Networked Embedded Systems WS 2016/17

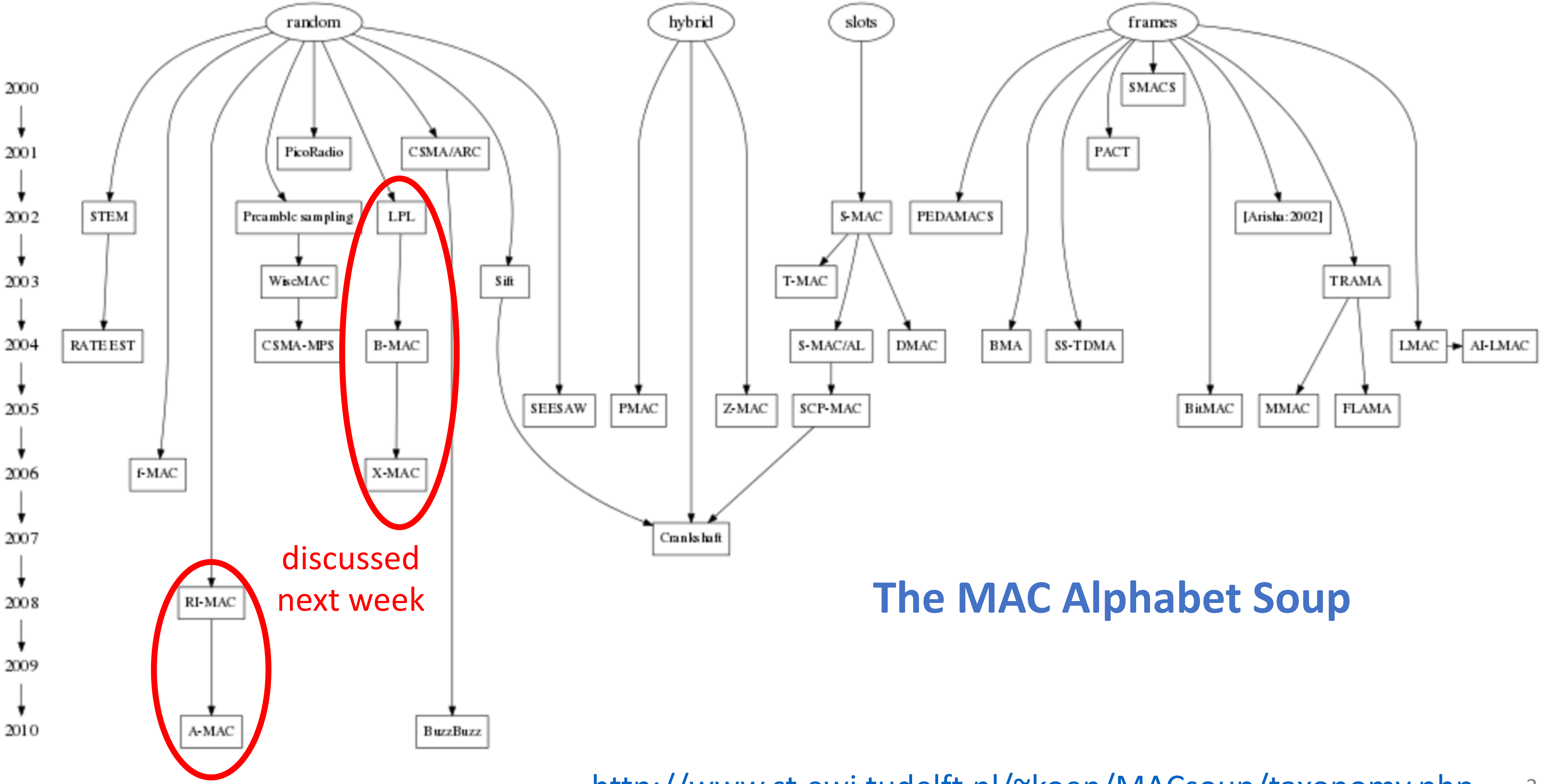
Lecture 4: Communication

Marco Zimmerling

TECHNISCHE UNIVERSITÄT DRESDEN

cfaed CENTER FOR ADVANCING ELECTRONICS DRESDEN

Where Did We Stop Last Time?



discussed next week

The MAC Alphabet Soup

http://www.st.ewi.tudelft.nl/~koen/MACsoup/taxonomy.php

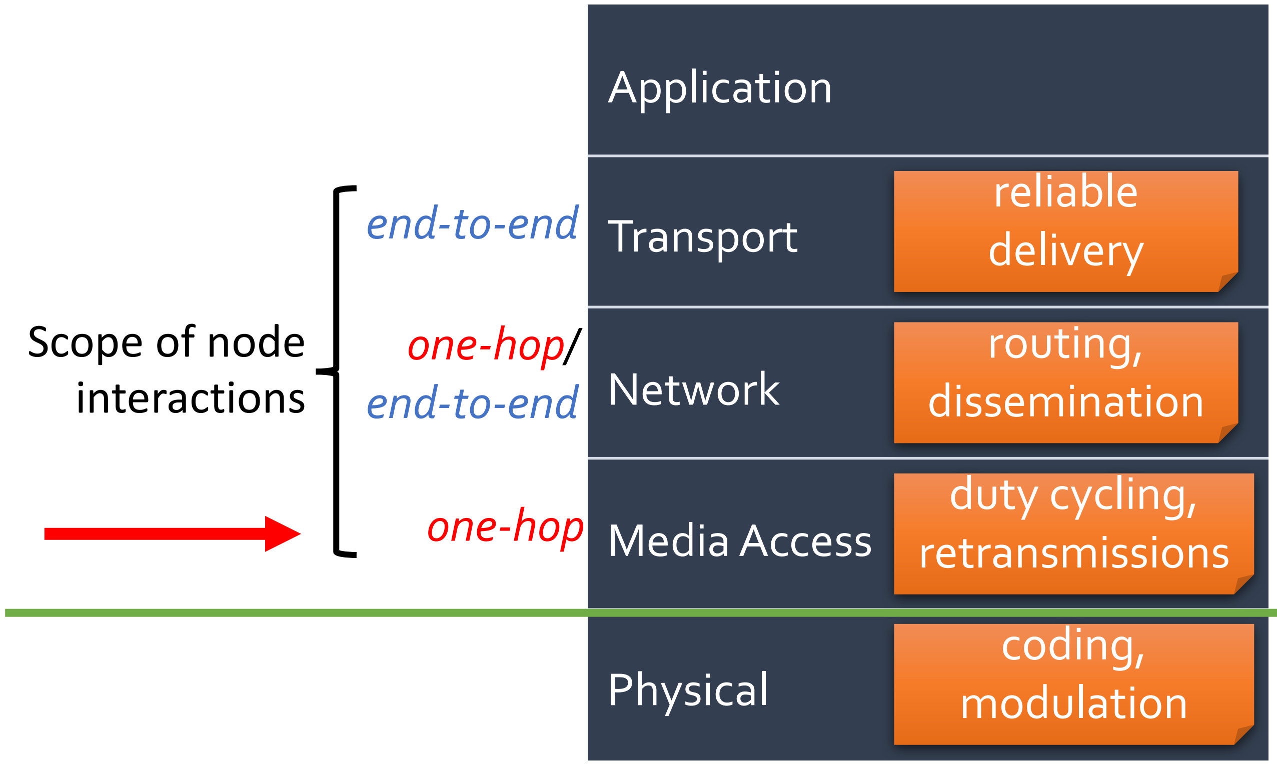
Goal of Today’s Lecture

- Traditional low-power wireless communication stack: key principles
- Low-power wireless bus

Goal of Today's Lecture

- Traditional low-power wireless communication stack: key principles
- Low-power wireless bus

Low-power Wireless Communication Stack



Scope of node interactions

end-to-end — Transport — reliable delivery

one-hop/ *end-to-end* — Network — routing, dissemination

one-hop — Media Access — duty cycling, retransmissions

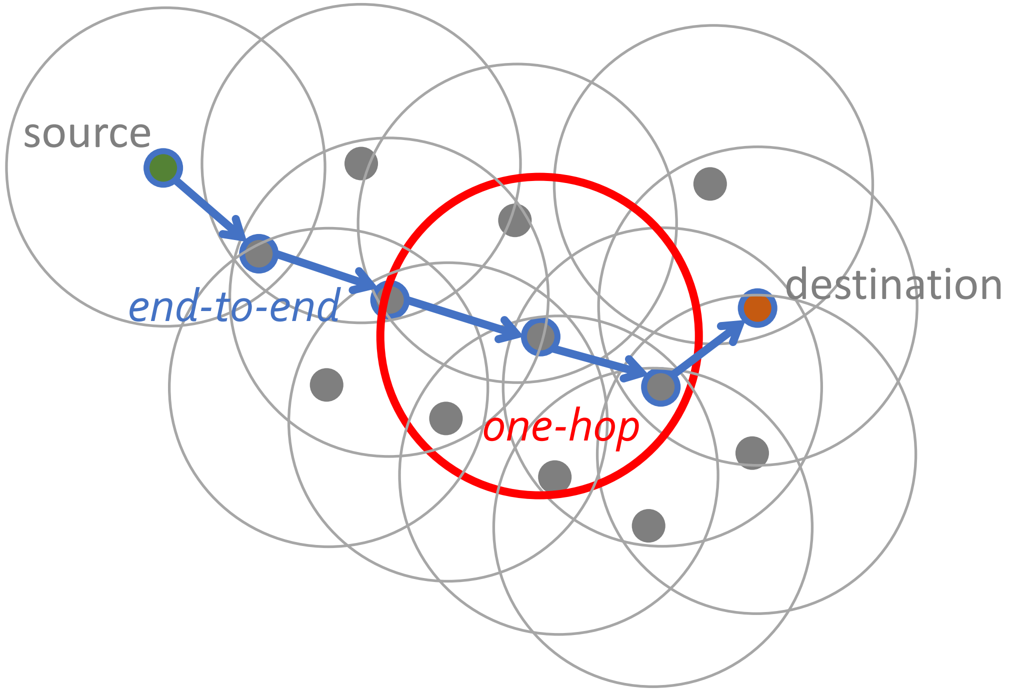
Physical — coding, modulation

Application

Software

Hardware

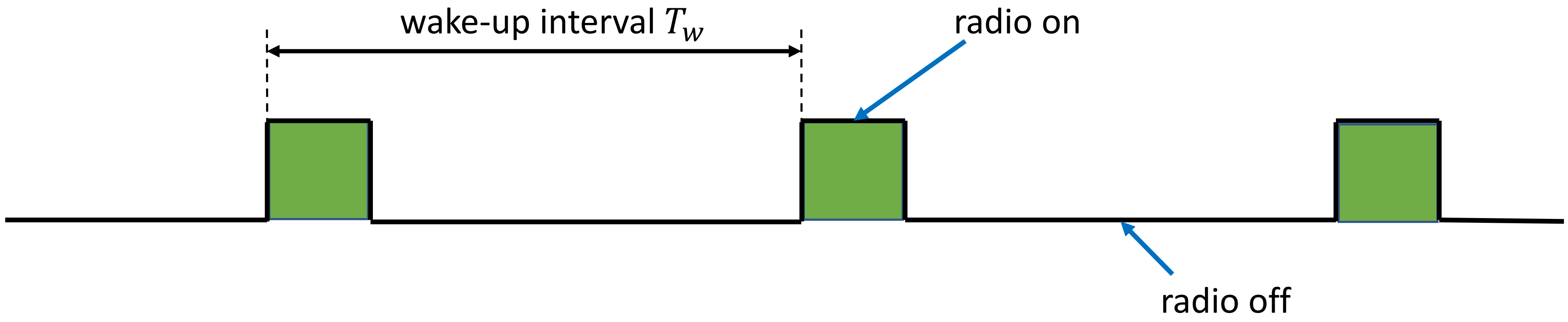
Bridged by radio driver



Low-power Media Access Control (MAC)

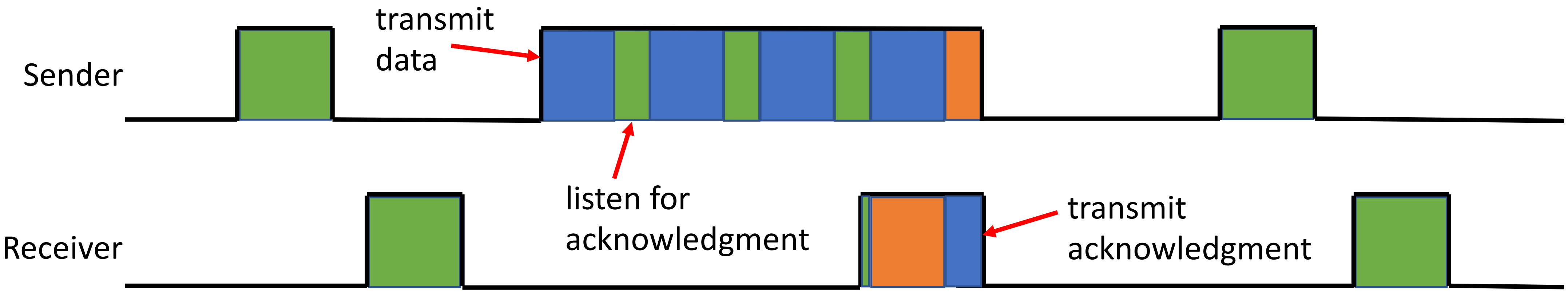
- Radio transceiver may consume significant amount of energy
  - Current draw in receive, transmit, or idle listening mode: a few mA
  - Current draw in deep sleep mode: a few μA

- Key principle: *radio duty cycling*

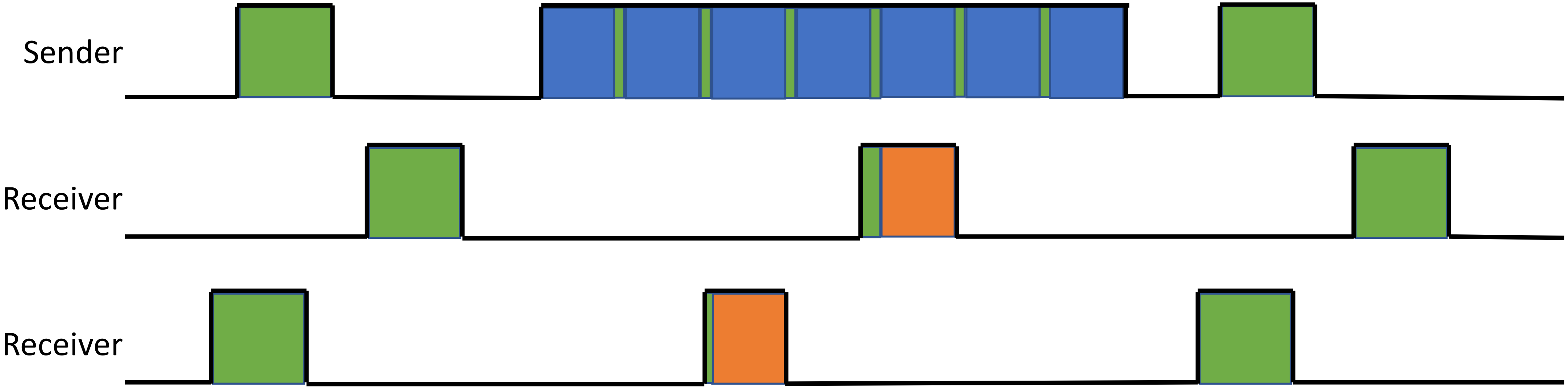


Low-power Listening (LPL)

- *Unicast*: sender initiates transmission to an asynchronous receiver

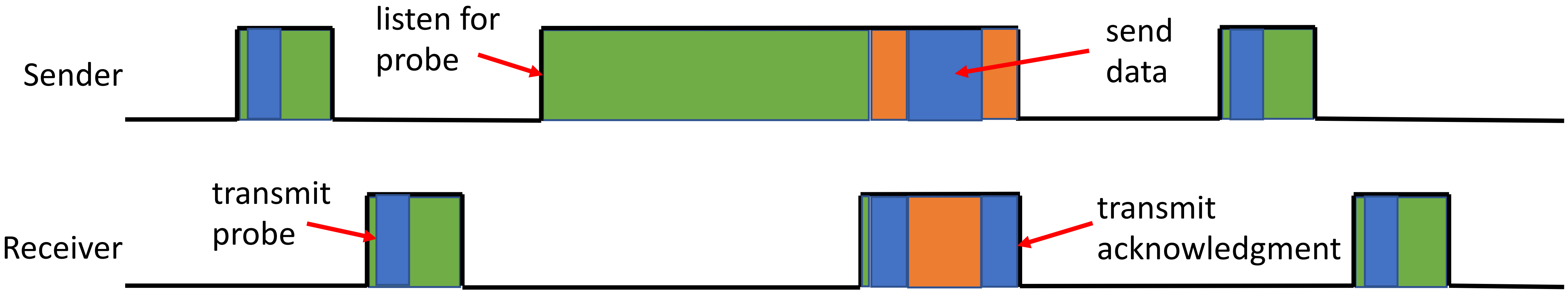


- *Broadcast*: repeatedly send data packet to reach all neighbors

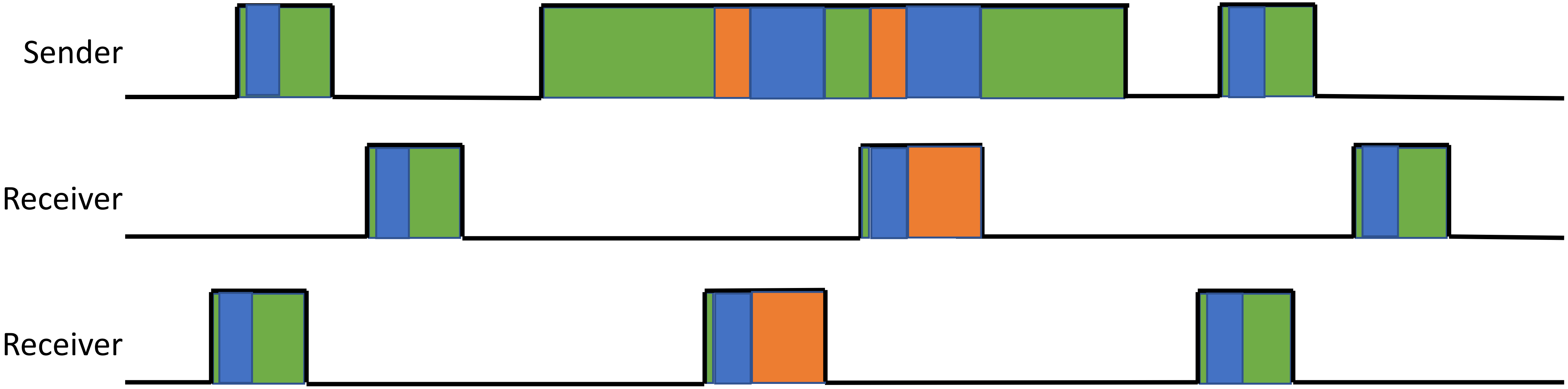


Low-power Probing (LPP)

