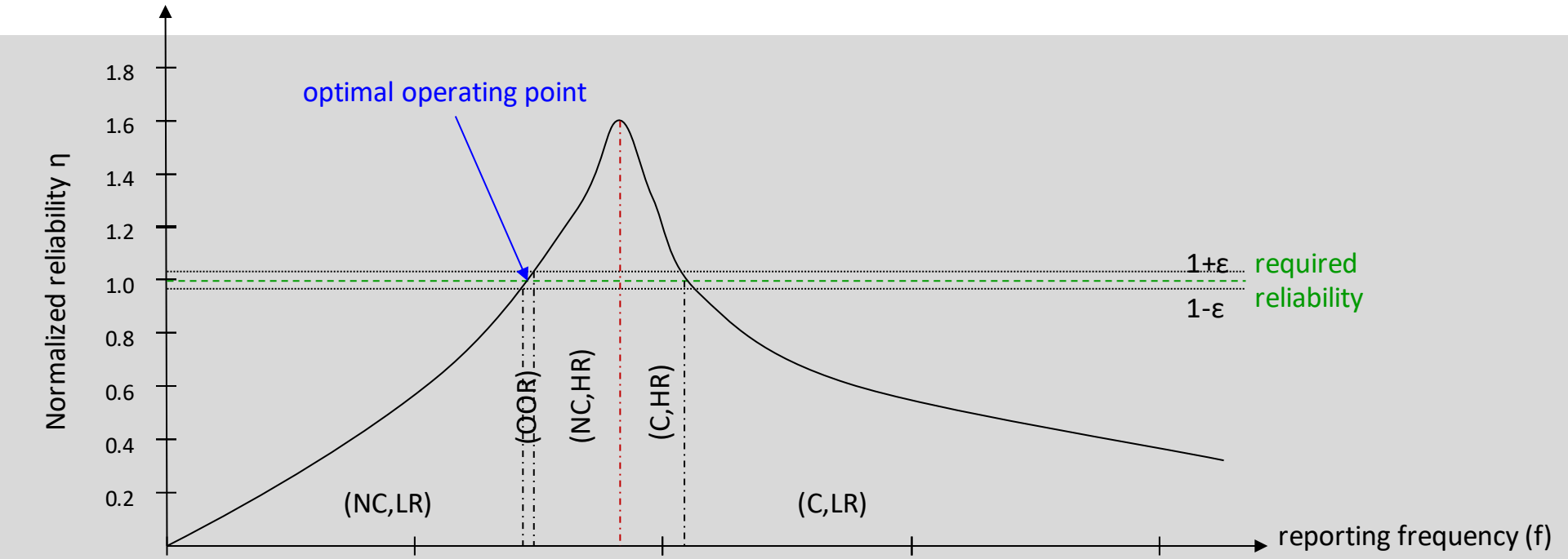
- send sensor data with event reporting rate (frequency) f.
- monitor buffer level and notify congestion to sink.

Sink

- measures observed event reliability r_i at the end of decision interval i.
- detects congestion based on feedback from sensor nodes.
- updates f based on η_i to achieve desired event reliability R.
- calculates and updates (broadcasts) new event reporting rate f to sensor nodes.

2. Congestion Control Protocols

3.2 ESRT Variation of Reporting Rate



2. Congestion Control Protocols

3.3 ESRT Network States

State	Description	Condition
(NC,LR)	(No Congestion, Low Reliability)	$f < f_{max}$ and $\eta < 1 - \varepsilon$
(NC,HR)	(No Congestion, High Reliability)	$f \leq f_{max}$ and $\eta > 1+ \varepsilon$
(C,HR)	(Congestion, High Reliability)	$f > f_{max}$ and $\eta > 1$
(C,LR)	(Congestion, Low Reliability)	$f > f_{max}$ and $\eta \leq 1$
OOR	Optimal Operating Region	$f < f_{max}$ and $\eta \in [1- \varepsilon, 1+ \varepsilon]$

2. Congestion Control Protocols

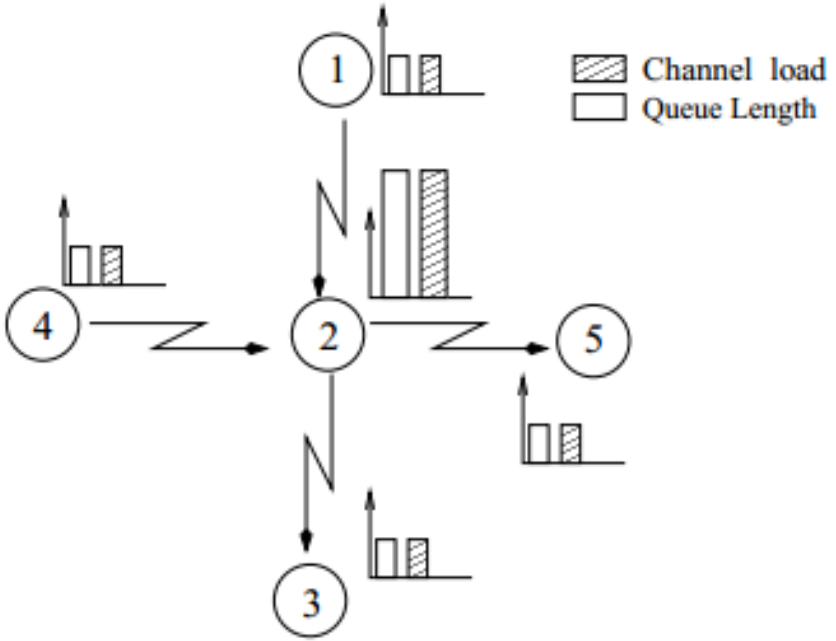
3.4 ESRT Protocol Actions

State	Frequency Update	Comments
(NC,LR)	$f_{i+1} = f_i / \eta_i$	Multiplicative increase Achieve desired reliability asap
(NC,HR)	$f_{i+1} = f_i / 2 * (1 + 1/\eta_i)$	Conservative decrease No compromise on reliability
(C,HR)	$f_{i+1} = f_i / \eta_i$	More aggressive decrease to state (NC,HR)
(C,LR)	$f_{i+1} = f_i \, \eta_i/k$	Exponential decrease (k: number of successive decision intervals in state (C,LR)) Relieve congestion asap
OOR	$f_{i+1} = f_i$	No change

