Scheduling Transmission of Bulk Data in Sensor Networks using a Dynamic TDMA Protocol

Volker Turau and Christoph Weyer

Institute of Telematics
Hamburg University of Technology

Data Intensive Sensor Networks
Mannheim, Germany, May 11, 2007

TUHH
Technische Universität Hamburg-Harburg

Introduction

- Sensor networks are increasingly used in applications where sensors periodically measure data with high rates
- Problem: Reliable transport of high volumes of sampled data through an unreliable multi-hop network
- Difficulties derive from the tight resources
  - Limited storage space to buffer intermediate packets
  - Bounded energy reserve

Motivation

- Application measures water pressure in tideland area
- Wireless sensor network takes a sample every 100 ms
- Batteries cannot be replaced easily
- Data routed towards sink & transmitted via GSM to central location
- No communication during flood tide (6 - 8 hours)
- Data is stored in EEPROM of each node during flood tide
- During ebb tide, nodes forward stored data to sink

Problem

How to transport the data reliably and efficiently using low power?

Motivation

- Application measures water pressure in tideland area
- Wireless sensor network takes a sample every 100 ms
- Batteries cannot be replaced easily
- Data routed towards sink & transmitted via GSM to central location
- No communication during flood tide (6 - 8 hours)
- Data is stored in EEPROM of each node during flood tide
- During ebb tide, nodes forward stored data to sink

Problem

How to transport the data reliably and efficiently using low power?

Reliable Data Forwarding

- Starting position:
  - Multi-hop network
  - Nodes have accumulated a large amount of data
  - Data grouped into equally sized packets stored in a queue
  - Nodes have limited buffer for intermediate packets
- Target:
  - Route data completely in minimal time and with minimal energy consumption to sink without loss of data

Outline

Major Sources of Energy Waste

- Collisions
- Overhearing
- Control packet overhead
- Idle listening
- Transmission power too high

We try to address all five issues.

Mac Protocols

CSMA:

- Does not use any topology or clock information
- Robust to any change in the network
- But: Wireless networks need some topology and time information anyway (maybe in a less accurate or reliable manner)
- Problems:
  - Contention among neighboring nodes must be resolved for every transmission
  - Performance under high contention is bad

Mac Protocols

TDMA:

- Uses topology information (neighboring & interference relations)
- Relies on synchrony among nodes
- Performance is strongly tied to accuracy of topology & synchrony
- Problems:
  - Interference relations change over time
  - Tight clock synchronization incurs overhead

Our Approach

- Simple variant of TDMA:

  At any time, only a single node uses the wireless channel
  $\Longrightarrow$ No collisions
- Every node turns on its transceiver exactly when needed
  $\Longrightarrow$ No overhearing and no idle listening
- Simple to implement
- Energy consumption is minimal (almost)
  - Nodes only listen when they truly receive a packet

Simple Reliable Forwarding Strategy

- Works on spanning tree
- Every node sends in assigned slot a packet to its parent
- Parent acknowledges packet once data is in its EEPROM
- Sending node can remove it from its storage
- Disadvantages:
  - Unlimited usage of buffer space
  - Low throughput:
    - In every round a node uses only a single slot
    - Slot of node that has forwarded all data is no longer used
- Remedy against buffer overflow: Parents advise children to suspend sending in case buffer is full (with acknowledgment)