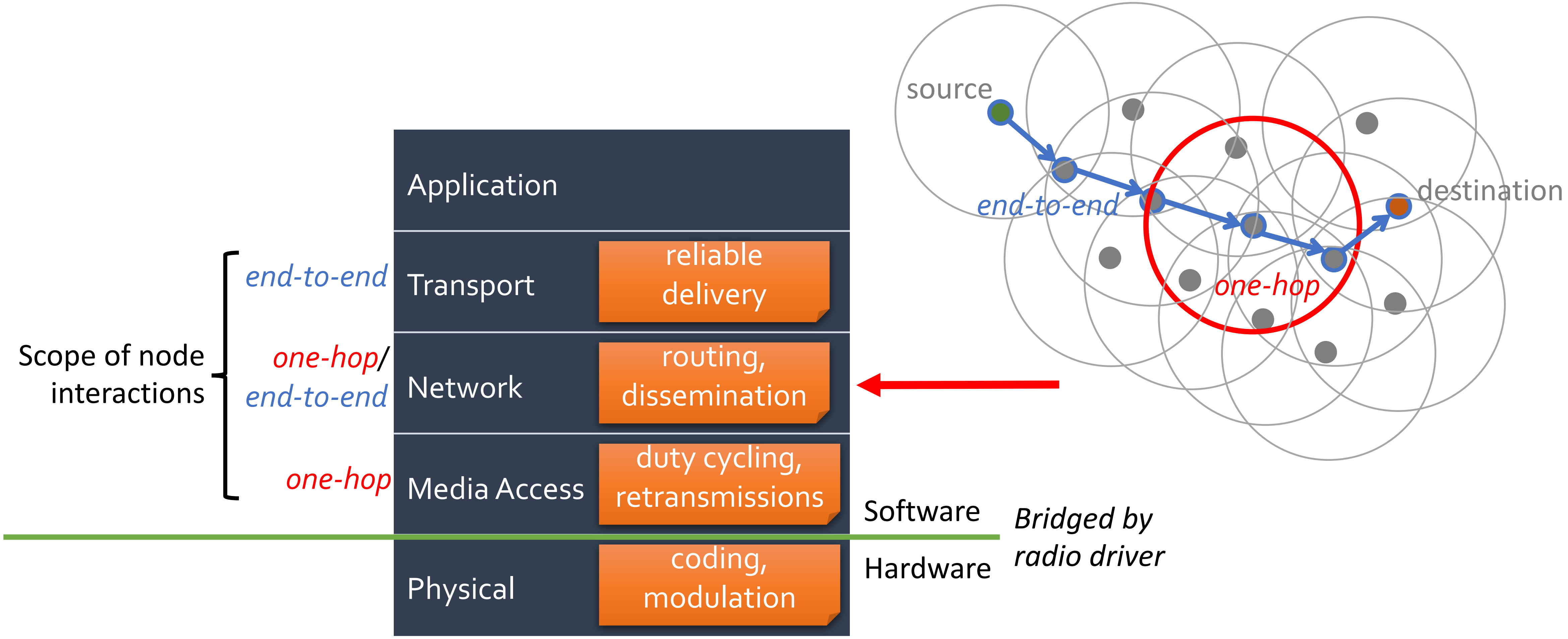
- *Unicast*: receiver initiates transmission by an asynchronous sender



- *Broadcast*: repeatedly reply to probes with data packet



Low-power Wireless Communication Stack



Scope of node interactions

end-to-end

one-hop/ *end-to-end*

one-hop

Application

Transport — reliable delivery

Network — routing, dissemination

Media Access — duty cycling, retransmissions

Physical — coding, modulation

Software

Hardware

Bridged by radio driver

source

end-to-end

one-hop

destination

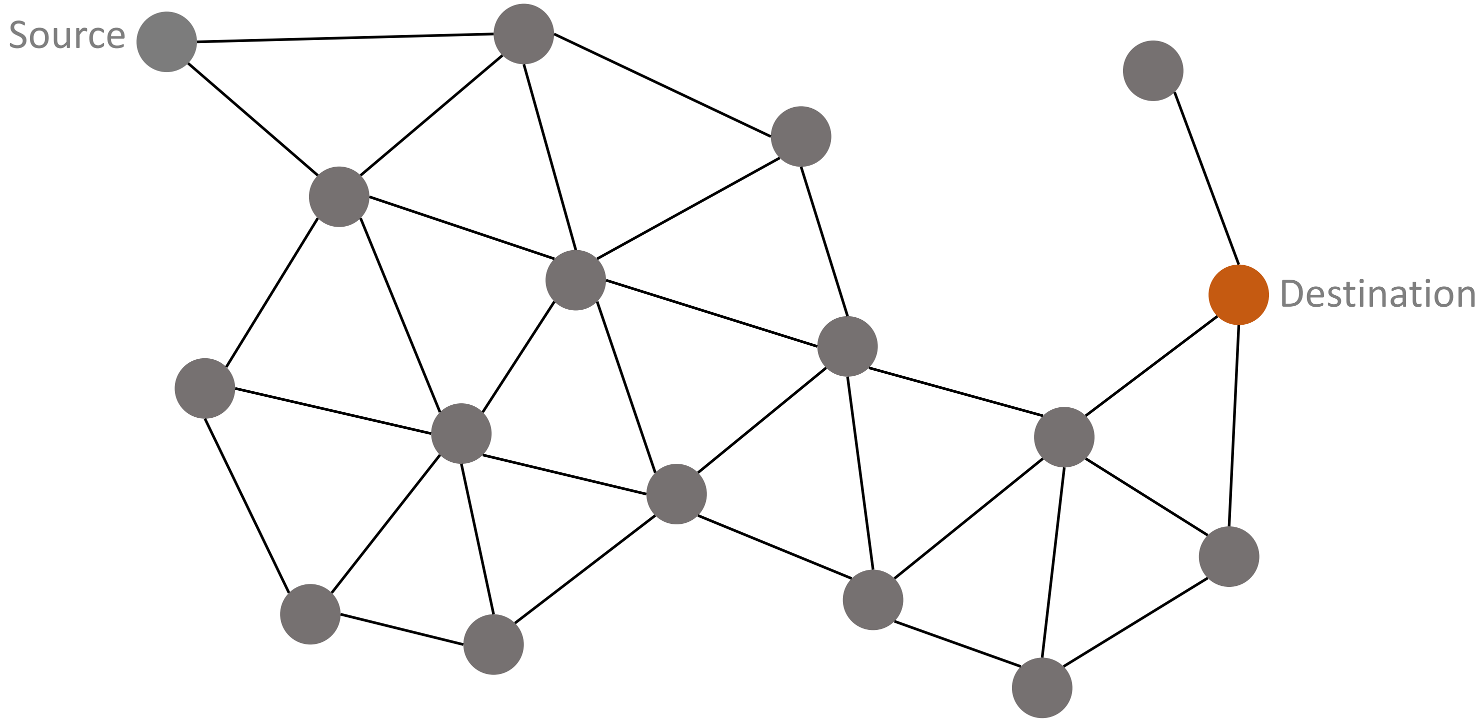
Network Protocols

- Collection (many-to-one or many-to-a-few)
  - Tree- or DAG-based routing (DAG = directed acyclic graph)
  - Select next-hop node based on some routing metric, such as expected number of transmissions (ETX)
  - Goal is to minimize number of transmissions while providing high reliability
- Dissemination (one-to-many or one-to-all)
- Flooding (one-to-all)
- Point-to-point routing
- …

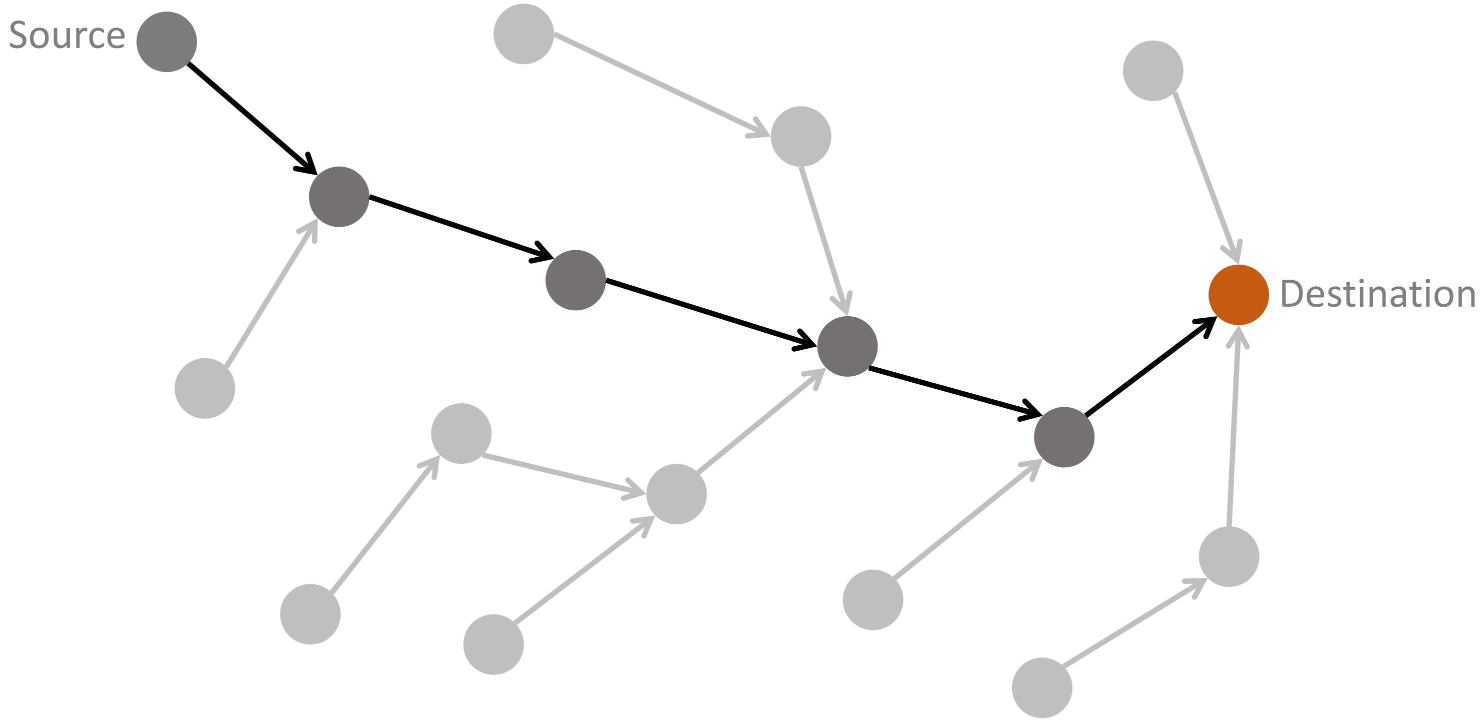
Goal of Today's Lecture

- Traditional low-power wireless communication stack: key principles
- Low-power wireless bus

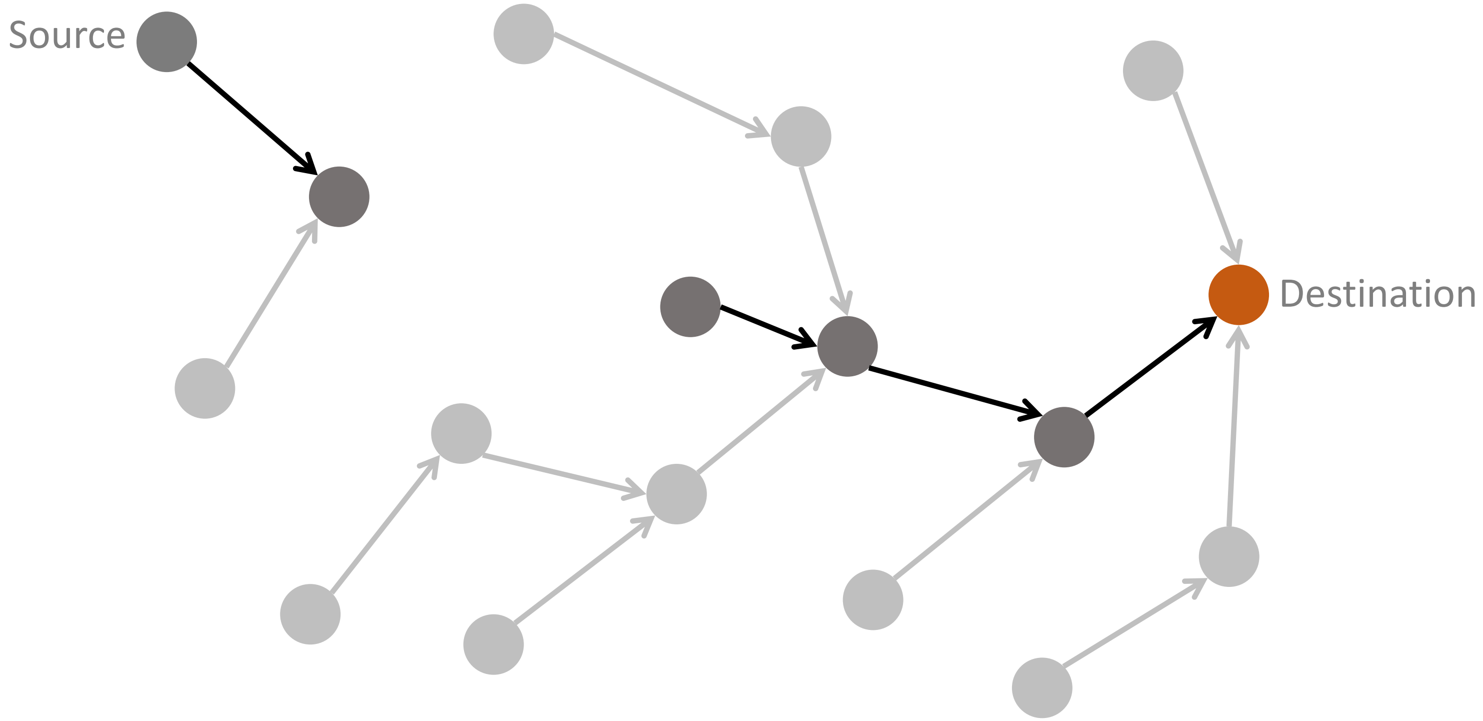
Existing low-power wireless protocols treat wireless channel as a point-to-point link



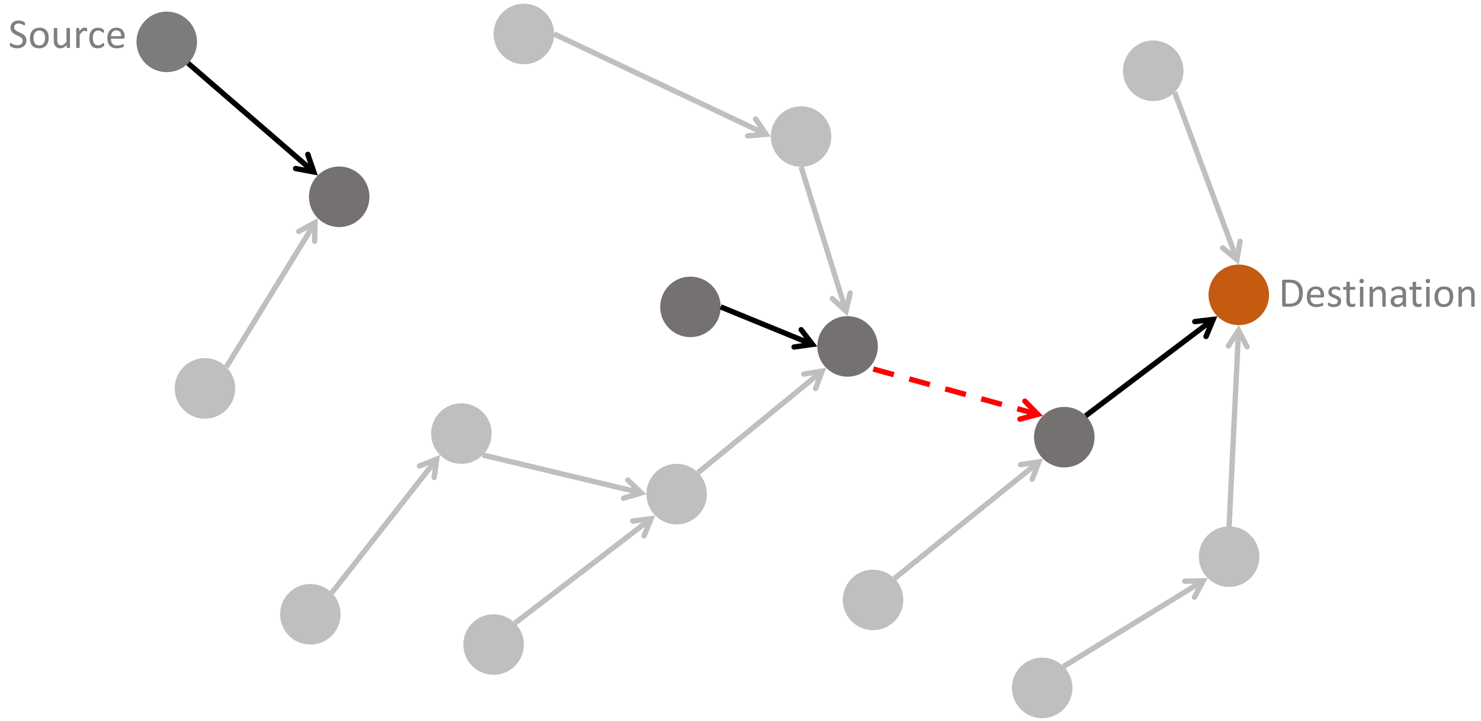
End-to-end behavior and performance are functions of all links on a path and the network state at each intermediate node



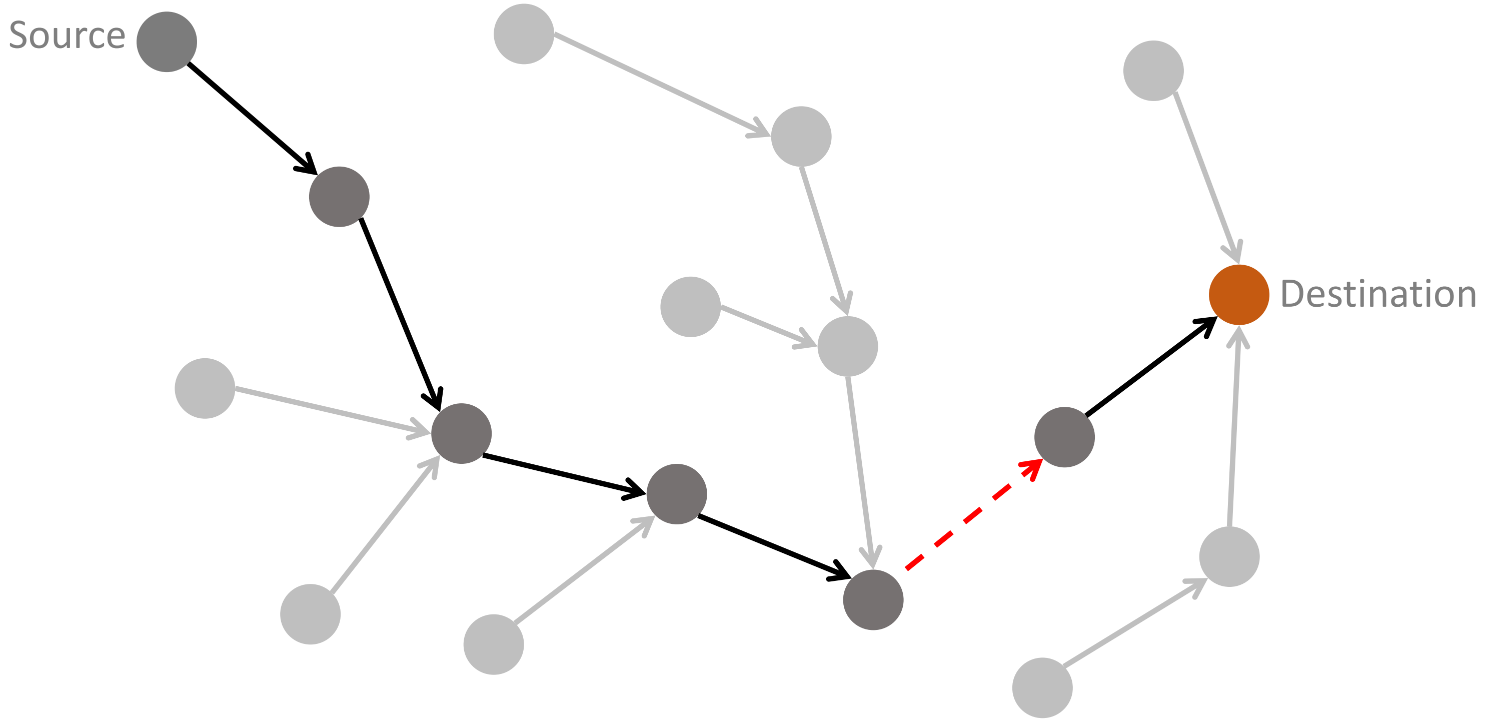
But links can suddenly disappear and their statistical properties change continuously