2. Congestion Control Protocols

4.1 Congestion Detection and Avoidance

- Problem
  - In case of events, many sensors might trigger packets towards sink.
    Then, low delays are desired,
    but risk for congestion increases.
- Solution: Congestion Control Mechanisms
  - Congestion detection
    - Monitoring of channel load and buffer occupancy
  - Congestion avoidance
    - Open-loop hop-by-hop backpressure
    - Closed-loop multi-source aggregation



2. Congestion Control Protocols

4.2 CODA Congestion Avoidance

Open-Loop Hop-by-Hop Backpressure

- Nodes broadcast backpressure signals towards sources as long as congestion is detected.
- Nodes should throttle sending rates or drop packets.
- Nodes can forward backpressure signals further dependent on congestion state.
- Local broadcast of Suppression message → All senders but one (to be selected) become silent.

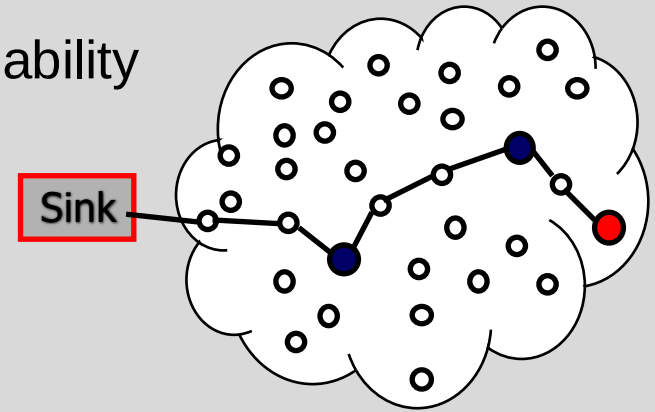
Closed-Loop Multi-Source Regulation

- Mechanism working on a slower time scale
- Assertion of congestion control over multiple sources in order to regulate all sources associated with a particular event.
- If the rate of a source exceeds a certain threshold, the source receives feedback from sink (e.g., ACKs) and adapts its event rate.

3. Reliability Protocols

1. Reliable Multi-Segment Transport

- designed as transport layer for Directed Diffusion Routing
- in-network caching
- provides end-to-end data-packet transfer reliability (in contrast to ESRT)
- in-order delivery is not guaranteed.
- Fragmentation / reassembly



UNIVERSITÄT BERN

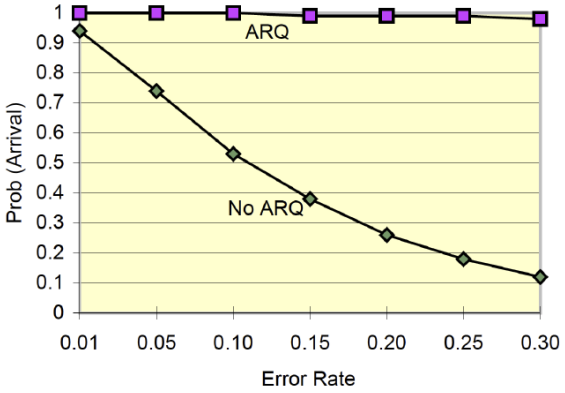
3. Reliability Protocols

1.1 RMST Reliability Placement

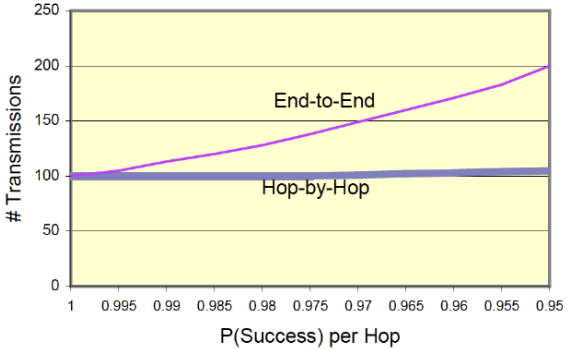
Where to place reliability?
Where to realize retransmissions?

- Application / transport layer → end-to-end
- MAC layer → hop-by-hop
- Hybrid

#Probability of arrival across 6 hops



#Transmissions to send 10 segments across 10 hops



3. Reliability Protocols

1.2.1 RMST Design Choices: MAC Layer Options

- No Automatic Repeat Request
  - All transmissions are broadcast.
  - No RTS/CTS or ACK
  - Reliability deferred to upper layers
  - Benefit: no control overhead
- ARQ always
  - All transmissions are unicast.
  - RTS/CTS and ACKs used
  - One-to-many communication done via multiple unicasts
  - Benefit: Packets 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.

UNIVERSITÄT BERN

3. Reliability Protocols

1.2.2 RMST Design Choices: Transport Layer Options

End-to-End Selective Request NACK

- Loss detection happens only at sinks (end points).
- Repair requests travel on reverse (multi-hop) 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 are sent to immediate neighbors.
- If data is not found in the caches, Repair Requests are forwarded to next hop towards source.