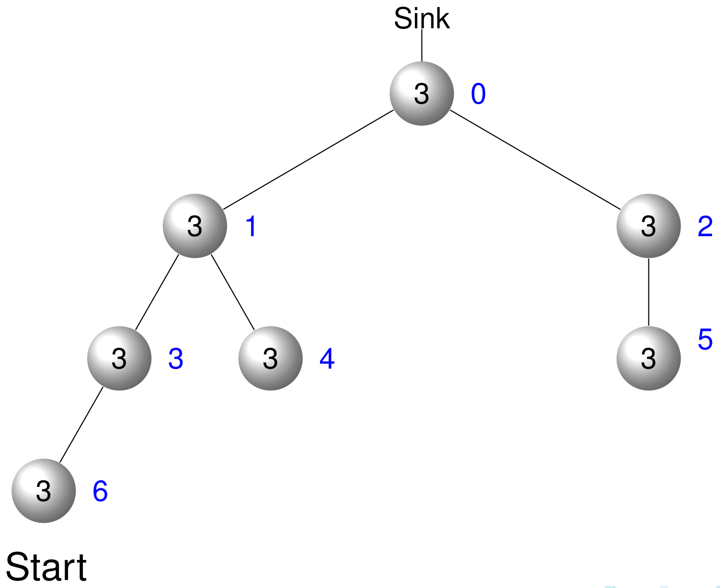
Improved Reliable Forwarding Strategy

- Remedy against low troughtput:
  - Increase number of slots used per round
  - Nodes that have forwarded all data hand over slots to parents
  - Forwarding slots in a simple bottom-up fashion leads to even more buffer overflow
  - Better strategy is needed!

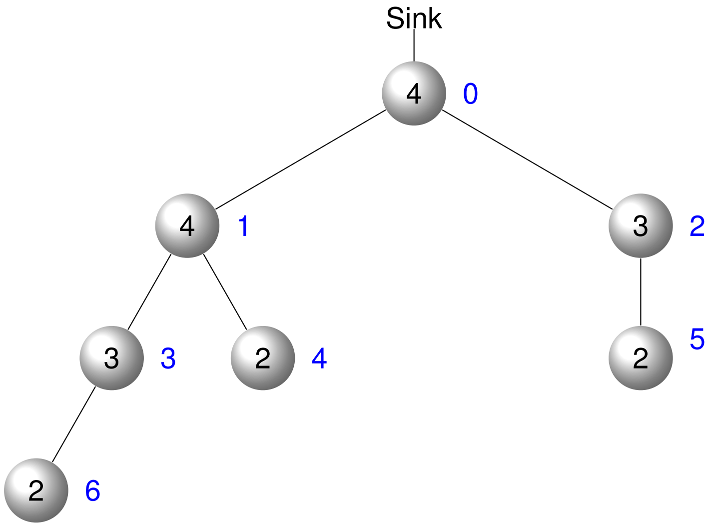
Slot Distribution Schemes

- Alternatives:
  - A node keeps every other slot handed over by a child, the other slots are passed on to the parent
  - A node keeps every $d + 1$ th slot, d is the depth of the node
- Advantage of 2^{nd} alternative:
  - If a node keeps a slot, all nodes on the path to the sink have already received an additional slot