But links can suddenly disappear and their statistical properties change continuously



And the network state at each intermediate node changes due to complex distributed interactions

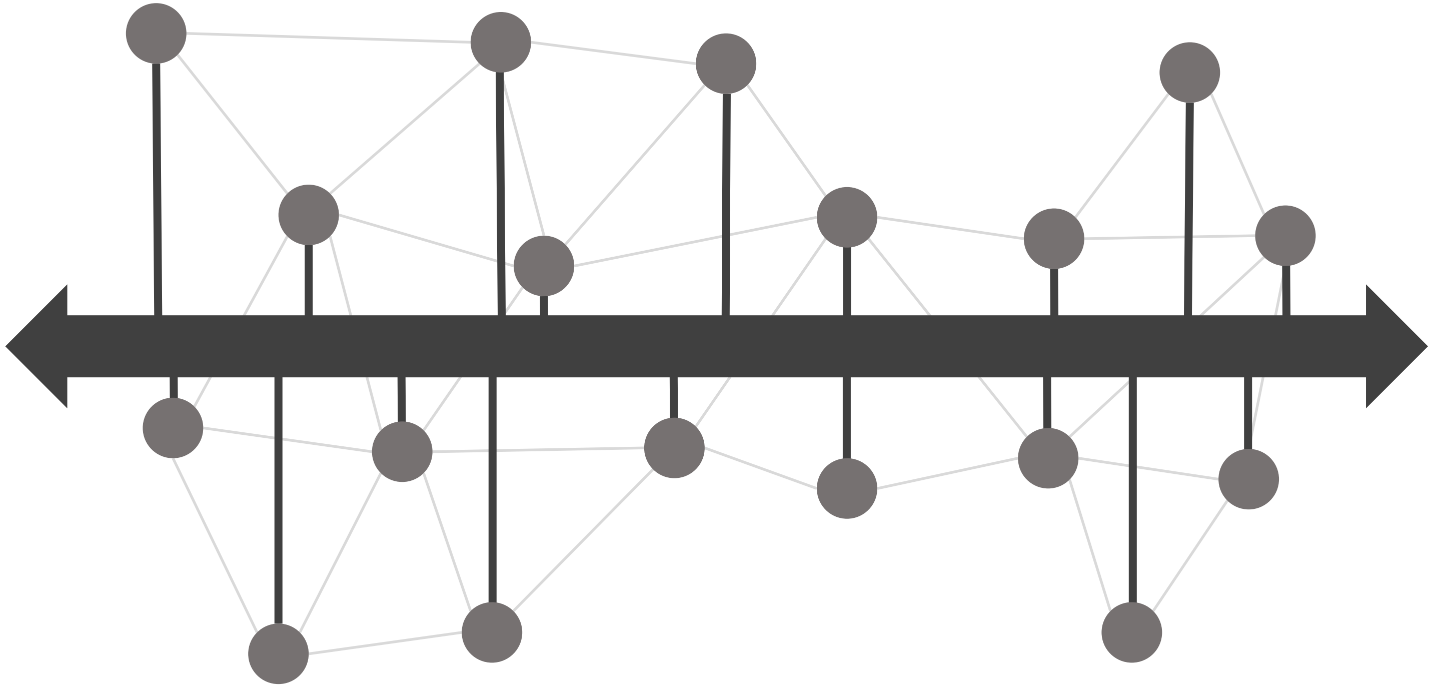


Traditional embedded systems use wired buses

FlexRay, Real-Time Ethernet

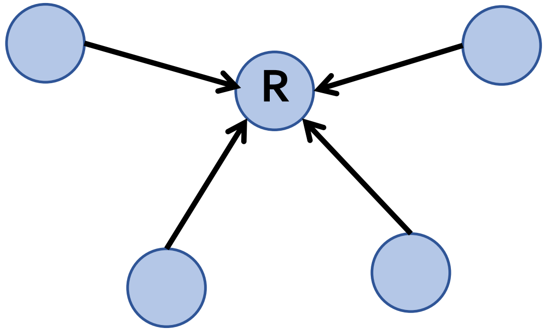
Time-Triggered Protocol (TTP)

Let us design and implement a wireless bus!

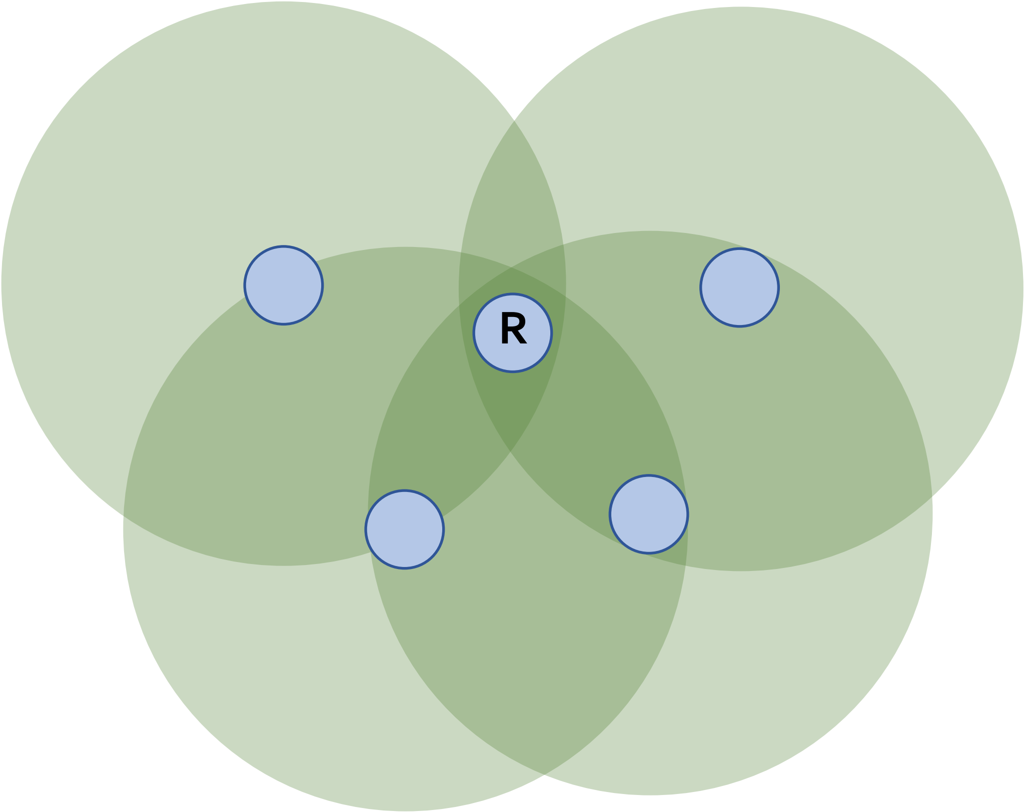


Synchronous transmissions

Do not avoid interference, but use it to your advantage

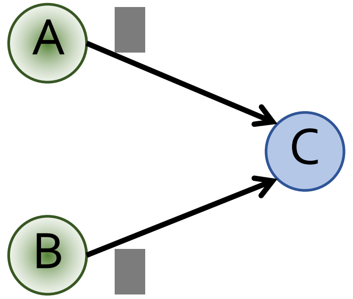


Link-based transmissions



Synchronous transmissions

Synchronous transmissions in 802.15.4 work thanks to power capture and constructive interference



Power capture

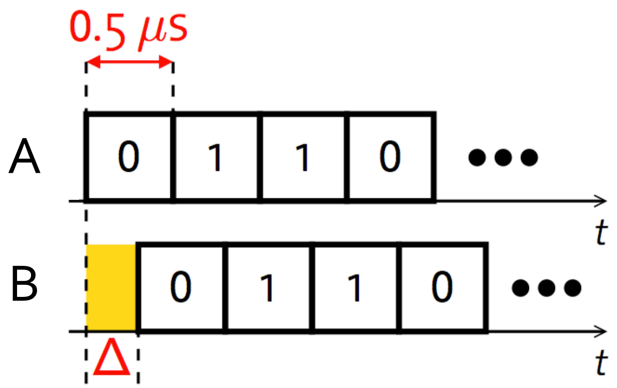
Occurs with high probability if one signal ~3dB stronger than the other

And the stronger signal arrives within 160 us after the weaker signal

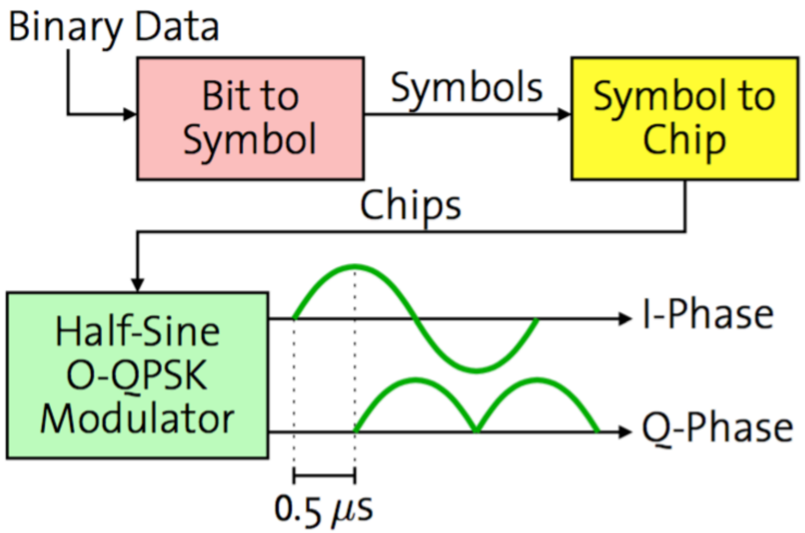
Constructive interference

Occurs with high probability if $\Delta \leq 0.5\,\mu s$

Significant impact of spreading code



Where does the 0.5us timing requirement come from?



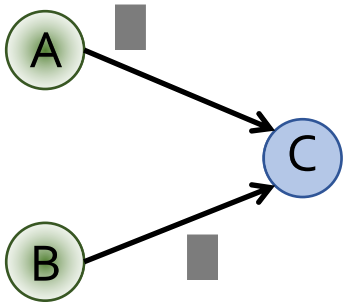
Data	**Symbols**	**Chips** ($c_0\, c_1\, \ldots\, c_{31}$)
0	4 →	01010010
0		00101110
1		11011001
0		11000011
0	14 →	10010110
1		00000111
1		01111011
1		10001100

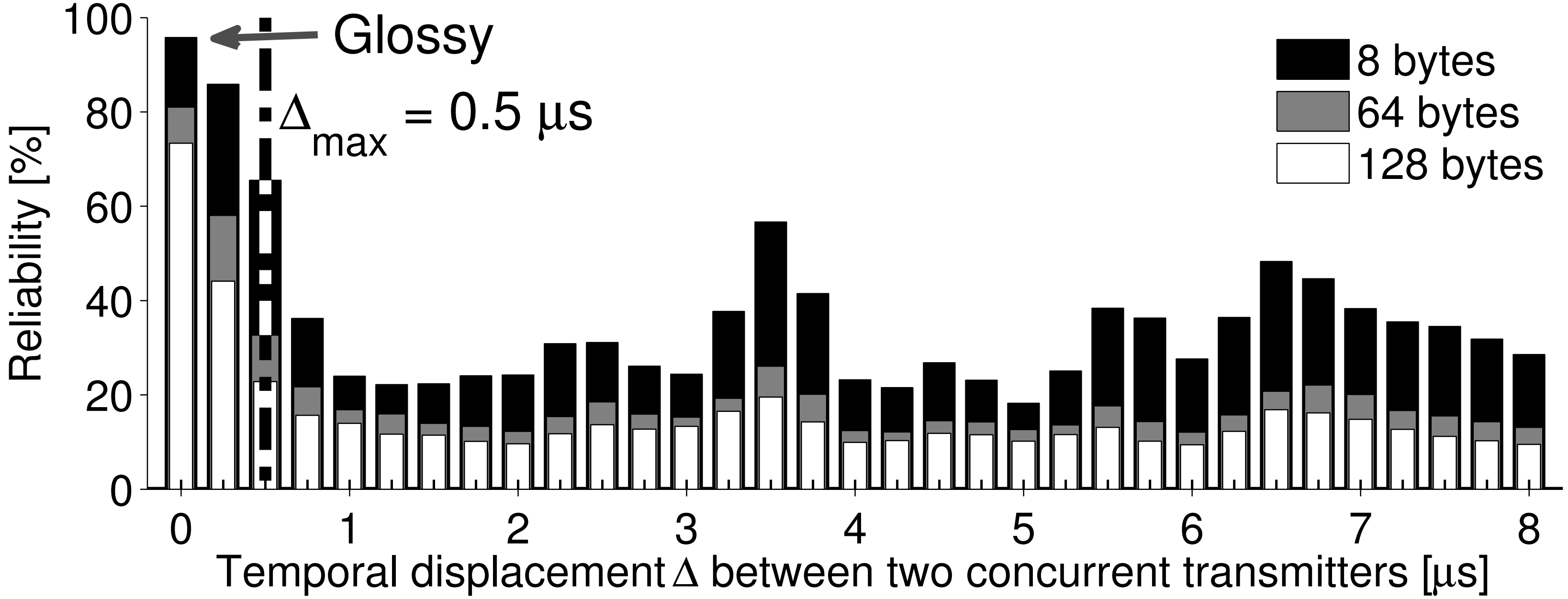
Redundancy of spreading code helps error correction

Max 90° phase shift of RF carrier every 0.5us

Time offset below 0.5us to avoid inter-symbol interference

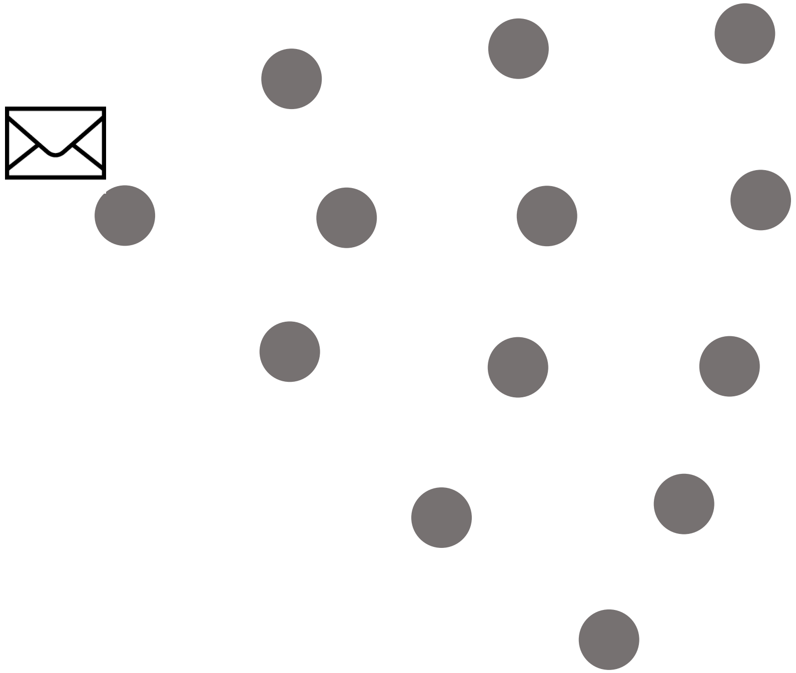
Robustness of synchronous transmissions: effect of time offset and packet size





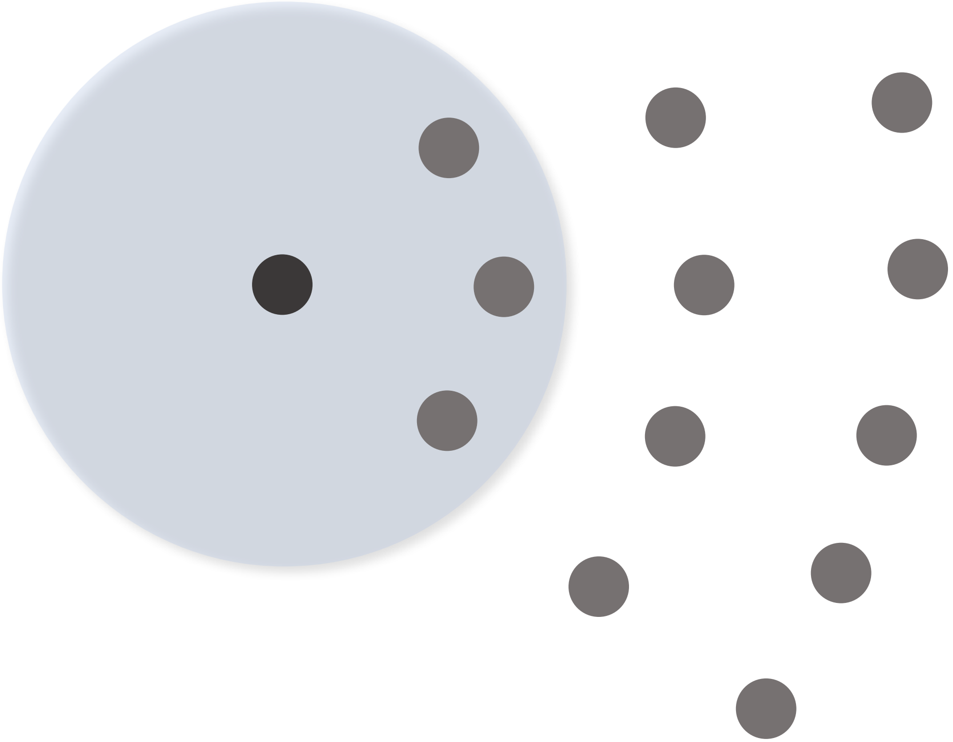
Glossy

Glossy exploits synchronous transmissions for network flooding (aka broadcasting)

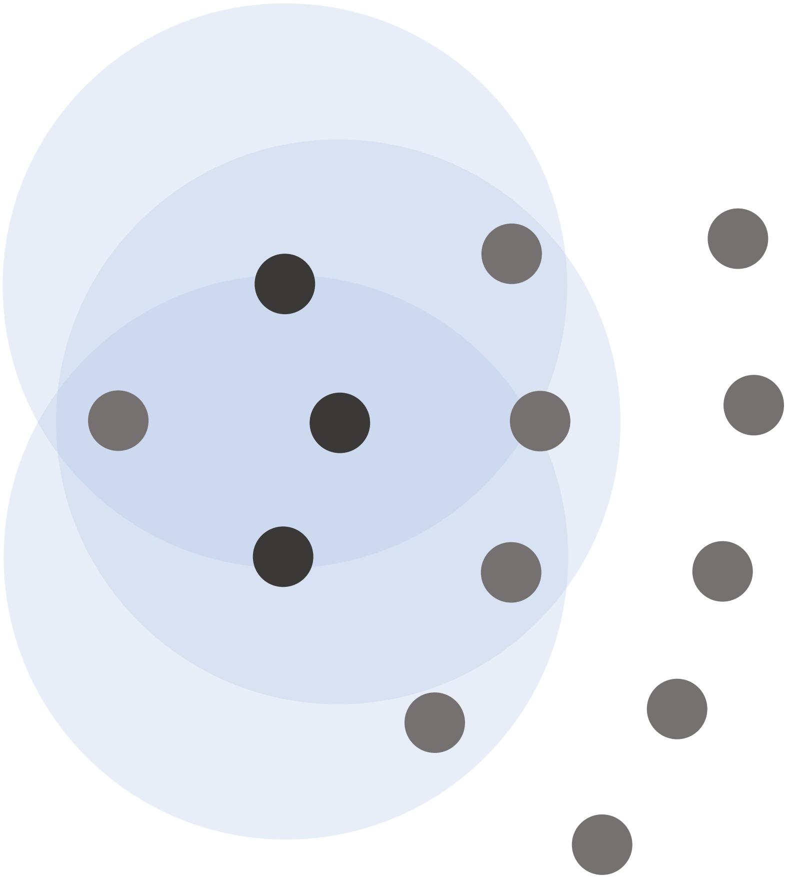


Ferrari, Zimmerling, Thiele, Saukh, ACM/IEEE IPSN, 2012. Best Paper Award.