3. Reliability Protocols

1.2.3 RMST Design Choices: Application Layer Options

Wait for next sensor reading

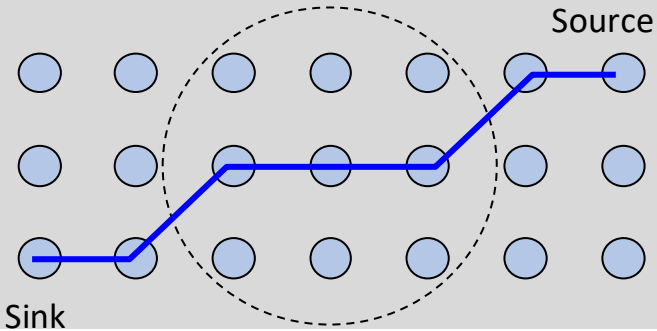
End-to-End Positive ACK

- Sink requests a large data entity.
- Source fragments data.
- Sink keeps sending interests until all fragments have been received.

3. Reliability Protocols

1.3.1 RMST Performance Evaluation

- 6 hop scenario
- max. 4 retries on MAC layer
- 5 KB message
- 100 byte fragments



- Results are presented as normalized byte transmissions to the “ideal byte count”.
- Amount of bytes are calculated and compared to the costs for sending the message without errors and ARQ schemes.
- Amount of bytes transmitted in the ideal case is divided by the amount of bytes transmitted in the examined setting.
- Pure CSMA, no radio duty cycling

3. Reliability Protocols

1.3.2 RMST Performance Evaluation

Packet Error Rate	No ARQ	ARQ Always	Selective ARQ
0.00	0.93	0.57	0.65
0.01	0.51	0.56	0.61
0.10	0.21	0.47	0.54

E2E Positive ACK

Packet Error Rate	No ARQ	ARQ Always	Selective ARQ
0.00	0.99	0.60	0.68
0.01	0.95	0.57	0.67
0.10	0.76	0.48	0.61

Hop-by-hop

Selective Request NACK

Packet Error Rate	No ARQ	ARQ Always	Selective ARQ
0.00	1.00	0.61	0.67
0.01	0.90	0.60	0.66
0.10	n/a	0.49	0.61

End-to-end

UNIVERSITÄT BERN

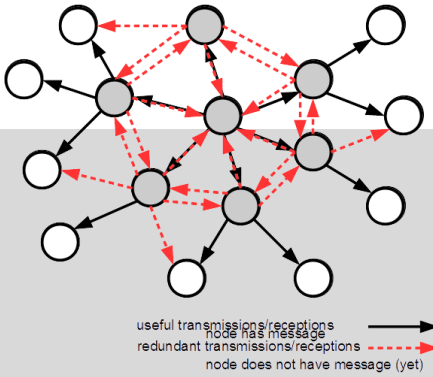
3. Reliability Protocols

2. Pump Slowly Fetch Quickly

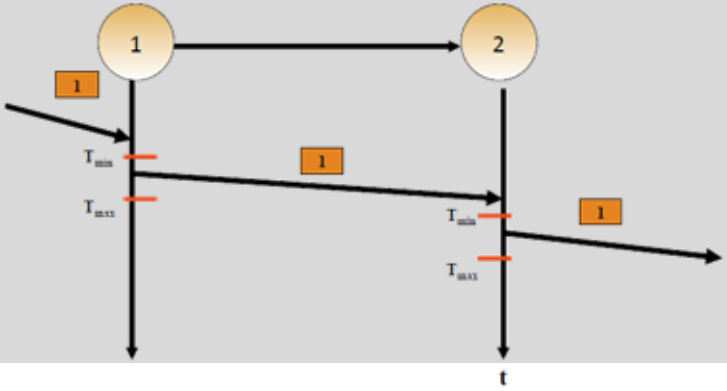
- Pace the data from a source node at a relatively low speed to allow intermediate nodes to fetch missing data segments from their neighbours
  - End-to-end data-packet transfer reliability (in contrast to ESRT) by hop-by-hop recovery
- Protocol Functions
  - Message relaying (pump operation)
  - Error recovery (fetch operation)
  - Selective status reporting (report operation)
- Goals
  - Recover from losses locally
  - Ensure data delivery with minimum support from transport infrastructure
  - Minimize signalling overhead for detection/recovery operations

3. Reliability Protocols

2.1 PSFQ Pump Operation



- PSFQ supports broadcast-based code distribution by a sink.
  - Inject message:
    <file ID, file length, sequence no., TTL, payload>
- Node receiving a packet performs the following protocol operations.
  - For any new message:
    buffer packet and decrement TTL
  - If it is an “in sequence” message and not a duplicate and TTL > 0:
    - Delay packet randomly
    - Rebroadcast packet, if packet has not been forwarded by 4 other nodes and if it has not been broadcast earlier (cache required)



3. Reliability Protocols

2.2.1 PSFQ Fetch Operation

Retransmission requests (NACK)