Example: Nodes keep every second slot

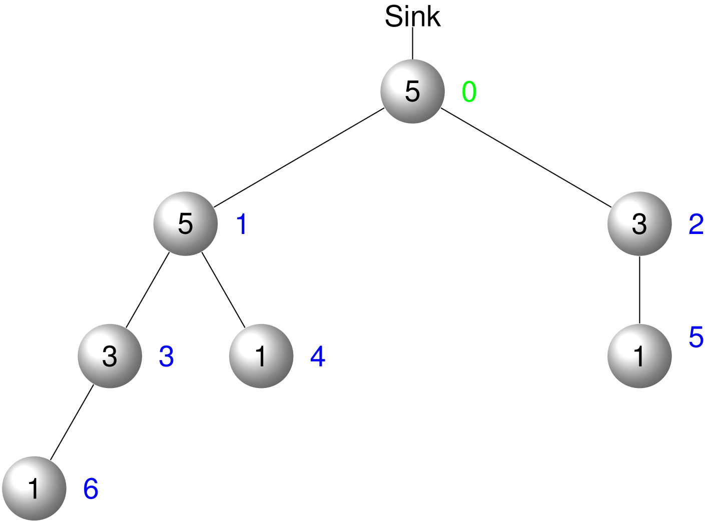


Example: Nodes keep every second slot



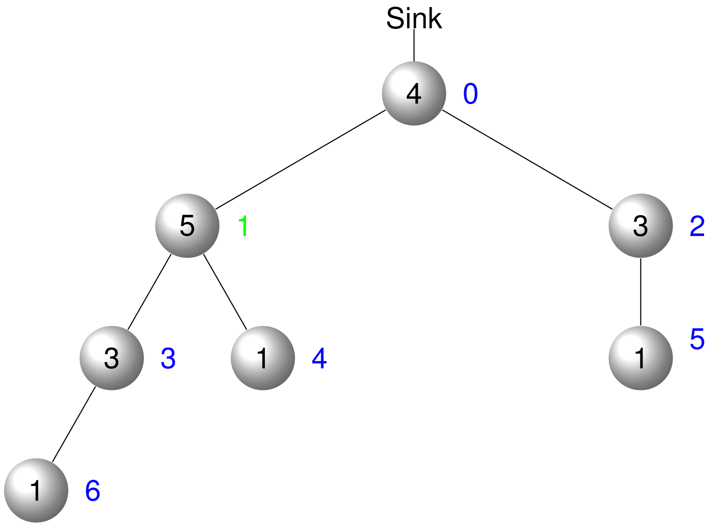
After Round 1

Example: Nodes keep every second slot



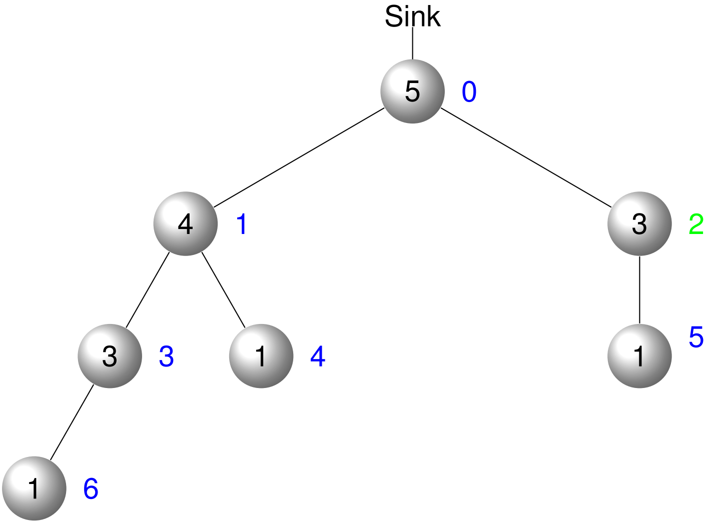
After Round 2

Example: Nodes keep every second slot



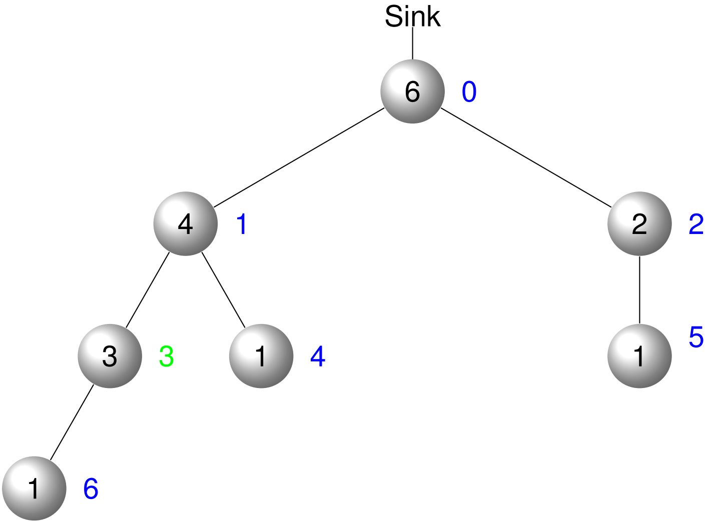