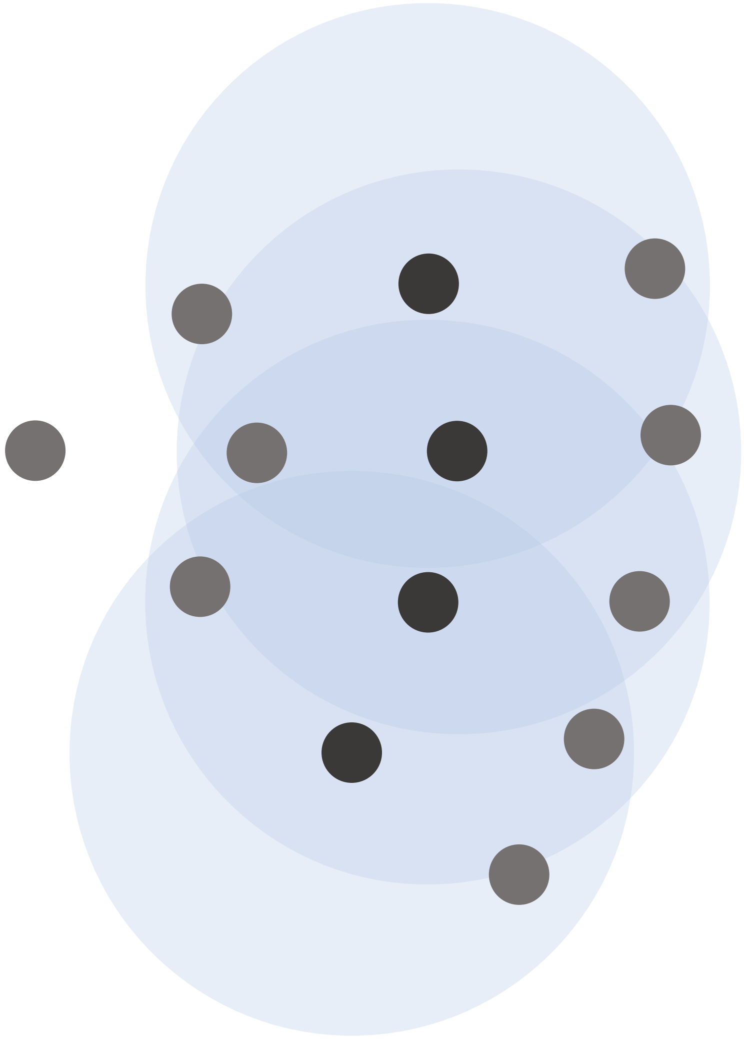
One node starts the flood by transmitting the message



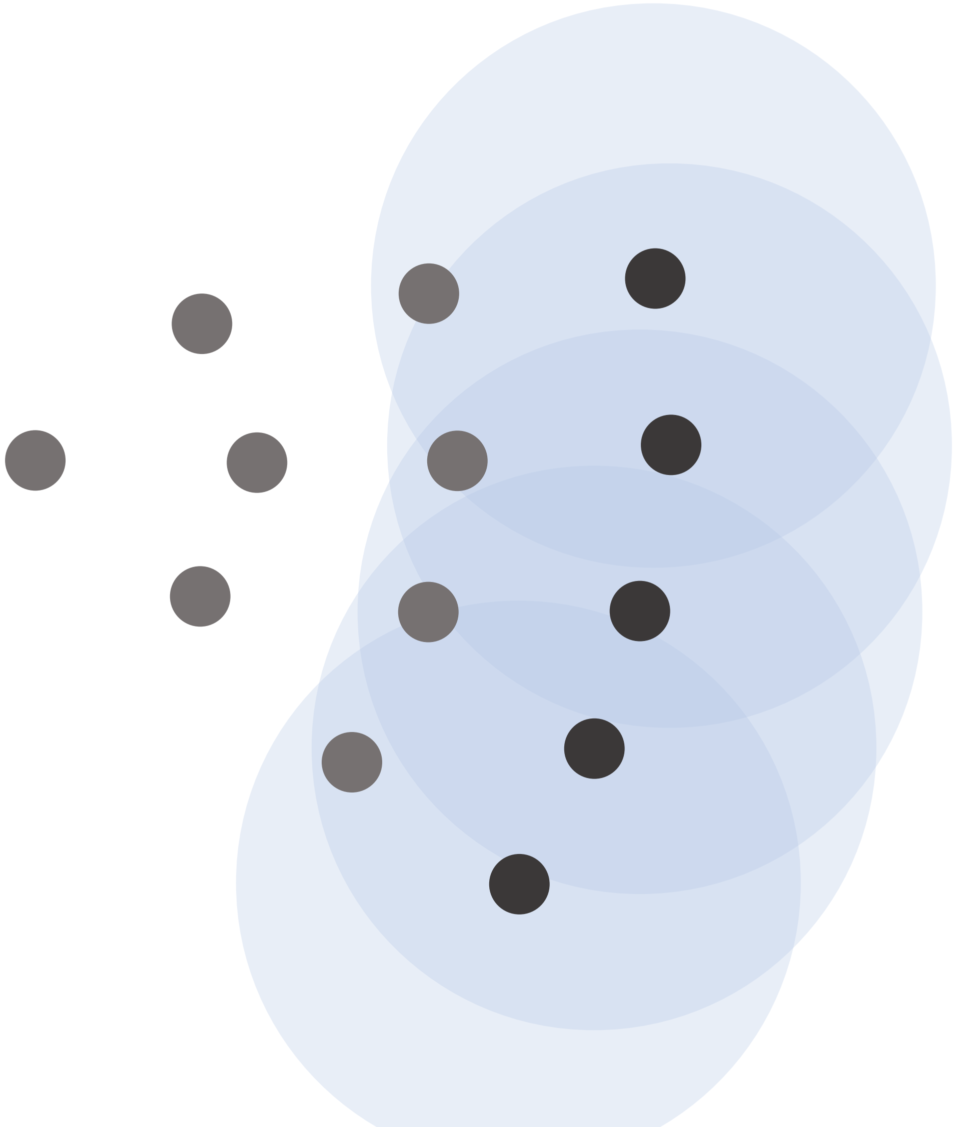
Make sure that all receivers relay the same message at the same time



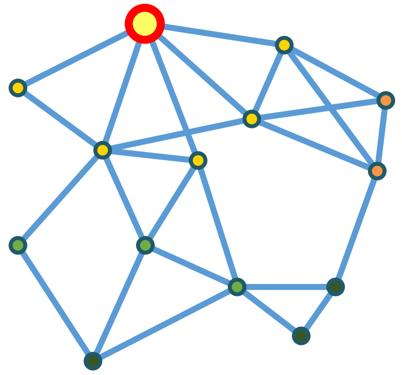
Nodes blindly relay irrespective of the network topology: no explicit routing, no network state



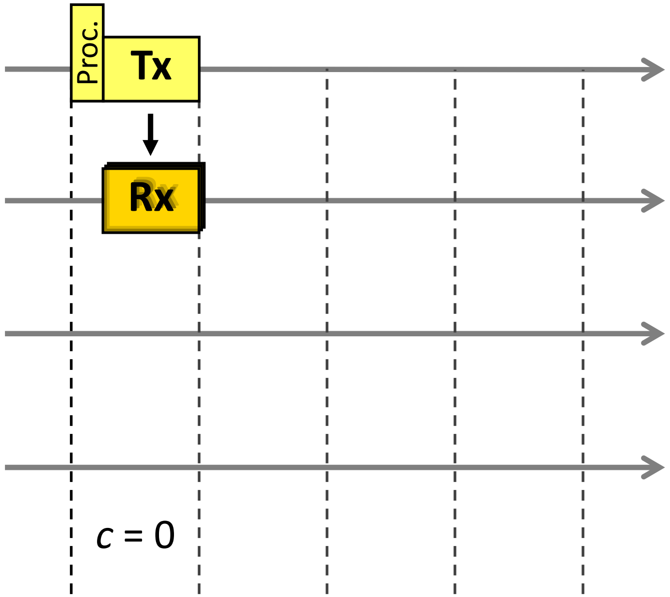
Nodes blindly relay irrespective of the network topology: no explicit routing, no network state

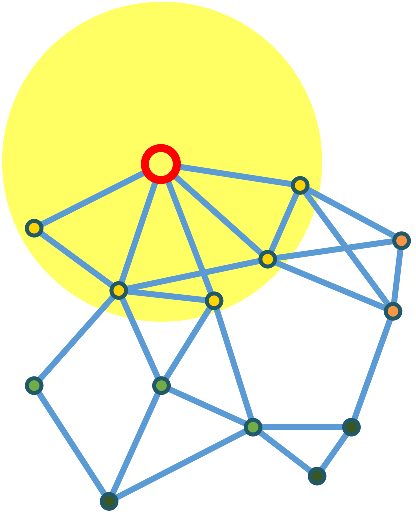


If requested, Glossy also time-synchronizes the entire network at nearly no additional cost

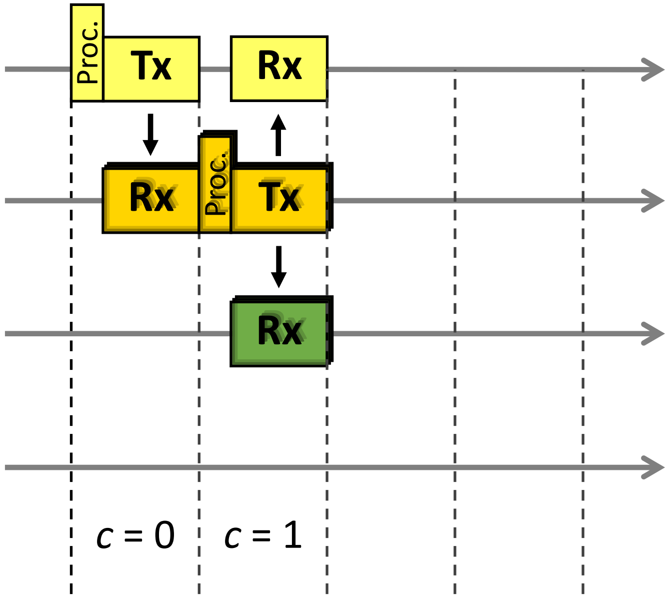


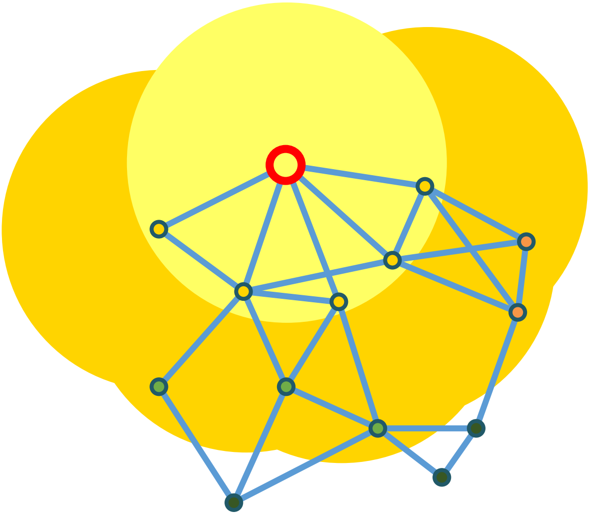
The message contains a *relay counter c* that is set to 0 before the flood starts



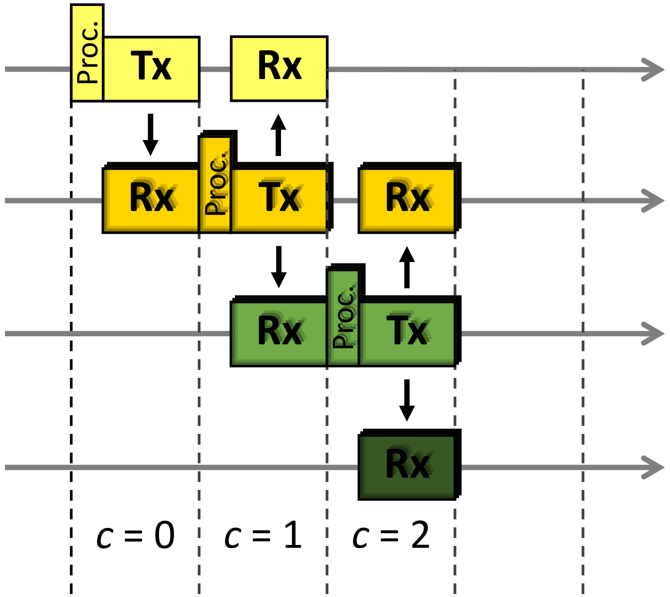


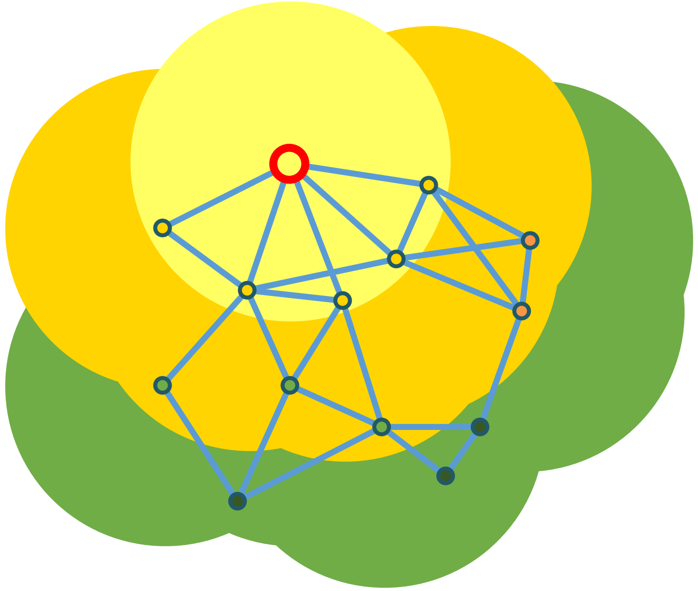
Nodes increment *c* before relaying the message



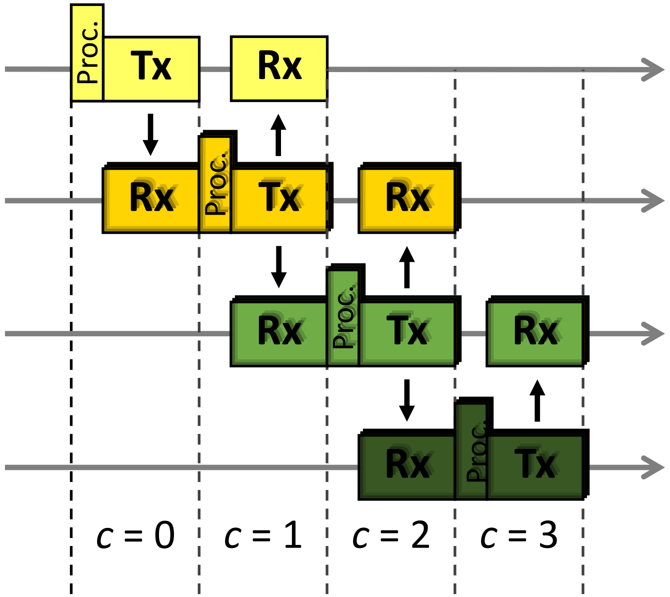


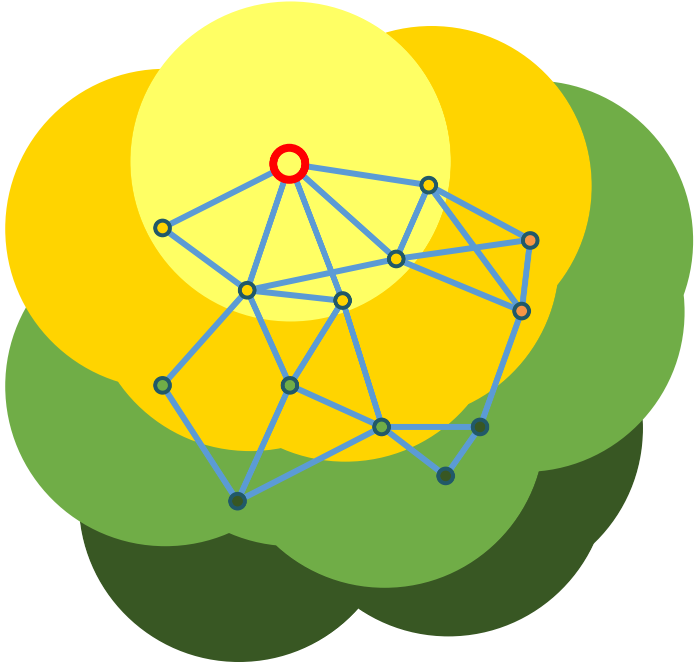
Nodes increment *c* before relaying the message



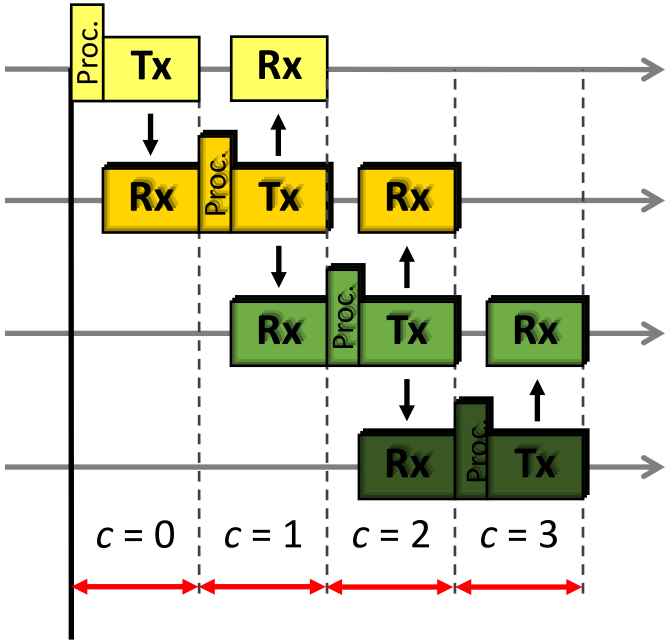


Nodes increment *c* before relaying the message



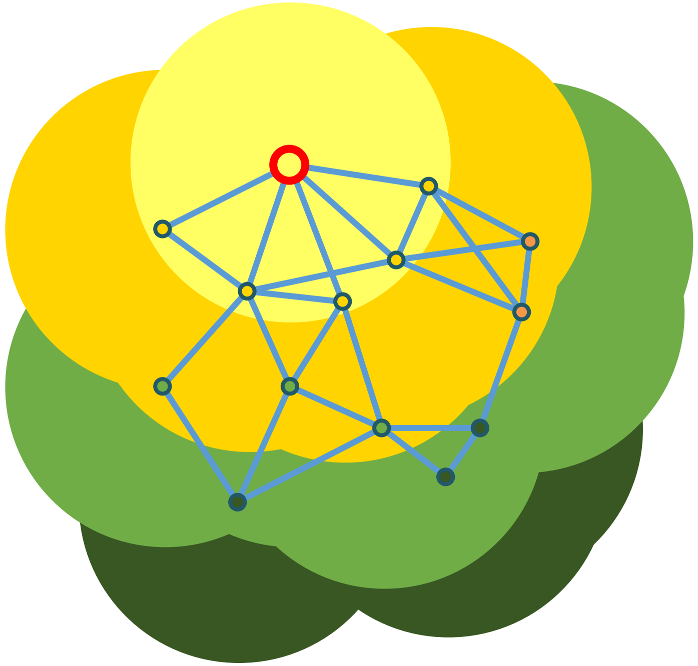


Glossy can synchronize all nodes to within sub-microsecond accuracy

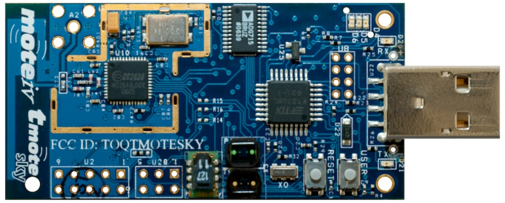


Compute common reference time

Estimate constant relay duration



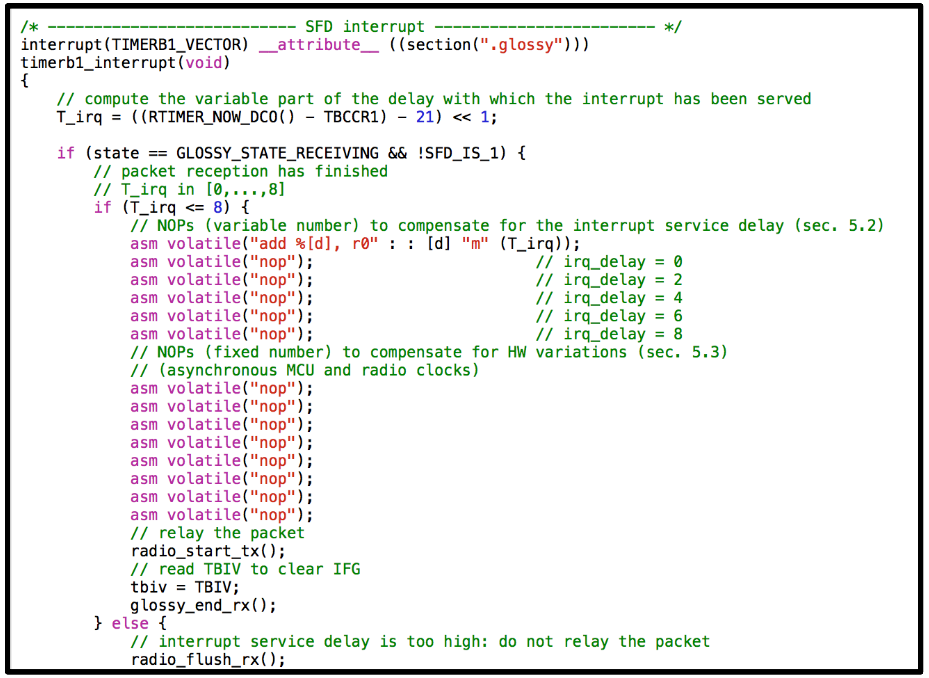
Can this be implemented on standard hardware? Yes (*)