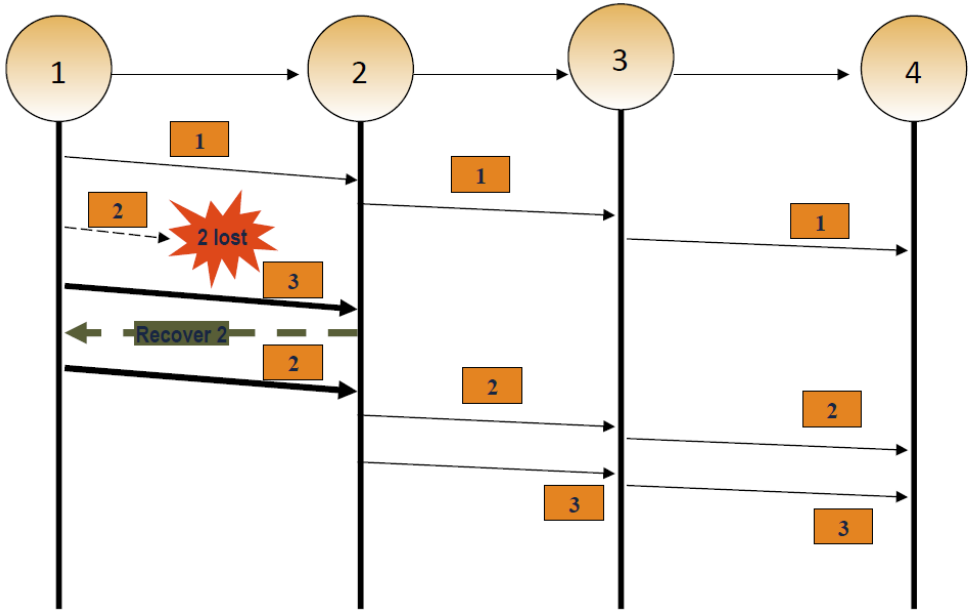
- Node goes into fetch mode when a sequence number gap is detected.
- Node aggressively sends NACK messages to its immediate neighbors to request missing segments.
- Periodic NACK retransmissions, if loss can not be repaired.
- “Proactive fetch” if the last segment has not been received (derived from file length in Inject) and no new packet has been received after a certain period of time.

Retransmissions

- Retransmissions for a window of packets.
- Each neighbor discovering a NACK retransmits after random delay → Repair
- After detecting Repairs, pending retransmissions are canceled.
- Loss recovery from different neighbors are possible.
- Retransmissions trigger forwarding of in-sequence packets
  → avoidance of propagation of loss event

3. Reliability Protocols

2.2.2 PSFQ Fetch Operation



3. Reliability Protocols

2.3 PSFQ Report Operation

Feedback about data delivery status information from sources to sink is triggered by a bit in the Inject message.

Important:
Avoidance of feedback implosion

→ Only last node generates feedback message.
Other nodes piggy-back their status information on the way to the sink.

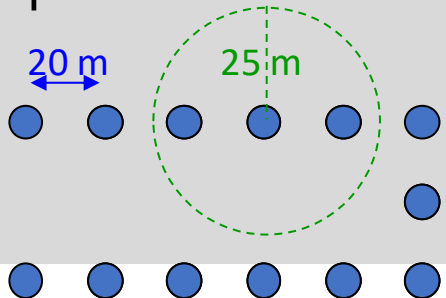
UNIVERSITÄT BERN

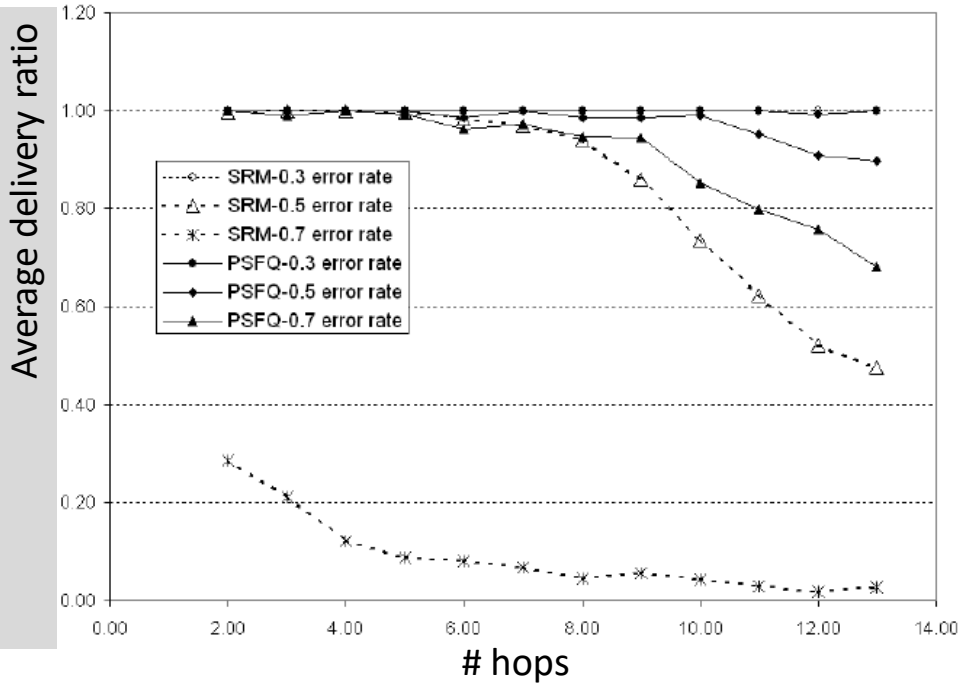
3. Reliability Protocols

2.4 PSFQ Performance Evaluation

Simulation parameters

- 2 Mbps bandwidth
- 2.5 KB file
- 50 bytes packets
- 100 packets / s

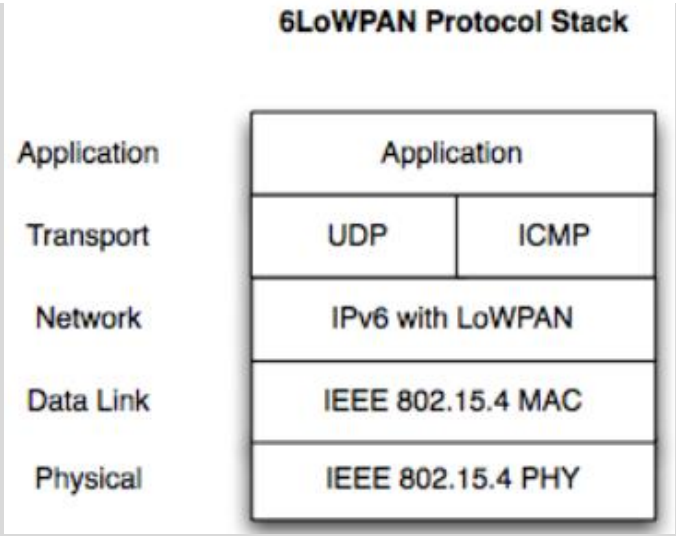




4. TCP in IoT

1. TCP-based Congestion Control & Reliability

- Non-TCP-based protocols require translation between sensor nodes and TCP/IP clients
- IPv6-based solutions: transport via UDP or TCP
- TCP problems
  - TCP has been designed for the wired Internet and showed to perform poorly in wireless multi-hop networks.
  - TCP is unable to distinguish between congestion and transmission losses.
  - TCP performs congestion control and error recovery only in the endpoints. Pre-processing or aggregation of data in intermediate nodes is desirable in sensor networks and often necessary.
  - TCP is heavyweight and might be “too big” for resource constrained WSN nodes