Round 3, Slot 0

Example: Nodes keep every second slot



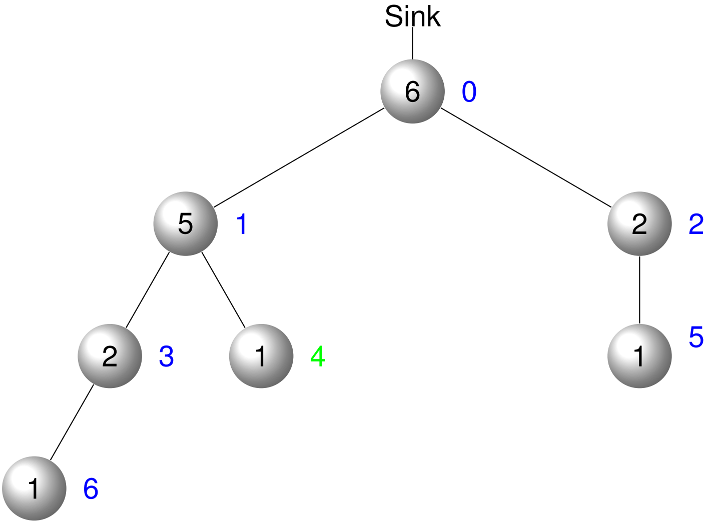
Round 3, Slot 1

Example: Nodes keep every second slot



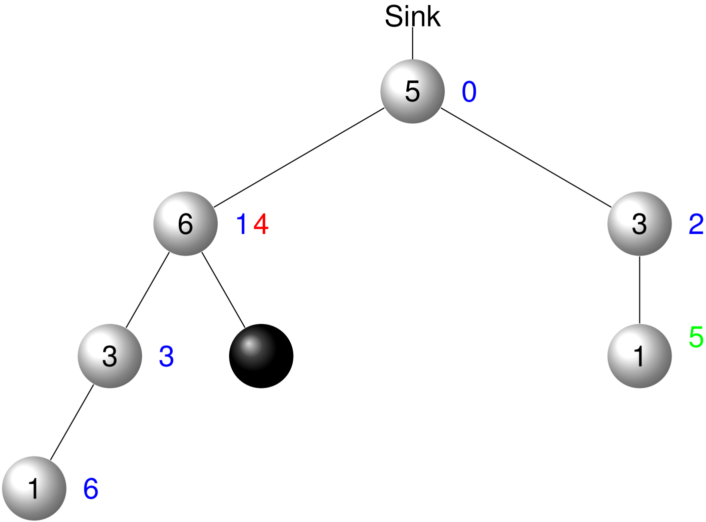
Round 3, Slot 2

Example: Nodes keep every second slot



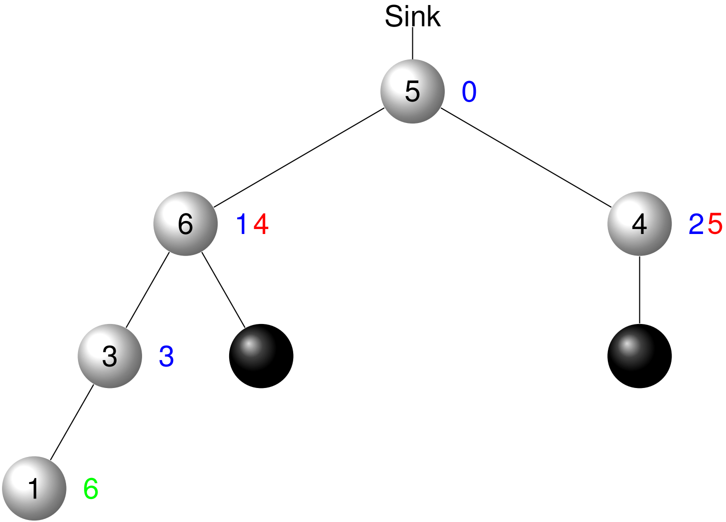
Round 3, Slot 3

Example: Nodes keep every second slot