TelosB

CC430 SoC

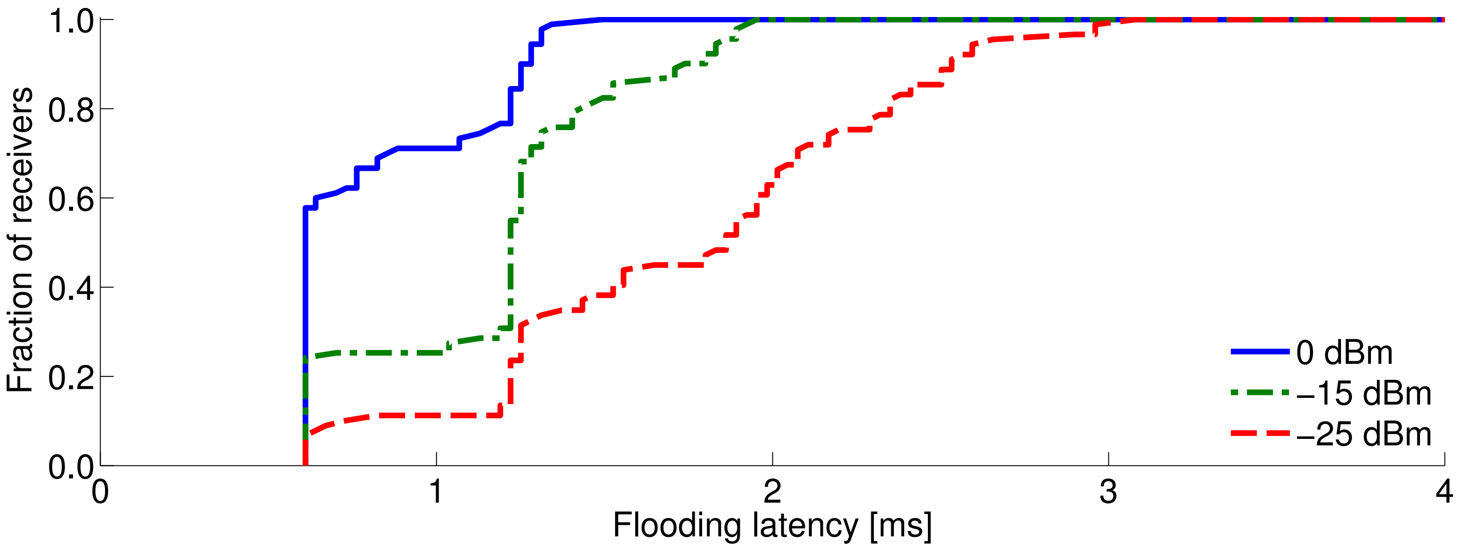
Is it trivial? No



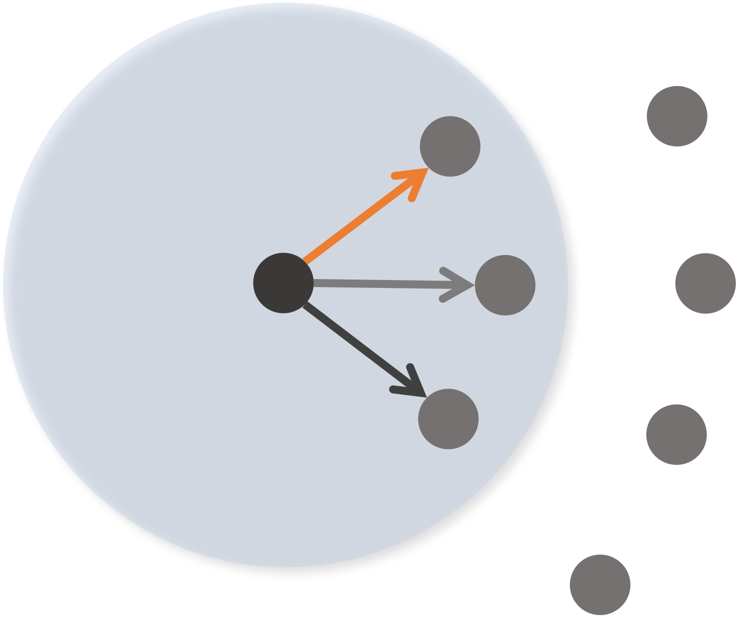
```
/* --------------------------- SFD interrupt ------------------------ */
interrupt(TIMERB1_VECTOR) __attribute__ ((section(".glossy")))
timerb1_interrupt(void)
{
    // compute the variable part of the delay with which the interrupt has been served
    T_irq = ((RTIMER_NOW_DCO() - TBCCR1) - 21) << 1;

    if (state == GLOSSY_STATE_RECEIVING && !SFD_IS_1) {
        // packet reception has finished
        // T_irq in [0,...,8]
        if (T_irq <= 8) {
            // NOPs (variable number) to compensate for the interrupt service delay (sec. 5.2)
            asm volatile("add %[d], r0" : : [d] "m" (T_irq));
            asm volatile("nop");                        // irq_delay = 0
            asm volatile("nop");                        // irq_delay = 2
            asm volatile("nop");                        // irq_delay = 4
            asm volatile("nop");                        // irq_delay = 6
            asm volatile("nop");                        // irq_delay = 8
            // NOPs (fixed number) to compensate for HW variations (sec. 5.3)
            // (asynchronous MCU and radio clocks)
            asm volatile("nop");
            asm volatile("nop");
            asm volatile("nop");
            asm volatile("nop");
            asm volatile("nop");
            asm volatile("nop");
            asm volatile("nop");
            asm volatile("nop");
            // relay the packet
            radio_start_tx();
            // read TBIV to clear IFG
            tbiv = TBIV;
            glossy_end_rx();
        } else {
            // interrupt service delay is too high: do not relay the packet
            radio_flush_rx();
```

(*) https://github.com/ETHZ-TEC/LWB

Flood 8-byte packet in 92-node network (up to 5 hops) in <3 ms and with a reliability of >99.99 % across all settings



By harnessing different types of diversity, Glossy is highly reliable and also highly resilient to network state changes

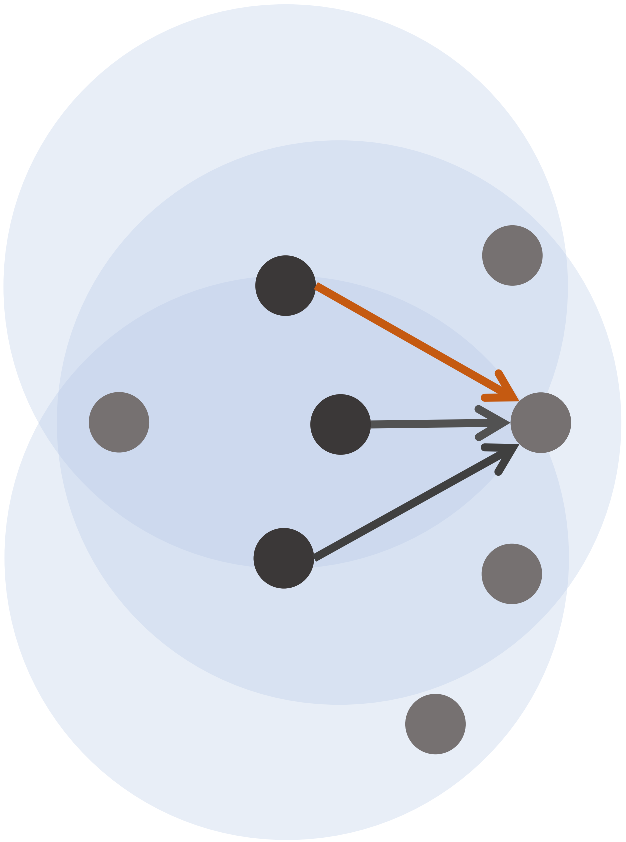


Receiver diversity

Sender diversity

Spatio-temporal diversity

By harnessing different types of diversity, Glossy is highly reliable and also highly resilient to network state changes

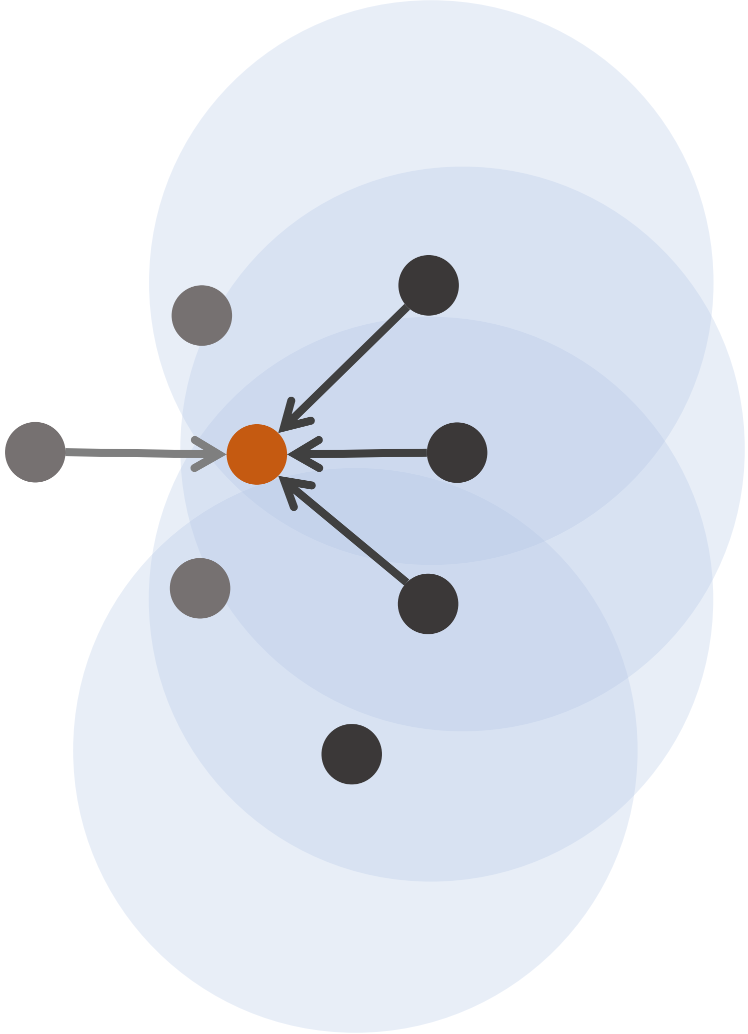


Receiver diversity

Sender diversity

Spatio-temporal diversity

By harnessing different types of diversity, Glossy is highly reliable and also highly resilient to network state changes

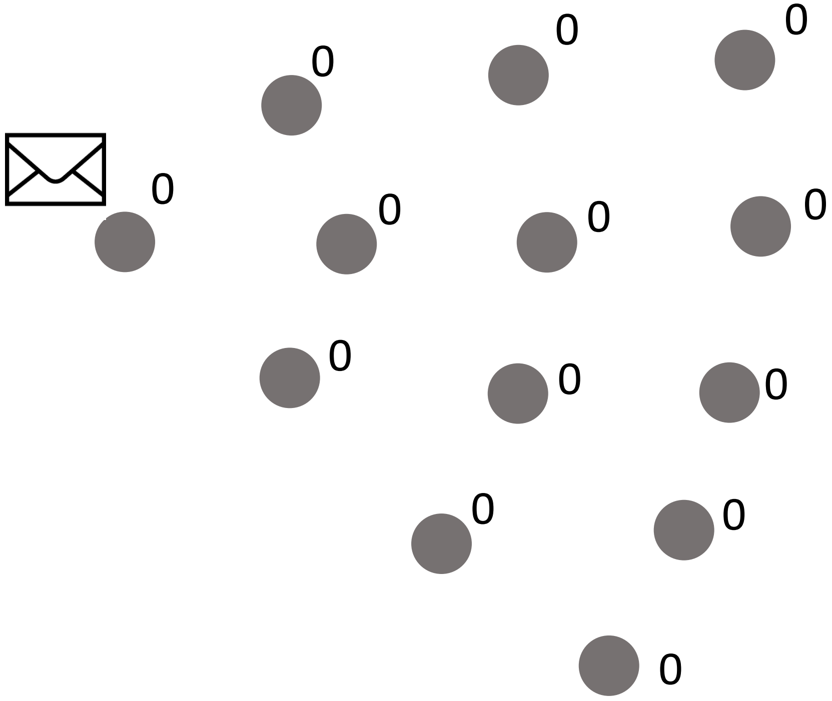


Receiver diversity

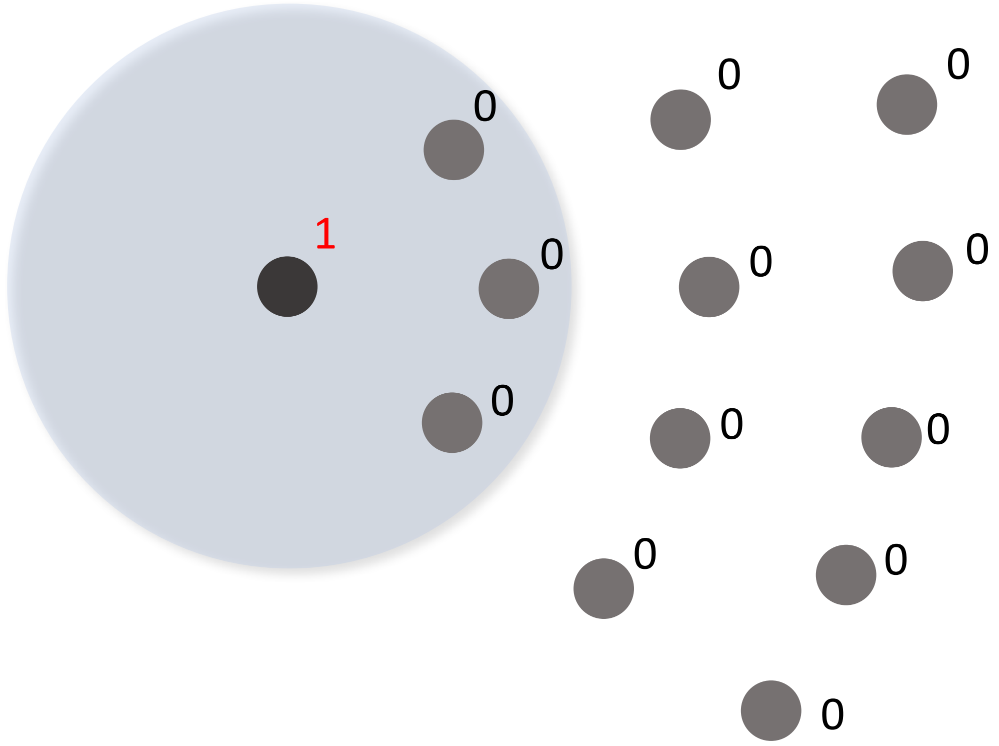
Sender diversity

Spatio-temporal diversity

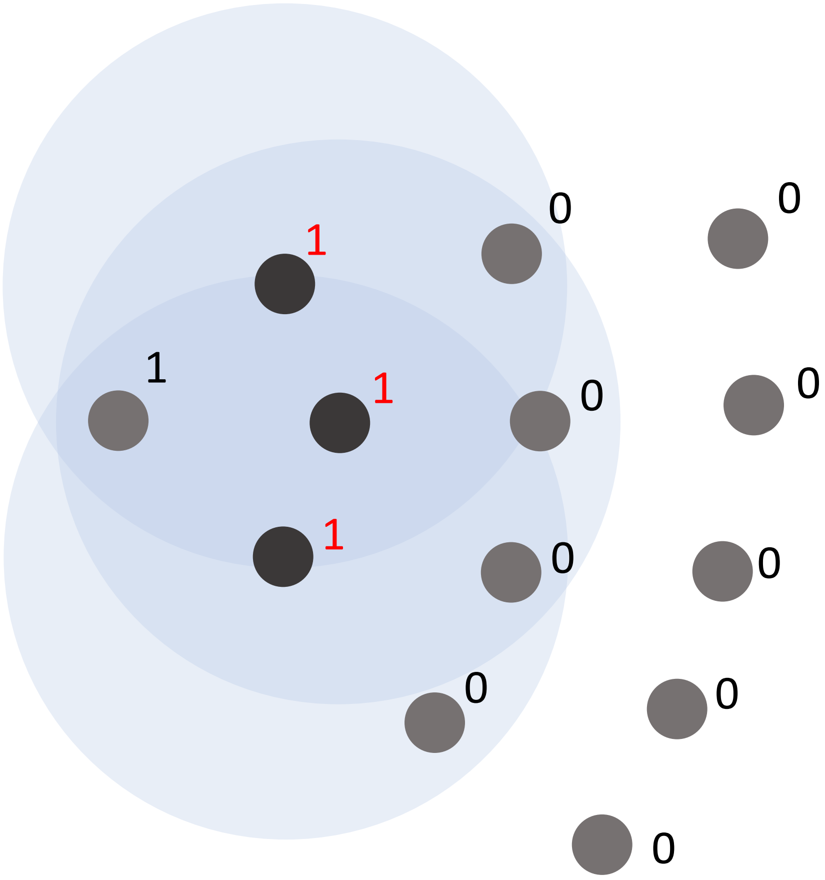
In fact, each node transmits up to N times during a flood (here 2), generating multiple “waves” that propagate through the network