4. TCP in IoT

2. TCP in uIP

- No dynamic memory allocation
- Single global buffer for packets, size: 1 MSS (maximum segment size)
  - Application must immediately process or copy packet.
  - 1 MSS flow control window
- RTT estimation using Karn's algorithm
- Retransmission in collaboration with application

Feature	uIP
IP and TCP checksums	x
IP fragment reassembly	x
IP options	
Multiple interfaces	
UDP	
Multiple TCP connections	x
TCP options	x
Variable TCP MSS	x
RTT estimation	x
TCP flow control	x
Sliding TCP window	
TCP congestion control	Not needed
Out-of-sequence TCP data	
TCP urgent data	x
Data buffered for rexmit	

4. TCP in IoT

3. TCP Issues in the IoT

Claimed issue	Discussion	Solution(s)
Congestion control activation after non-congestion losses	Not specific to TCP (inability to distinguish the cause of packet loss)	Experimental
Link layer interaction	Not specific to TCP (common for end-to-end ARQ)	Experimental
Header overhead	At least 12 bytes greater size than UDP header	TCP header compression (to be developed)
Long TCP connection infeasible due to sleep periods	False claim (if properly configured RDC in use)	Not needed

TFO: TCP Fast Open: embedding of data in SYN/SYN-ACK

Claimed issue	Discussion	Solution(s)
Lack of transport service flexibility	TCP always offers reliable service	No
High latency	Due to three-way handshake	i) Long-lived connections ii) TFO
Multicast incompatibility	TCP is a unicast protocol	No
RTO algorithm issues	RTO not designed for IoT scenarios	Use of CoCoA (see next section)
High complexity	False claim (TCP was designed for computers similar to today's IoT devices)	Not needed

4. TCP in IoT

4. TCP Profile for IoT Devices

- 6LoWPAN adaptation layer does not support packets > 1280 bytes
  → MSS to be set appropriately
- Often short messages in IoT (except software updates)
  → stop-and-wait behaviour using single-MSS window
- Long-lived connections to reduce communication overhead
- Use of TCP options such as Selective Acknowledgments for larger windows
- Use of Explicit Congestion Notification to detect congestion earlier and reduce packet loss

- Proposed Retransmission Timeout estimation algorithm: COAP Simple Congestion Control/Advanced
  - RTO estimator runs two copies of the algorithm defined in RFC6298, using the same variables and calculations to estimate the RTO
    - Strong estimator for completed initial transmissions
    - Weak estimator for exchanges that have run into retransmissions, where only the first two retransmissions are considered.

$$\mathrm{RTTVAR}_X = (1-\beta) \times \mathrm{RTTVAR}_X + \beta \times |RTT_X - RTT_{X\_new}|$$

$$RTT_X = (1-\alpha) \times RTT_X + \alpha \times RTT_{X\_new},$$

$$\mathrm{RTO}_{overall} = 0.5 \times \mathrm{RTO}_X + 0.5 \times \mathrm{RTO}_{overall}$$

$\alpha = \frac{1}{4}$ and $\beta = \frac{1}{8}$.

$$\mathrm{RTO}_X = RTT_X + K_X \times \mathrm{RTTVAR}_X,$$

with $K_X = 4$.