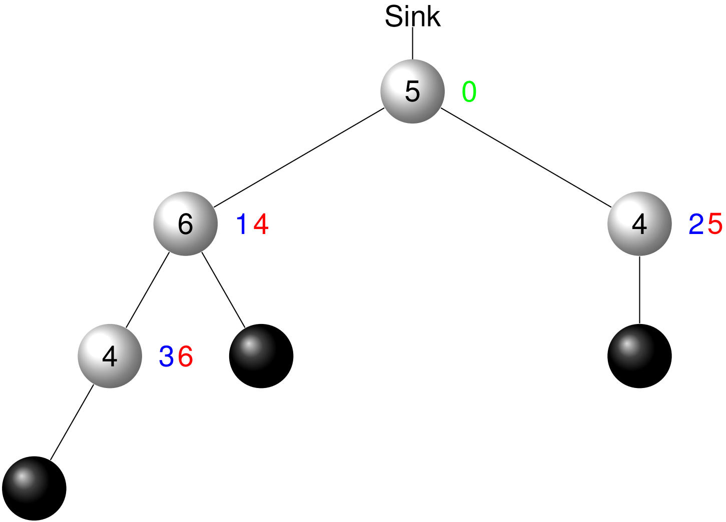
Round 3, Slot 4

Example: Nodes keep every second slot



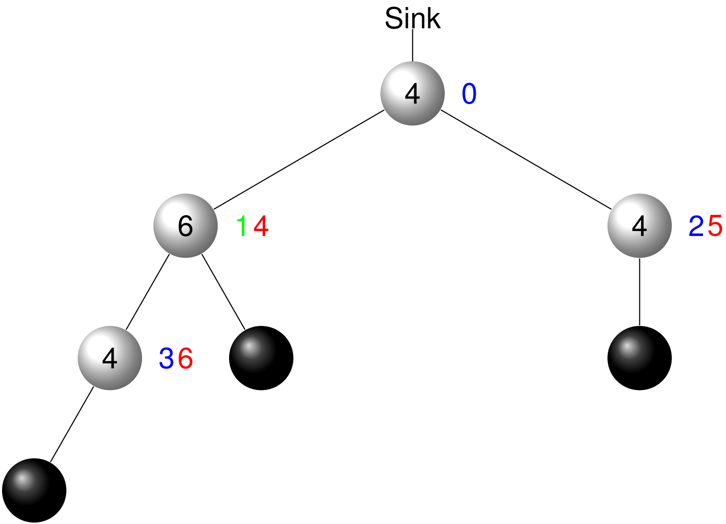
Round 3, Slot 5

Example: Nodes keep every second slot



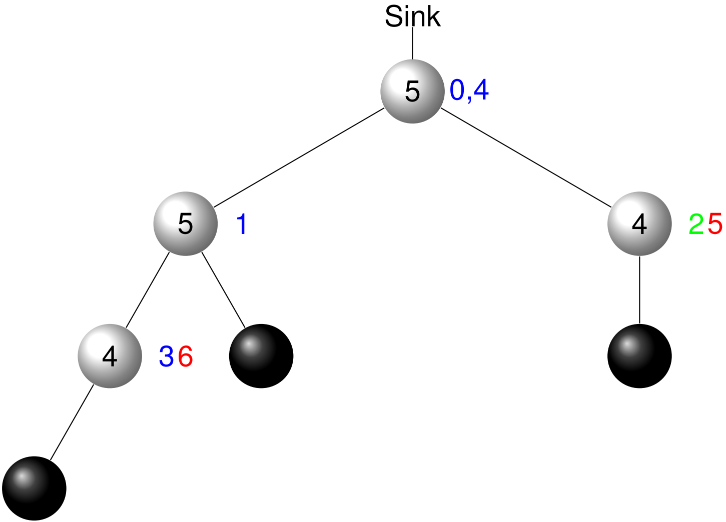
Round 3, Slot 6

Example: Nodes keep every second slot



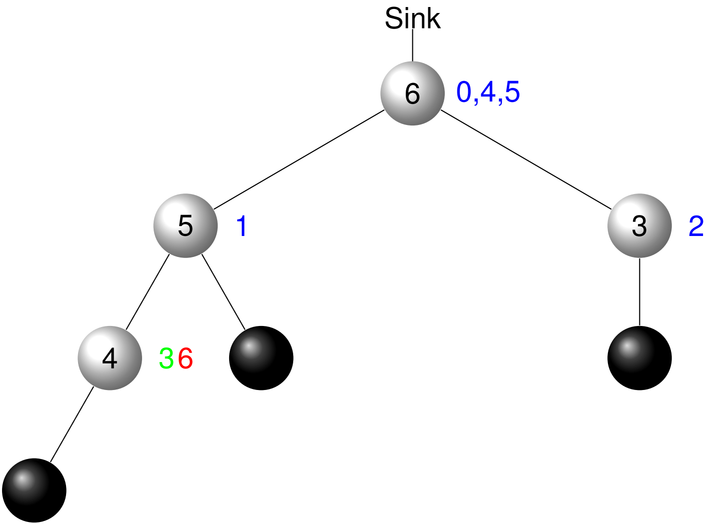
Round 4, Slot 0