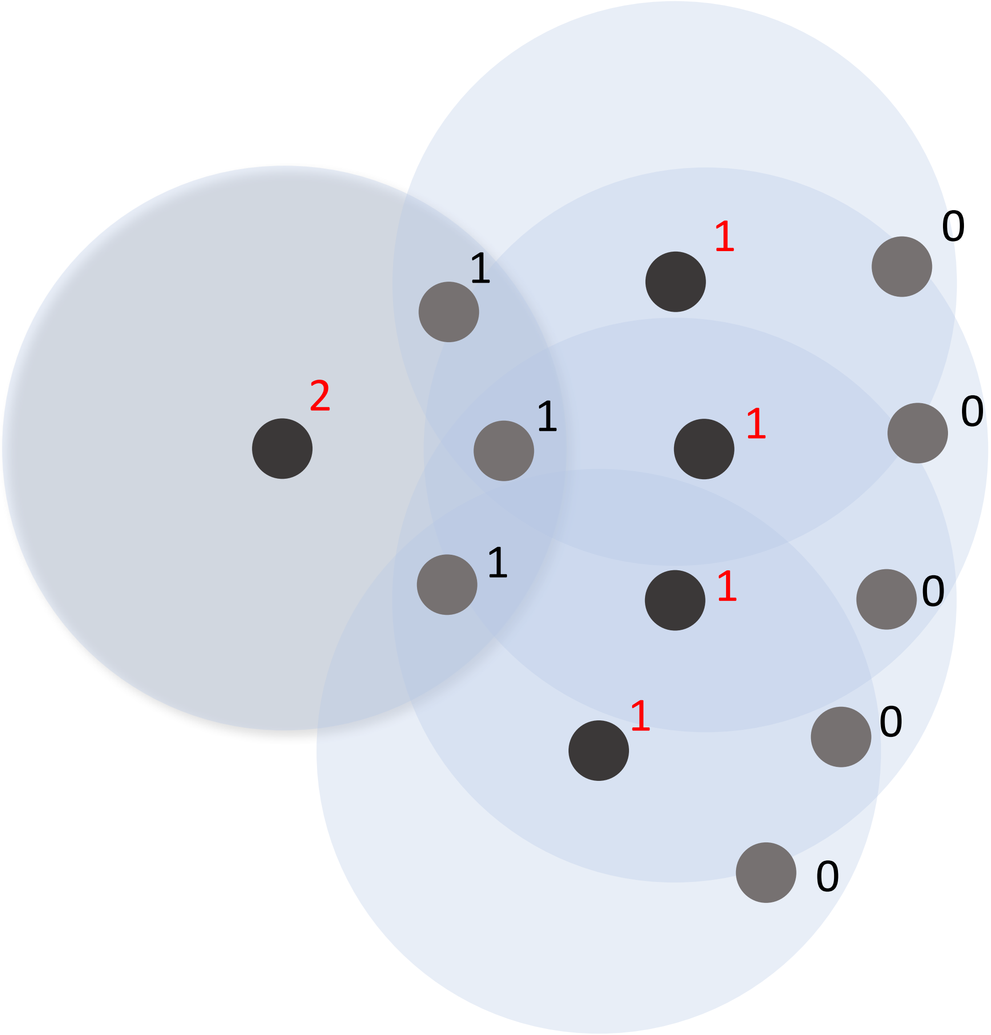
In fact, each node transmits up to N times during a flood (here 2), generating multiple "waves" that propagate through the network



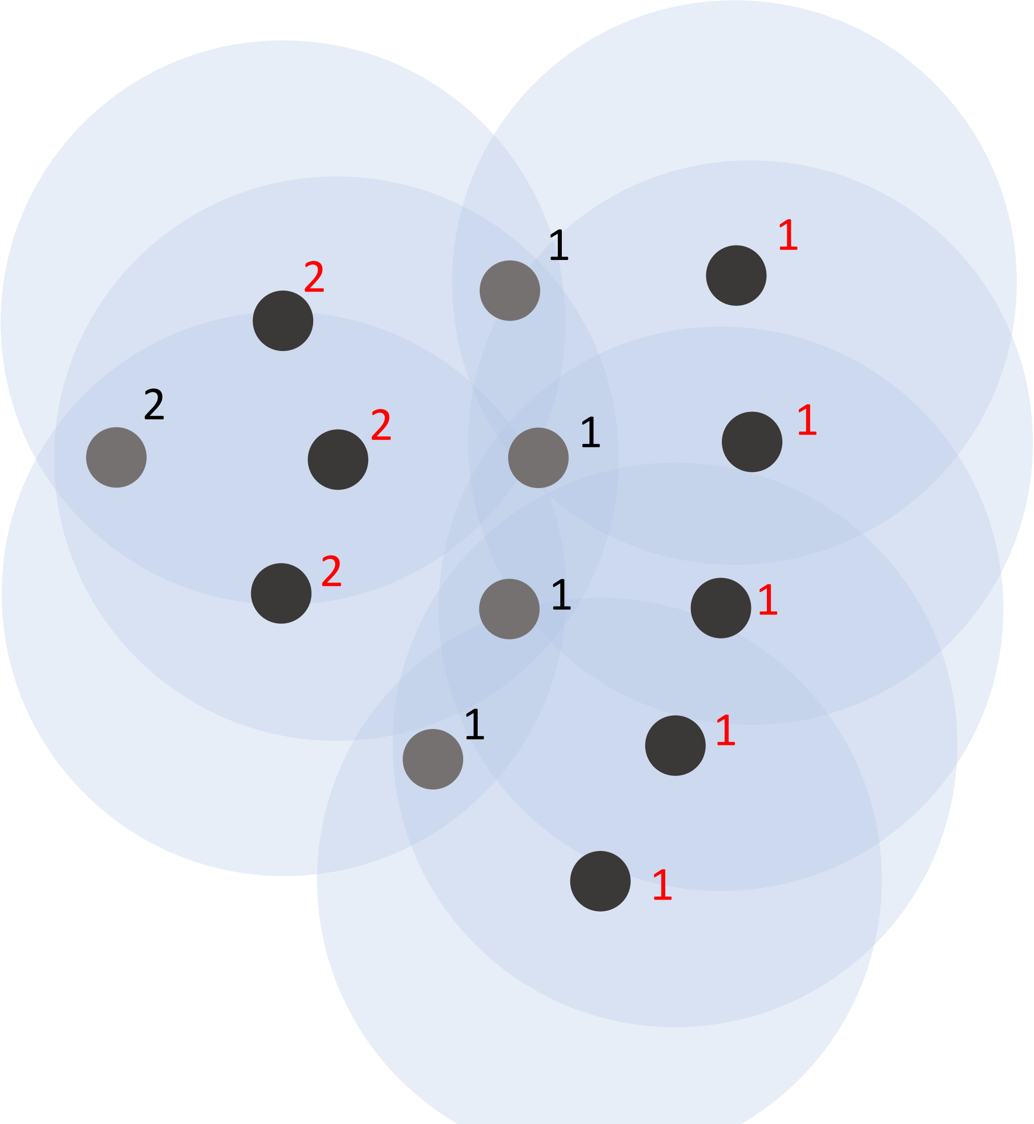
In fact, each node transmits up to N times during a flood (here 2), generating multiple "waves" that propagate through the network



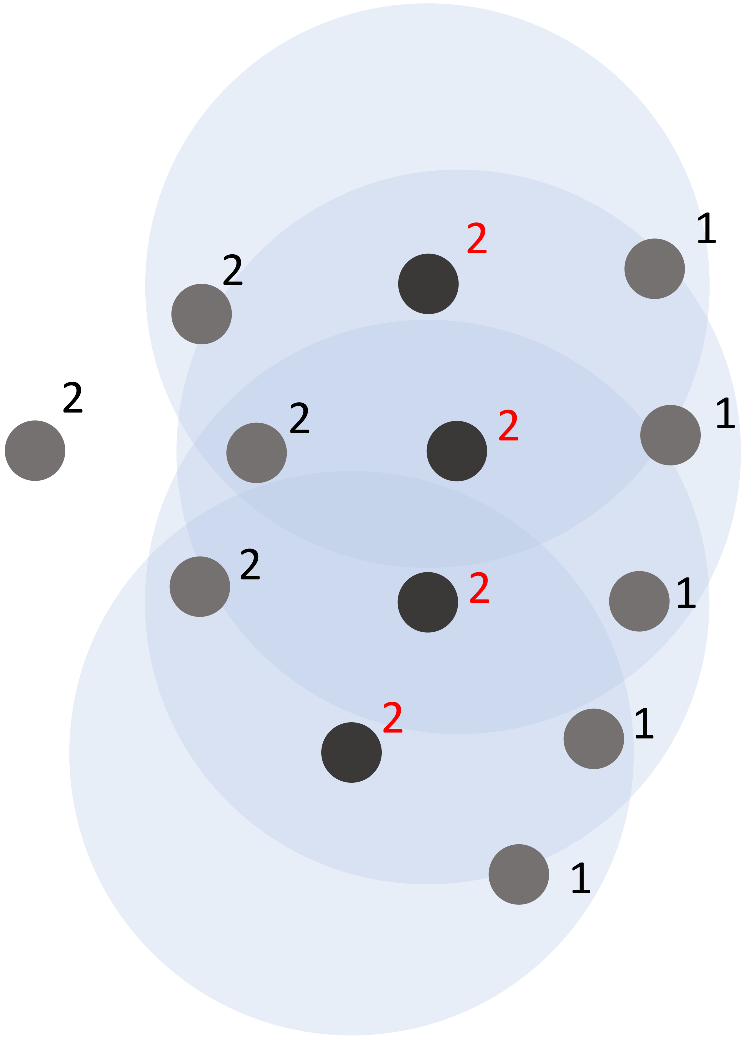
In fact, each node transmits up to N times during a flood (here 2), generating multiple “waves” that propagate through the network



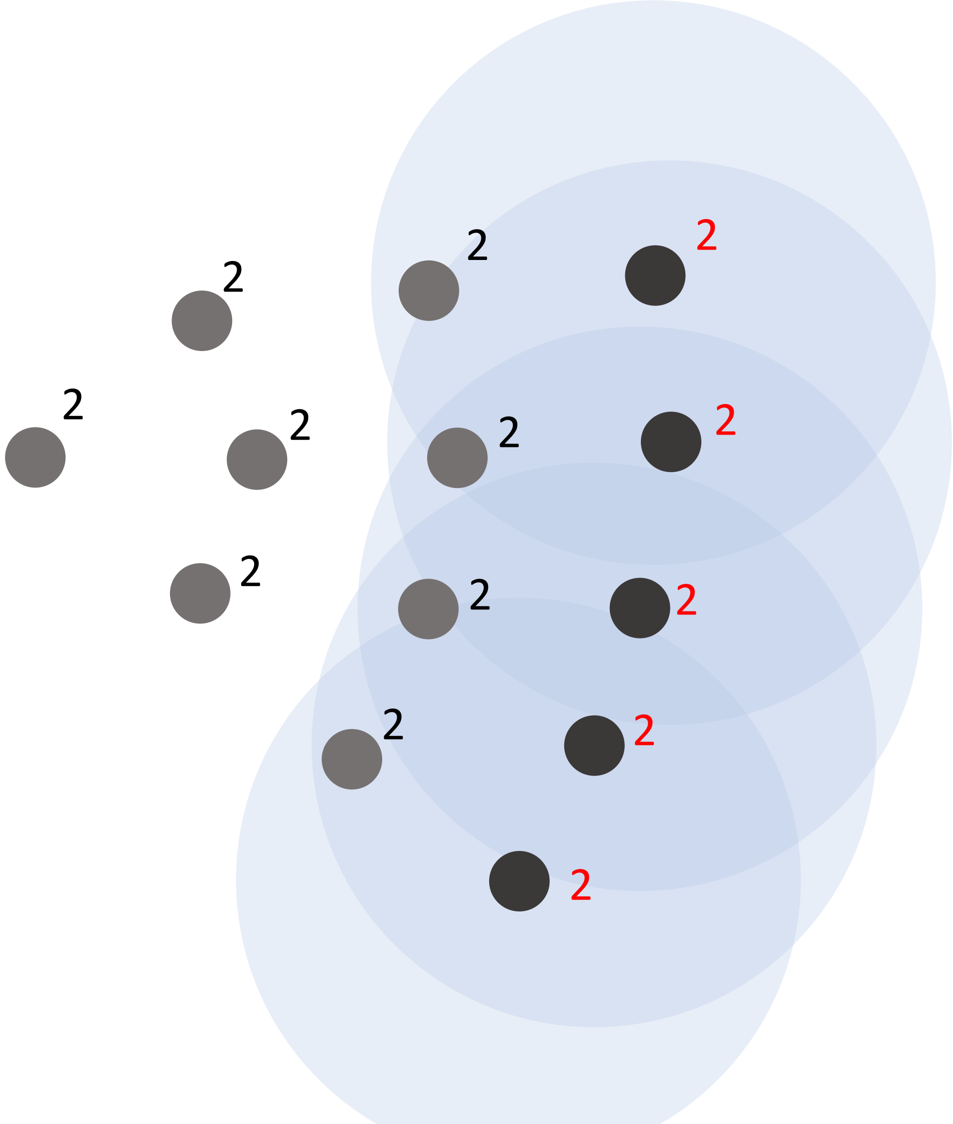
In fact, each node transmits up to N times during a flood (here 2), generating multiple “waves” that propagate through the network



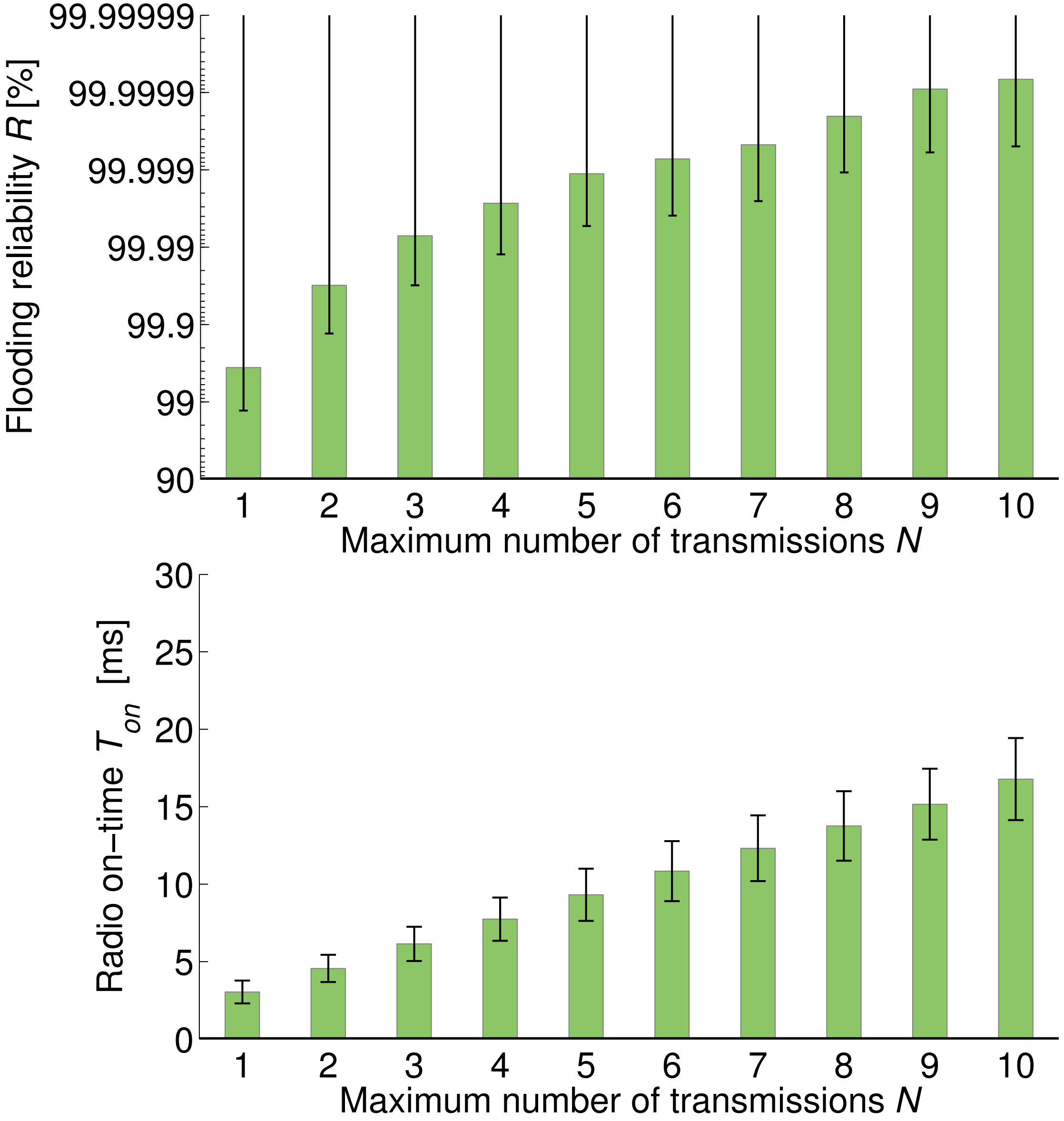
In fact, each node transmits up to N times during a flood (here 2), generating multiple "waves" that propagate through the network



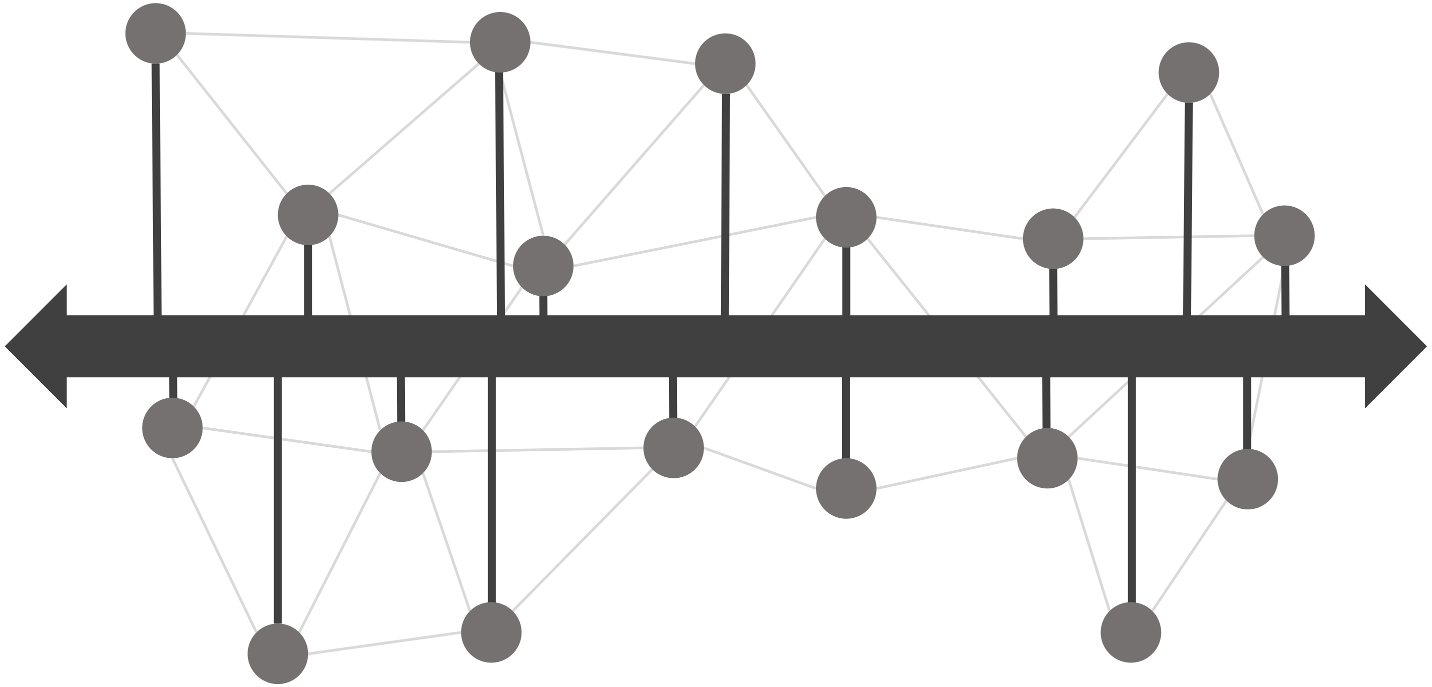
In fact, each node transmits up to N times during a flood (here 2), generating multiple “waves” that propagate through the network