4. TCP in IoT

5. Distributed TCP Caching

Approach

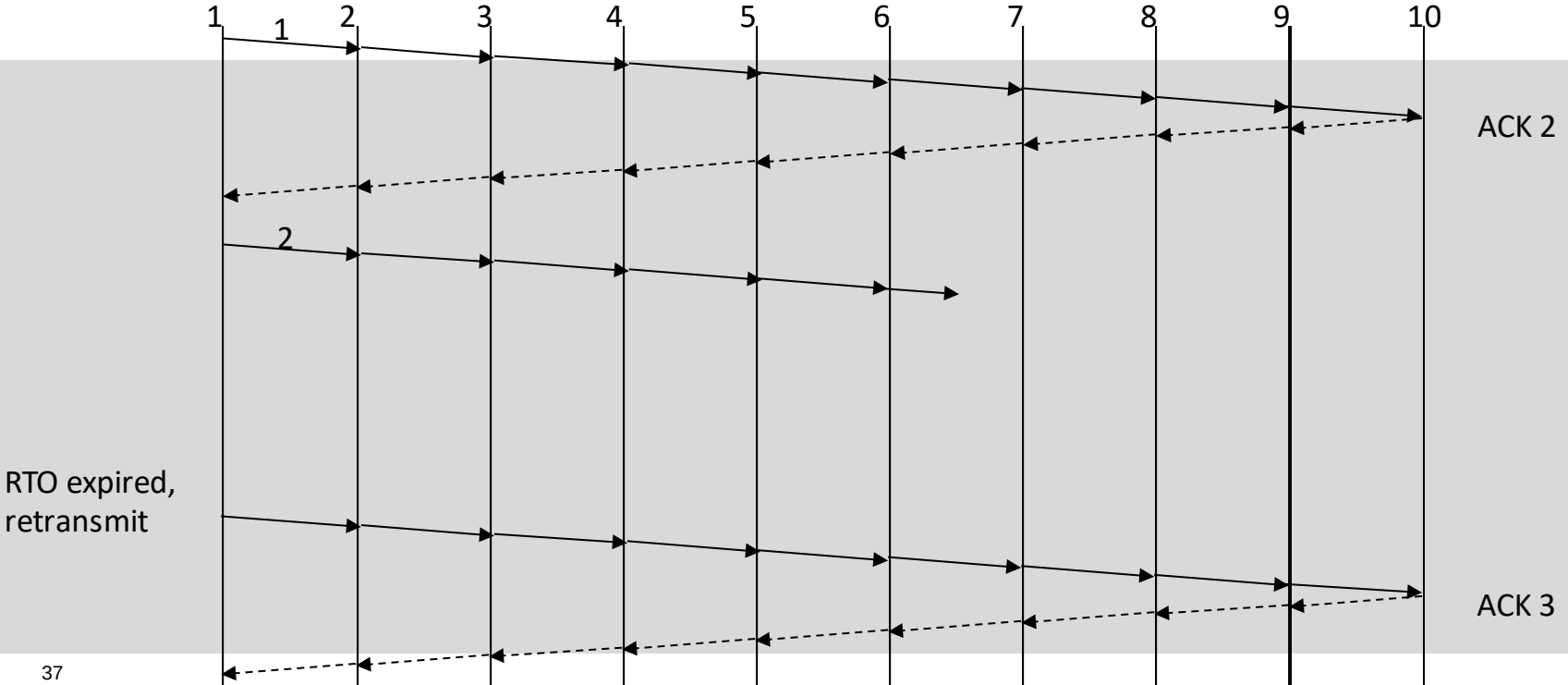
- Use TCP for reliable data transfer, in particular for configuration, management, re-programming
- Avoid expensive end-to-end retransmissions by caching inside the sensor network
- Assumption: A sensor node can only cache one TCP segment.

Protocol Mechanisms

- Caching
  - Each intermediate node caches segment with the highest seen sequence number with certain probability.
  - Lock cache for link level ACK timeout
  - Release cache for link level ACK or TCP ACK
- Timeout based packet loss detection and local retransmissions
  - Each node measures round-trip time (RTT) to destination.
  - Flying start: use SYN/SYNACK pair as initial value
- Selective TCP acknowledgements
  - Intermediate nodes acknowledge data locked in cache
- Local regeneration of TCP acknowledgements for duplicate TCP segments