Example: Nodes keep every second slot



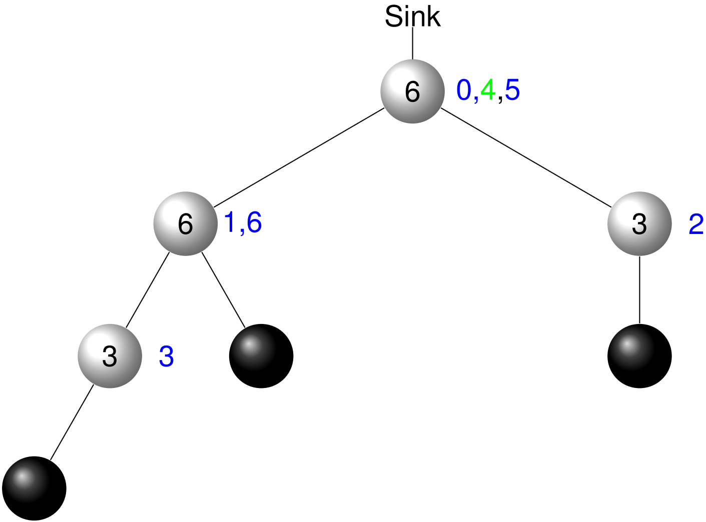
Round 4, Slot 1

Example: Nodes keep every second slot



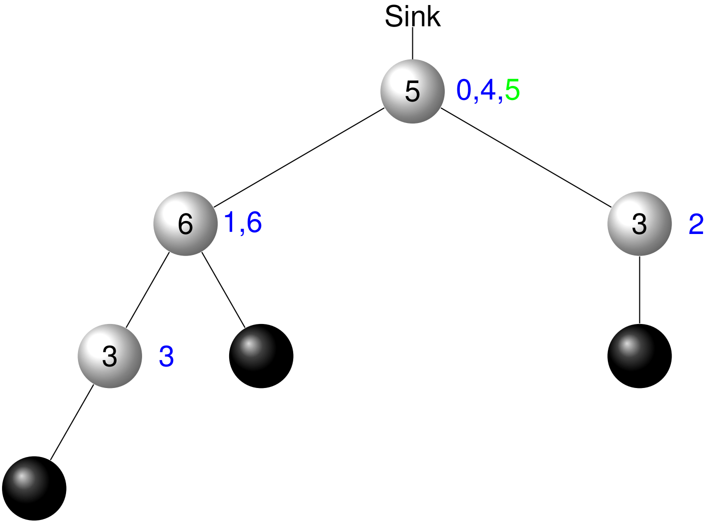
Round 4, Slot 2

Example: Nodes keep every second slot



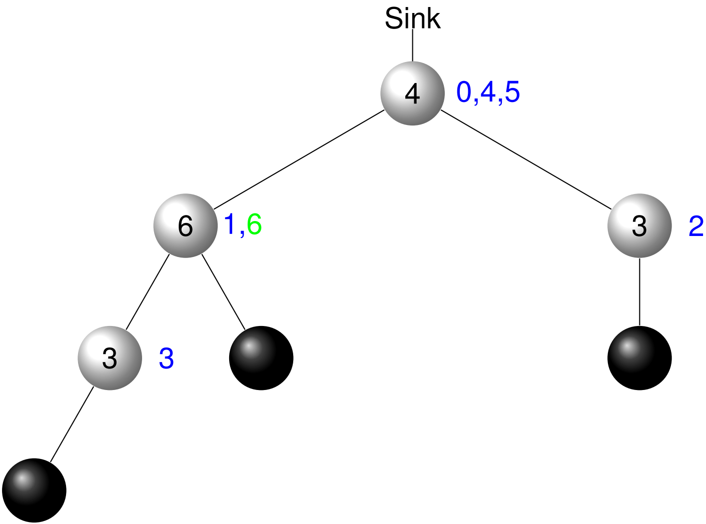
Round 4, Slot 3

Example: Nodes keep every second slot



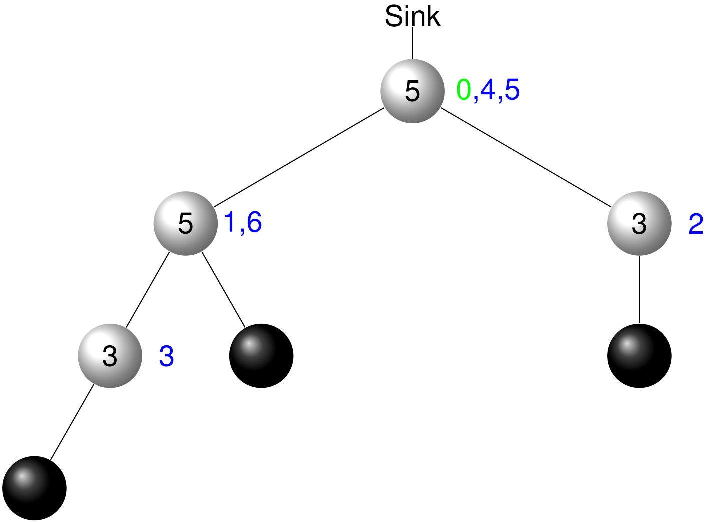