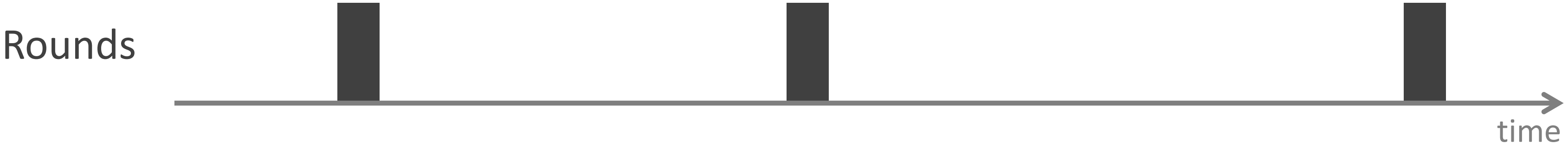
Increasing the maximum number of transmissions N boosts reliability at the expense of longer radio on-time



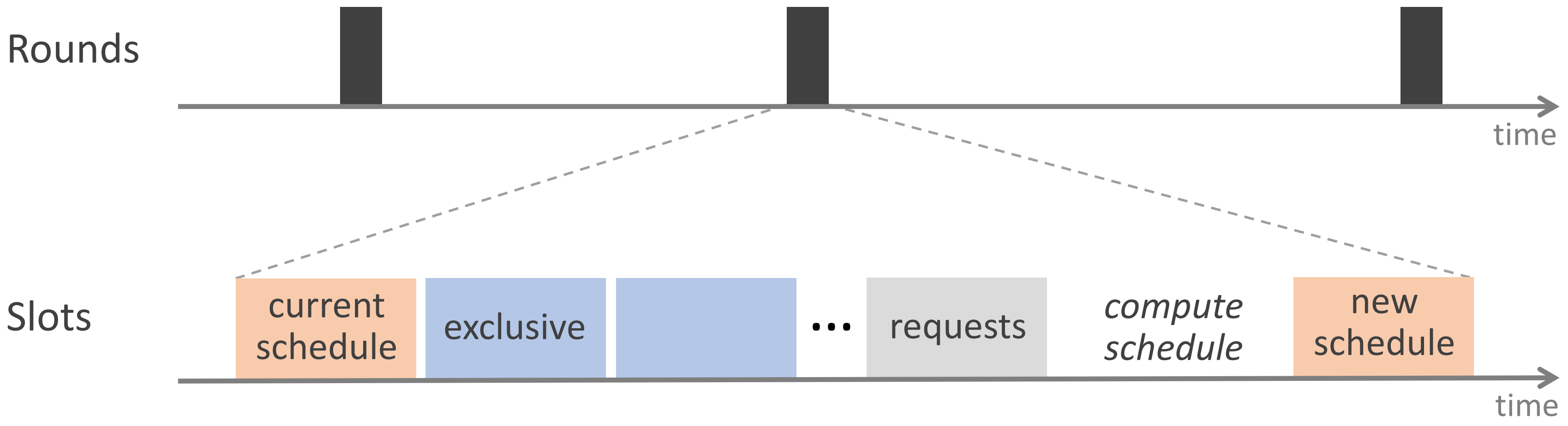
How does Glossy help build a wireless bus?



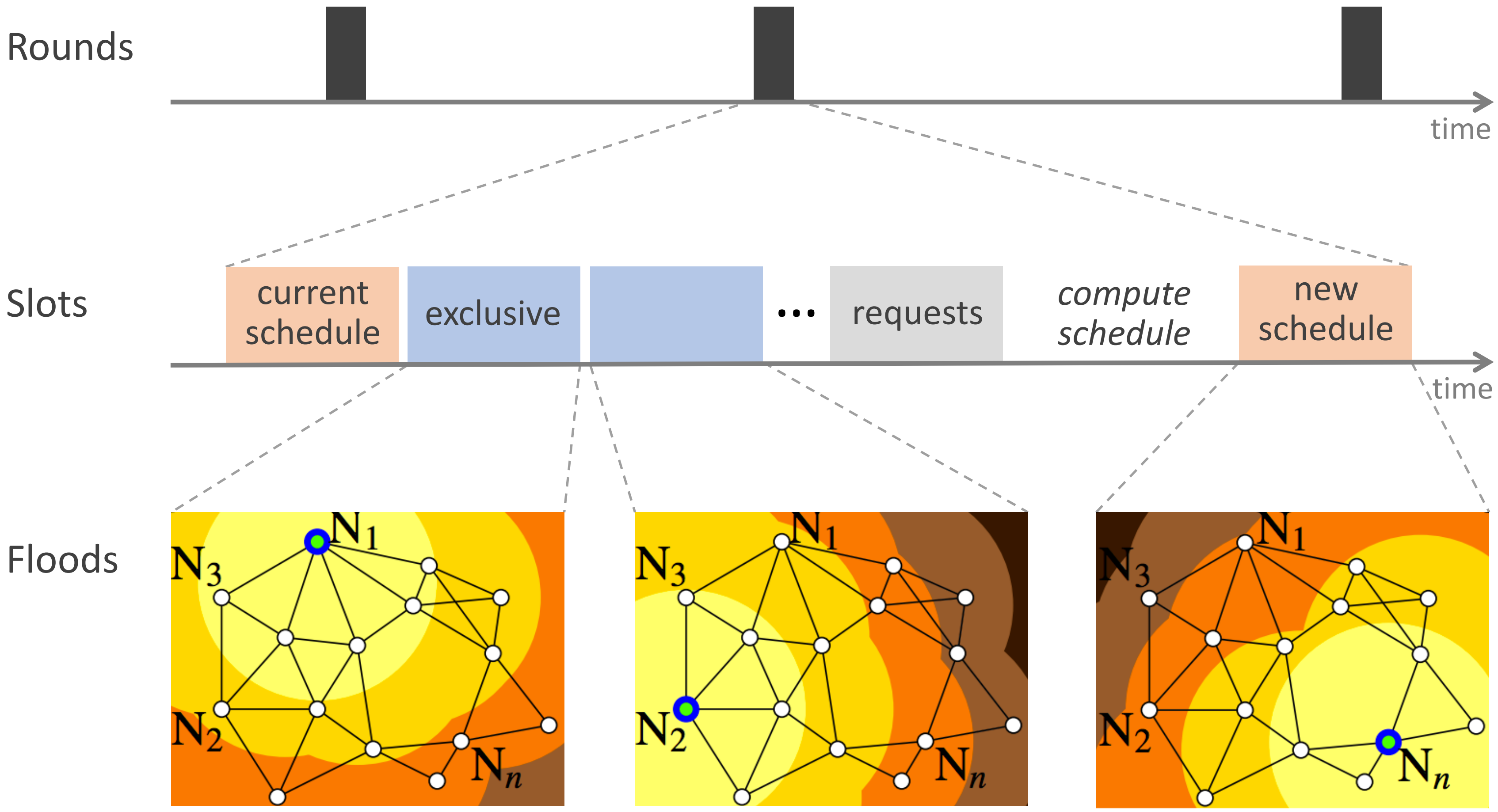
Globally time-triggered communication



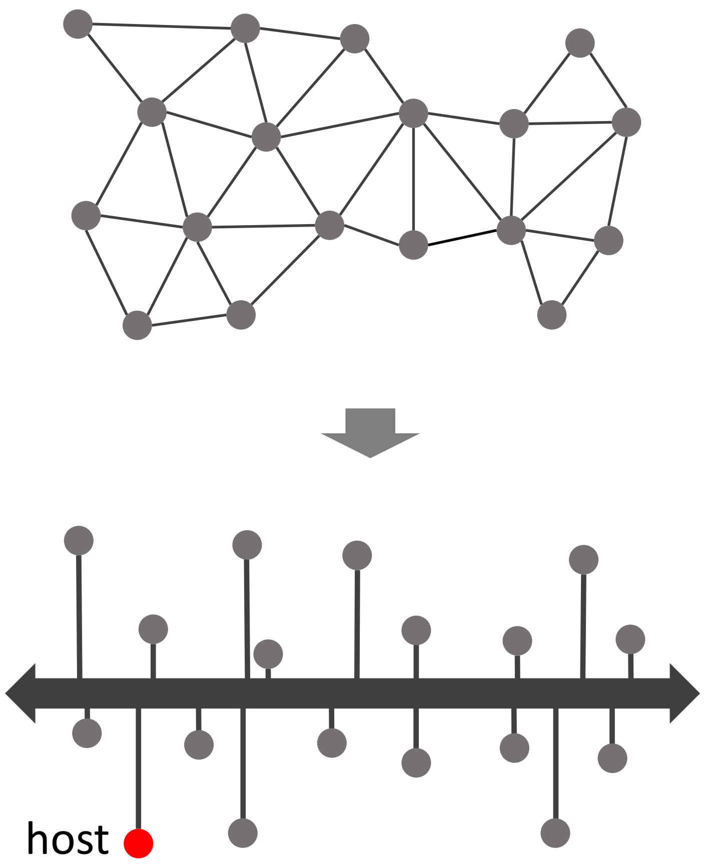
Globally time-triggered communication



Globally time-triggered communication



Turning a multi-hop low-power wireless network into a shared bus



Only one-to-all network flooding (Glossy)

All nodes can receive all messages

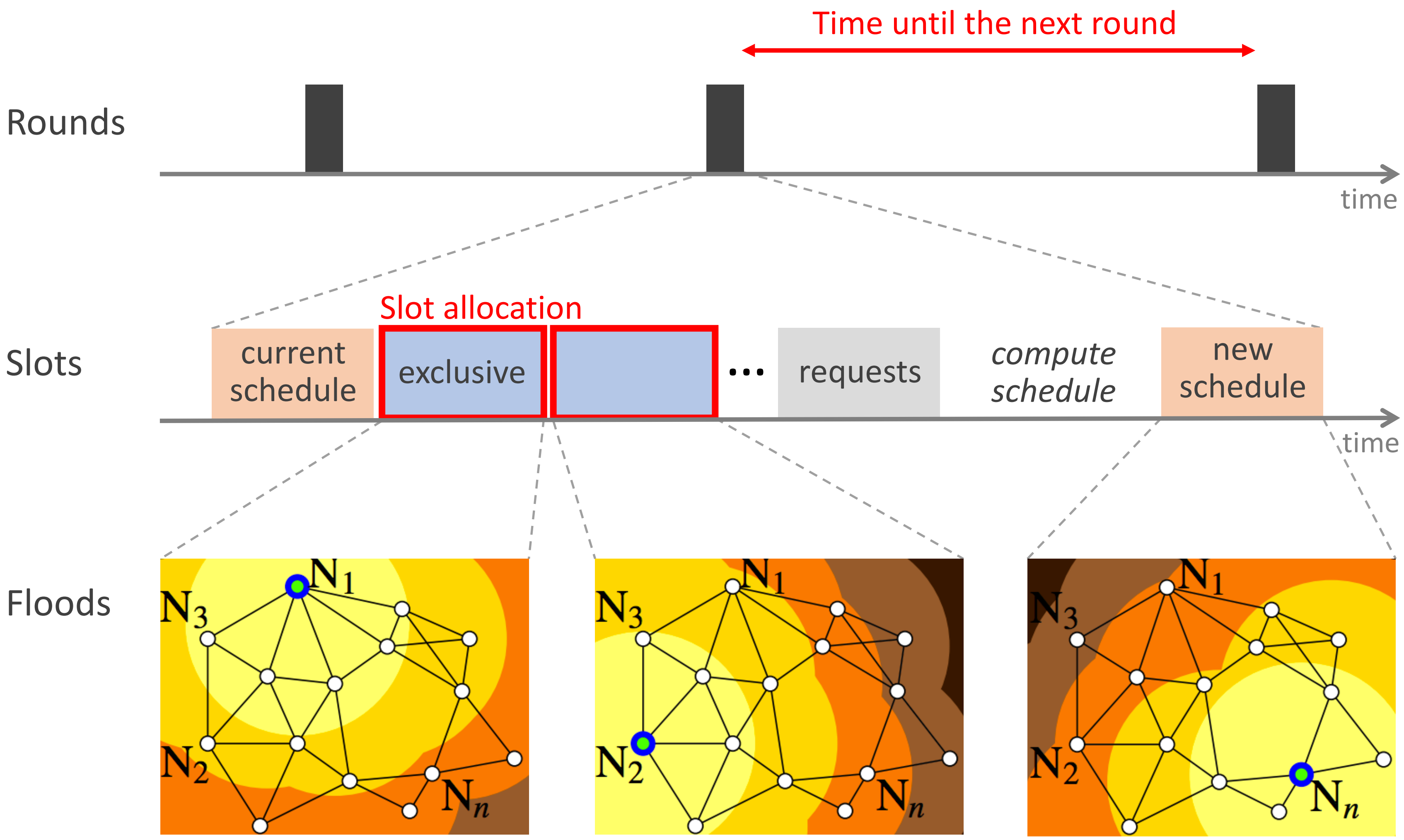
Globally time-triggered communication

as a function of a single network-wide schedule and the current global time

Adaptive online scheduling

only based on application requirements, specified in the form of periodic streams

Scheduler determines at runtime slot allocation and start time of the next round



Time until the next round

Rounds

time

Slot allocation

Slots

current schedule

exclusive

…

requests

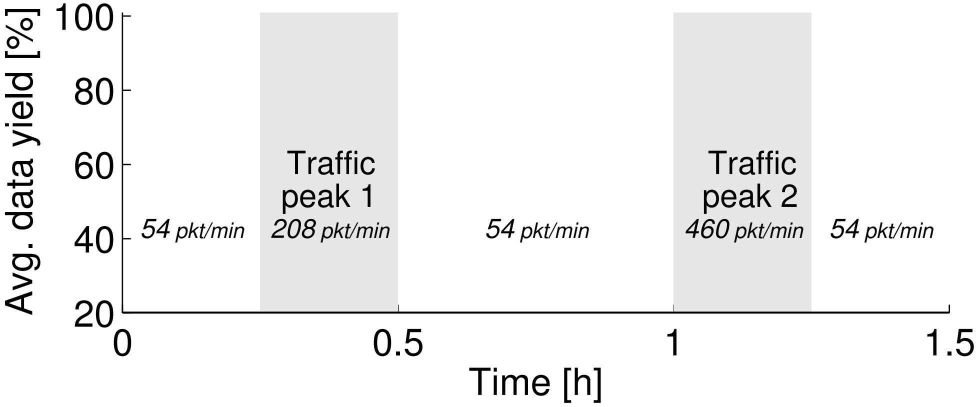
compute schedule

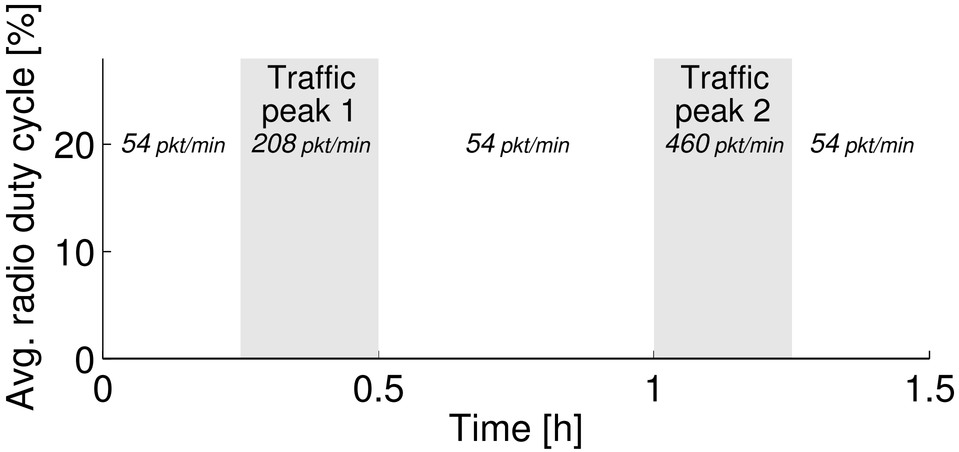
new schedule

time

Floods

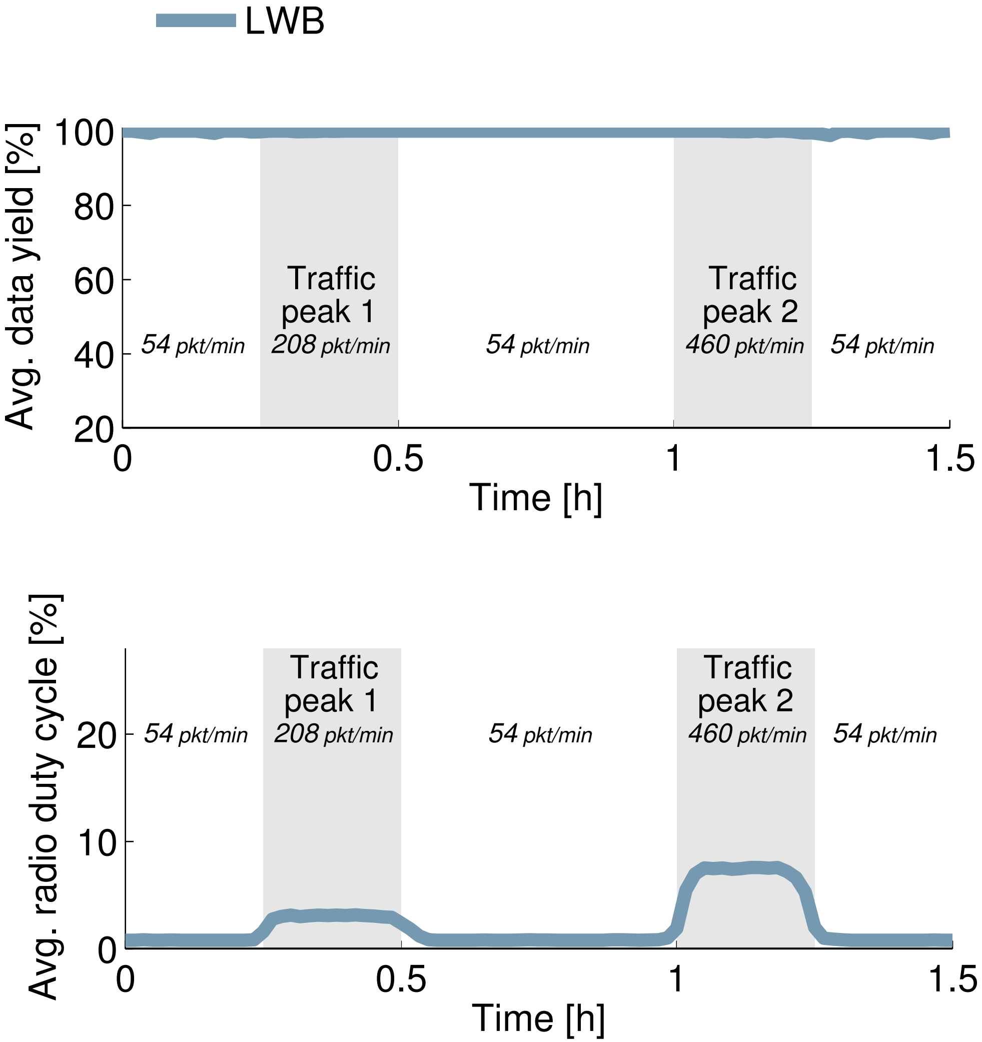
Energy-efficient scheduler (LWB): prompt adaptation to varying traffic load