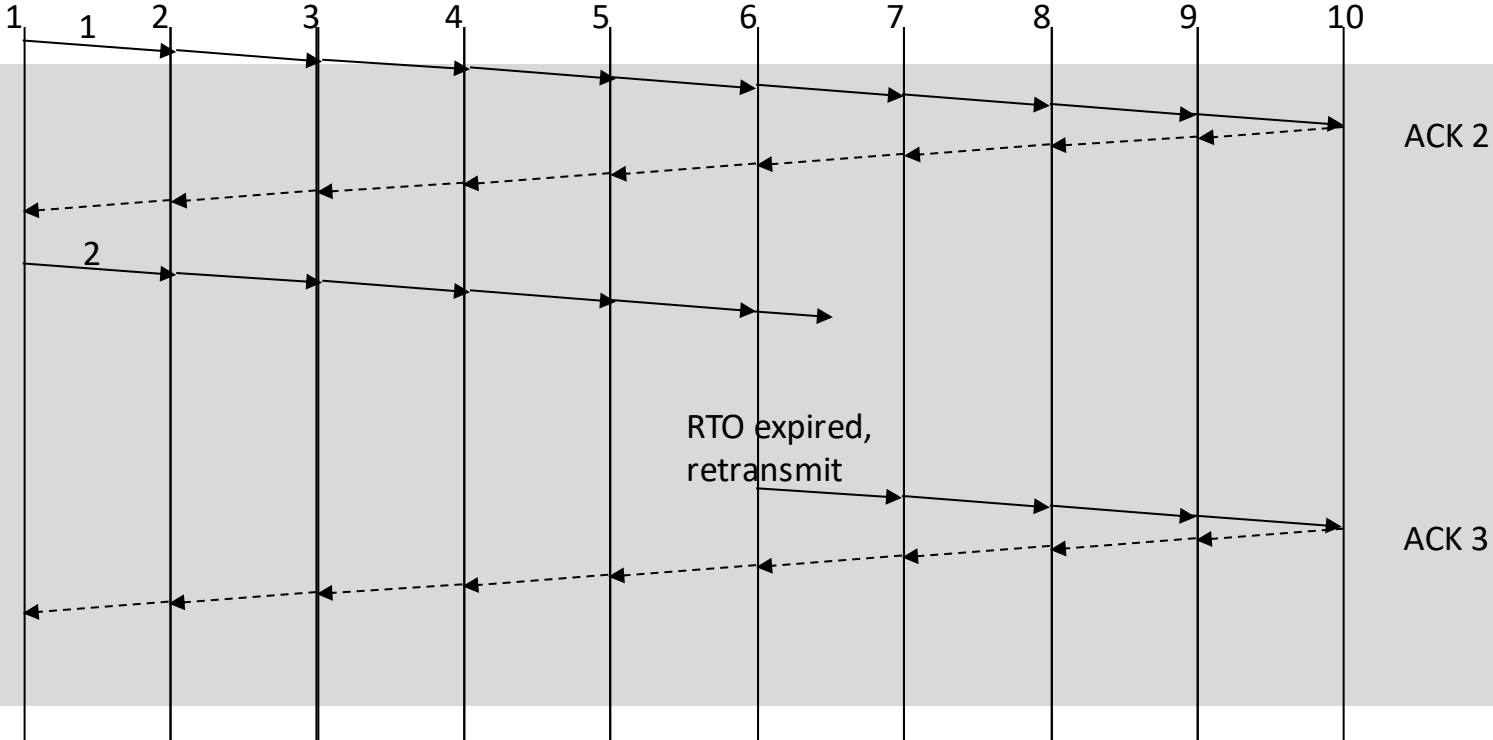
4. TCP in IoT

5.1 TCP-based End-to-End Retransmissions in DTC



4. TCP in IoT

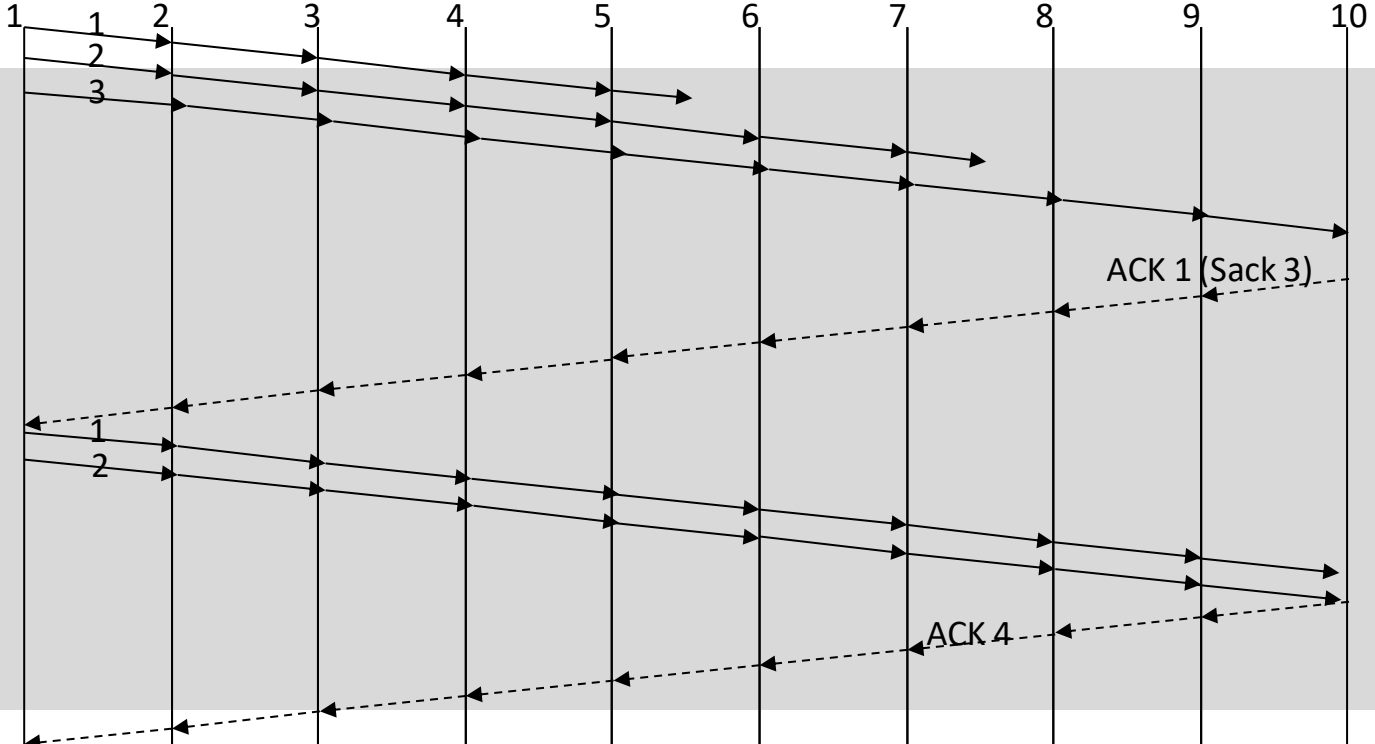
5.2 Local Retransmissions in DTC



UNIVERSITÄT BERN

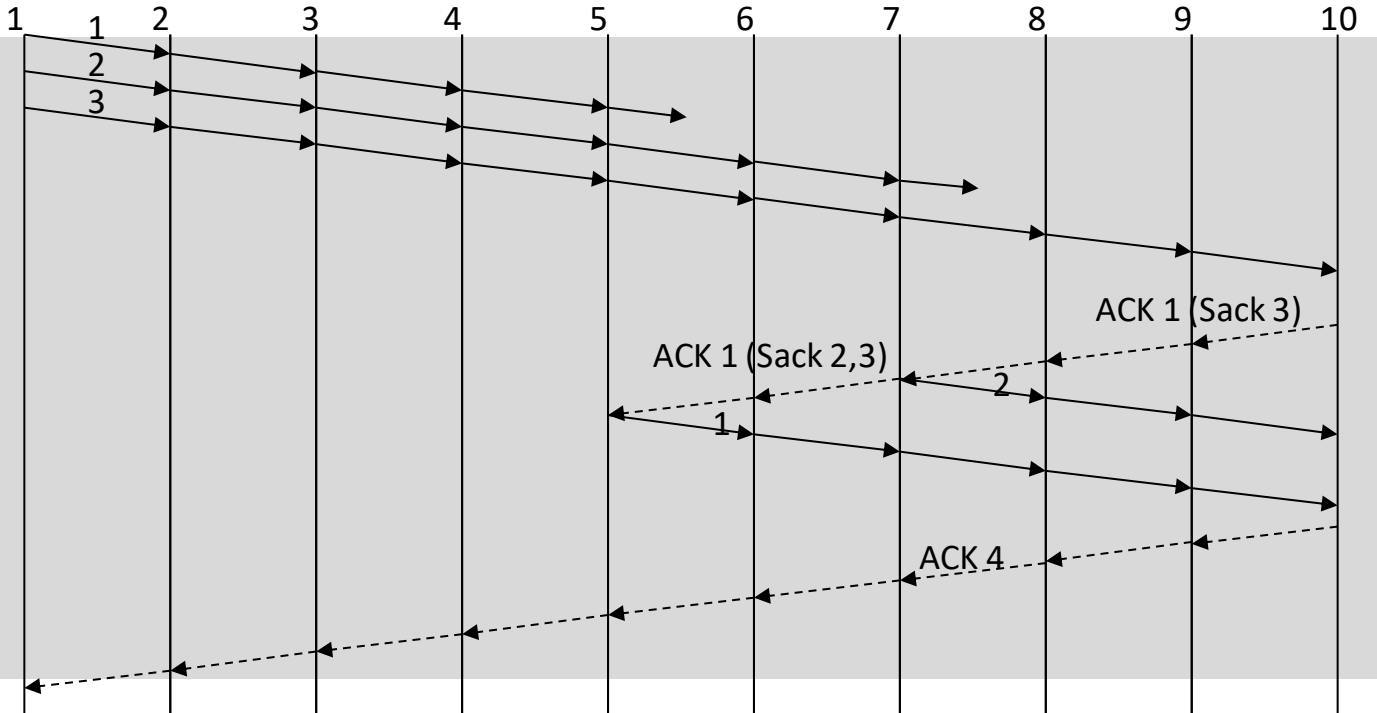
4. TCP in IoT

5.3 Selective TCP Acknowledgements in DTC



4. TCP in IoT

5.4 Local Caching & Selective TCP Acknowledgements in DTC



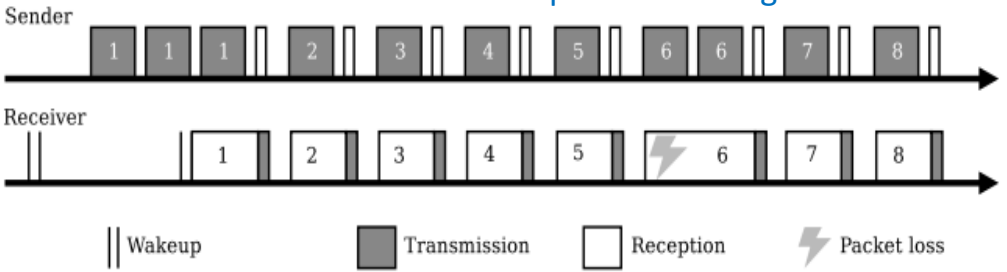
UNIVERSITÄT BERN

4. TCP in IoT

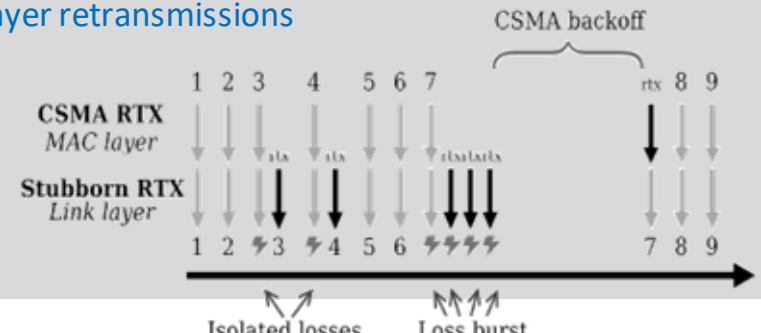
6.1 Burst Forwarding TCP

- Burst Forwarding Layer (below TCP/IP)
  - communicates and tunes MAC / link layer parameters.
  - transparently assists TCP in lossy wireless environment.
- Hop-by-Hop Reliability using intelligent MAC layer mechanisms:
  - Rapid (stubborn) link-layer retransmissions (on radio chip)