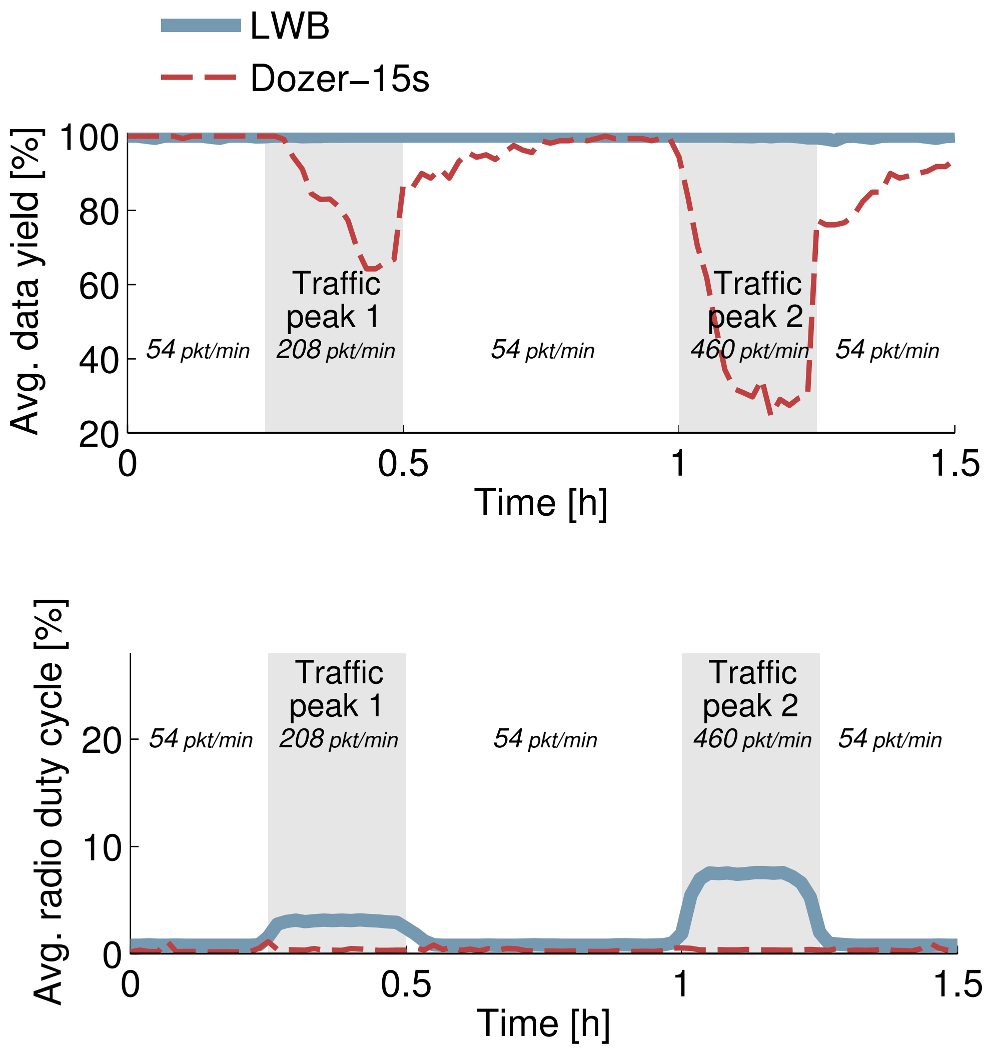
1 destination
54 sources
5 hops

Energy-efficient scheduler (LWB): prompt adaptation to varying traffic load



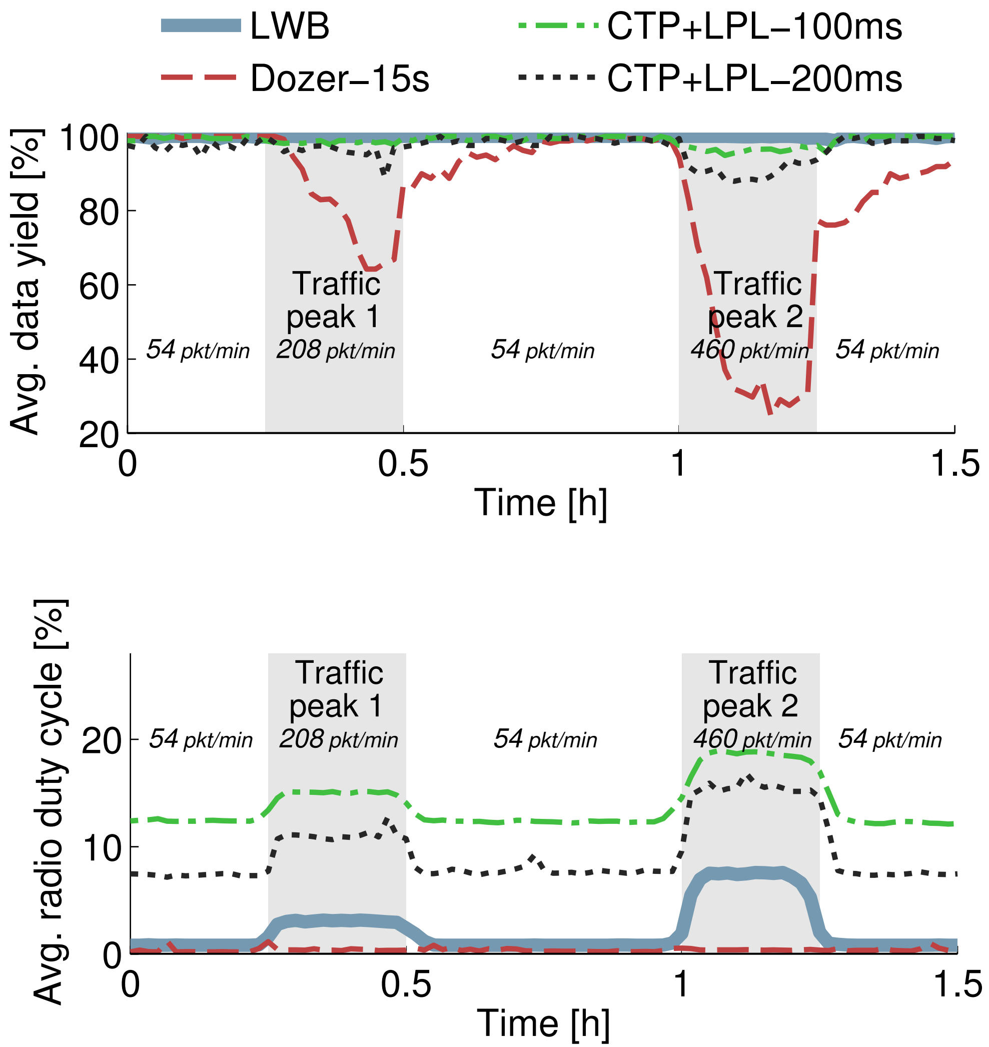
1 destination
54 sources
5 hops

Energy-efficient scheduler (LWB): prompt adaptation to varying traffic load



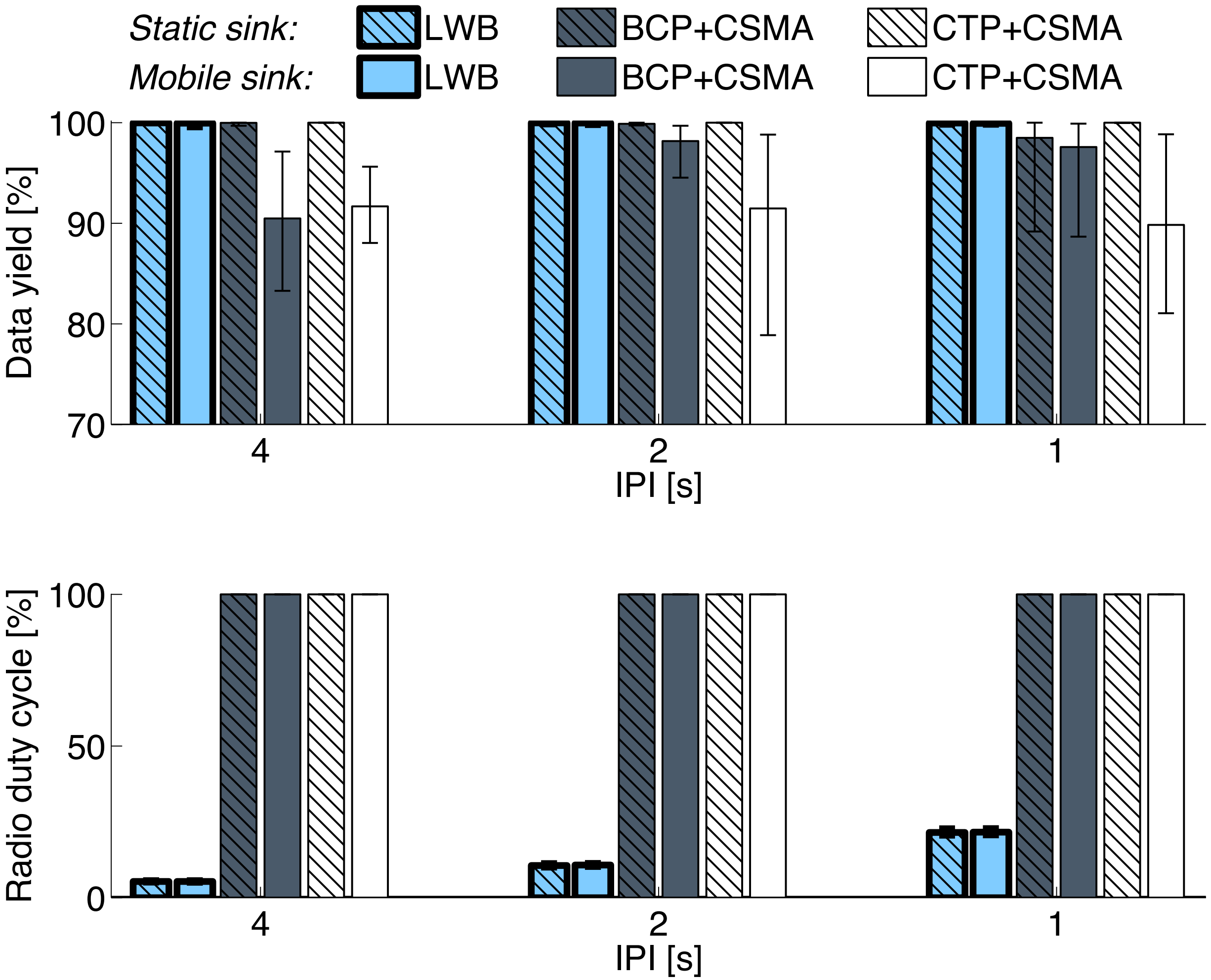
1 destination
54 sources
5 hops

Energy-efficient scheduler (LWB): prompt adaptation to varying traffic load



1 destination
54 sources
5 hops

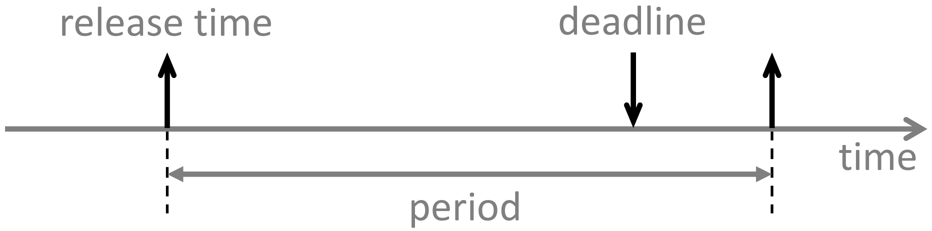
Energy-efficient scheduler (LWB): resilience to network state changes



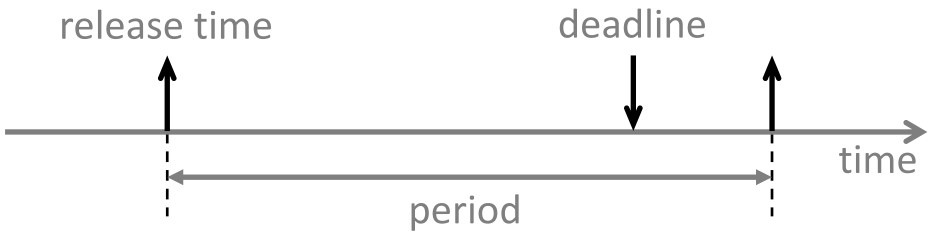
1 mobile destination
25 sources
3 hops

What else can we do with it?

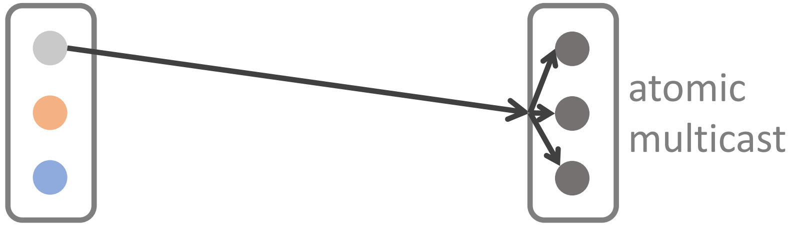
Provide hard real-time guarantees



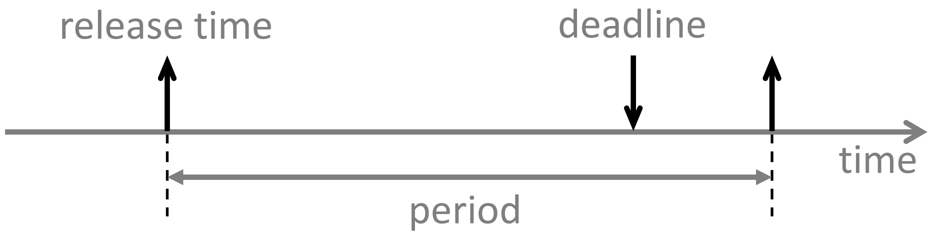
Provide hard real-time guarantees



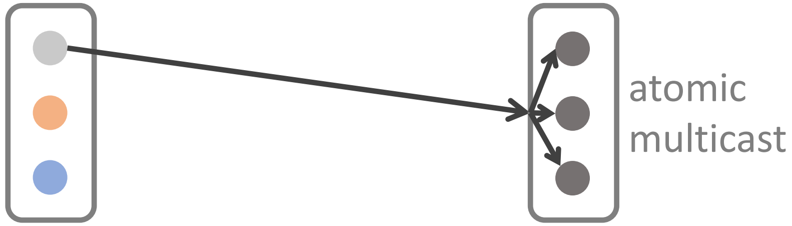
Provide virtual-synchrony guarantees



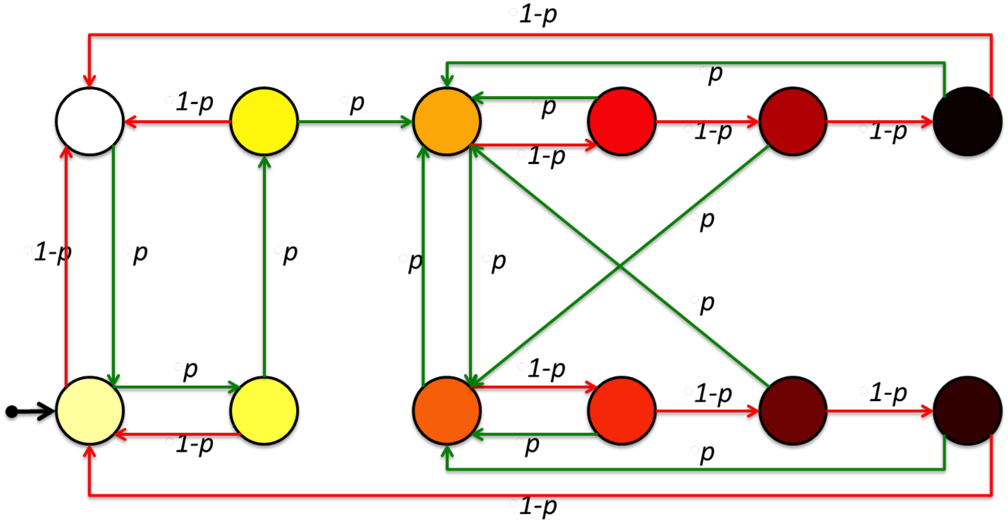
Provide hard real-time guarantees



Provide virtual-synchrony guarantees



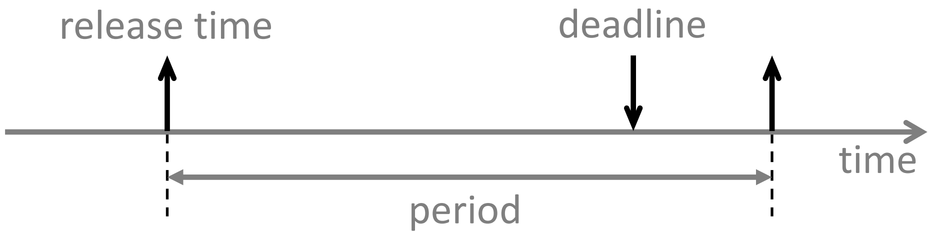
Build highly accurate performance models



🟩 = schedule received, with probability *p*

🟥 = schedule missed, with probability *1-p*

Provide hard real-time guarantees

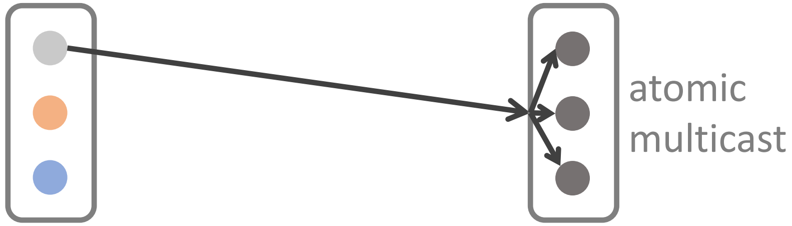

release time

deadline

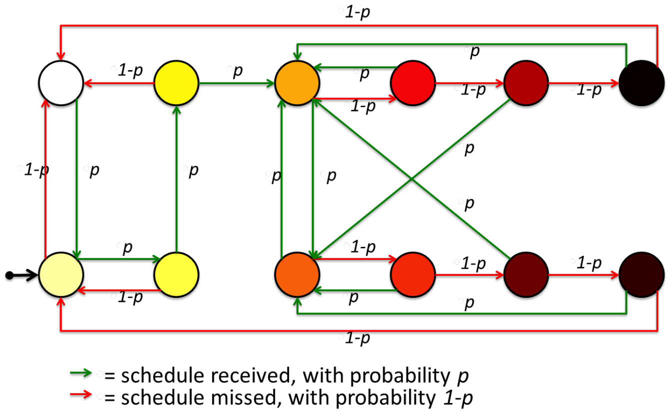
time

period

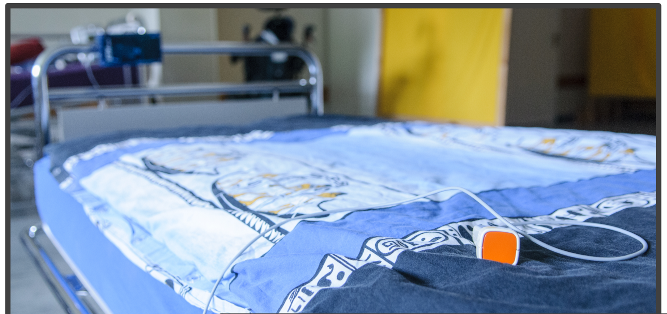
Provide virtual-synchrony guarantees


atomic multicast

Build highly accurate performance models



= schedule received, with probability *p*
= schedule missed, with probability *1-p*



Deploy as nurse call system in a summer camp for teenagers with muscular dystrophy

Summary of Today's Lecture

- Duty cycling is a key mechanism to achieve energy-efficient operation
  - Low-power MAC protocols duty cycle the radio to reduce communication energy consumption
  - Duty cycling other components (sensors, MCU, …) makes sense, too, and may in some cases be more effective than duty cycling the radio
- Synchronous transmissions enable a new class of protocols
  - Nearly stateless, fast, highly reliable flooding (Glossy)
  - Wireless bus with properties similar to wired buses (e.g., real-time guarantees)