  - MAC layer retransmissions
- Low-Power Packet Bursting
  - Transmission of consecutive frames in a burst over one hop

Contiki MAC packet bursting



Link Layer retransmissions

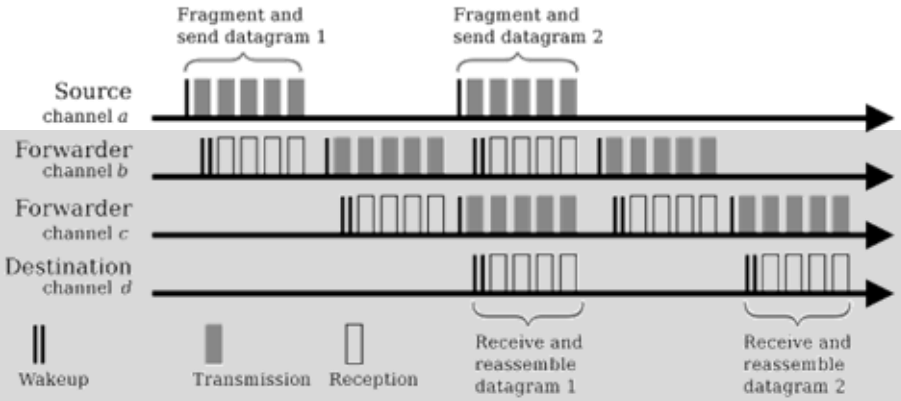


4. TCP in IoT

6.2 Burst Forwarding TCP

Burst Coordination

- Coordination of burst transmissions across multiple hops
- Use of multiple channels to increase spatial reuse of the wireless medium
- Nodes pick channel at random and change it if needed.
  → Nodes learn each others' channel and save it.



Fragmentation

- No fragmentation and reassembly on each hop
- Nodes learn next hop after first fragment and forward frames without passing it to TCP/IP layers.

Thanks

for Your Attention

Prof. Dr. Torsten Braun, Institut für Informatik

Bern, 26.04.2021 – 03.05.2021