Round 4, Slot 4

Example: Nodes keep every second slot



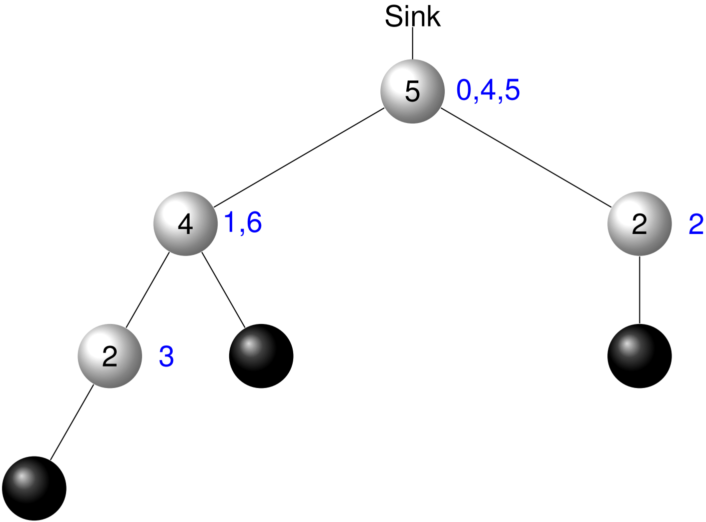
Round 4, Slot 5

Example: Nodes keep every second slot



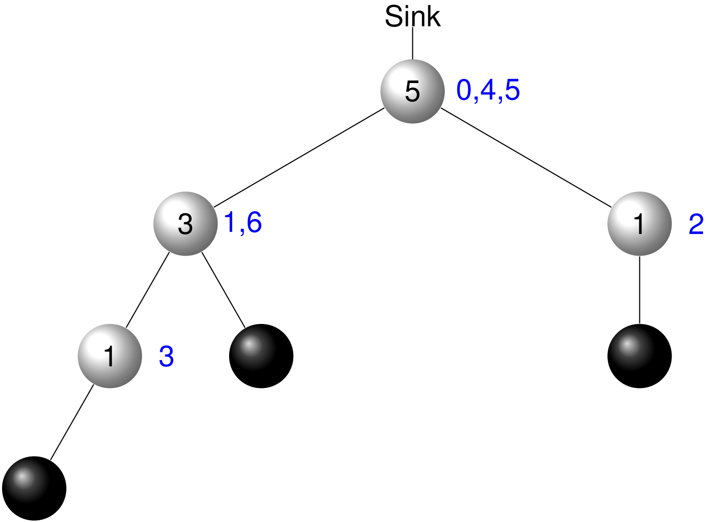
Round 4, Slot 6

Example: Nodes keep every second slot



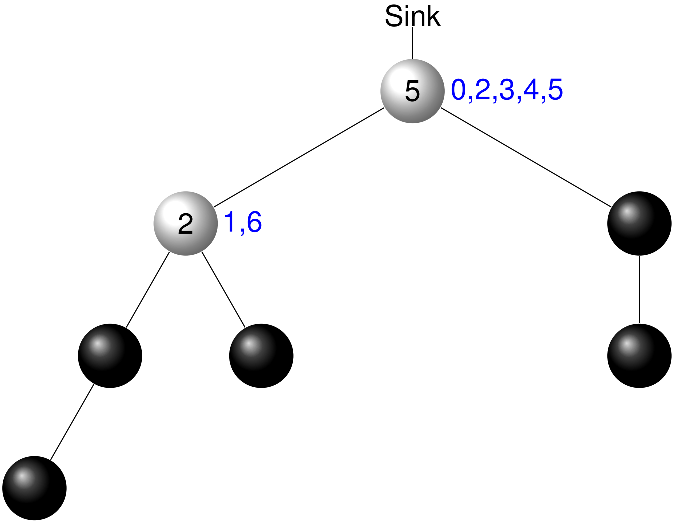
After Round 5

Example: Nodes keep every second slot



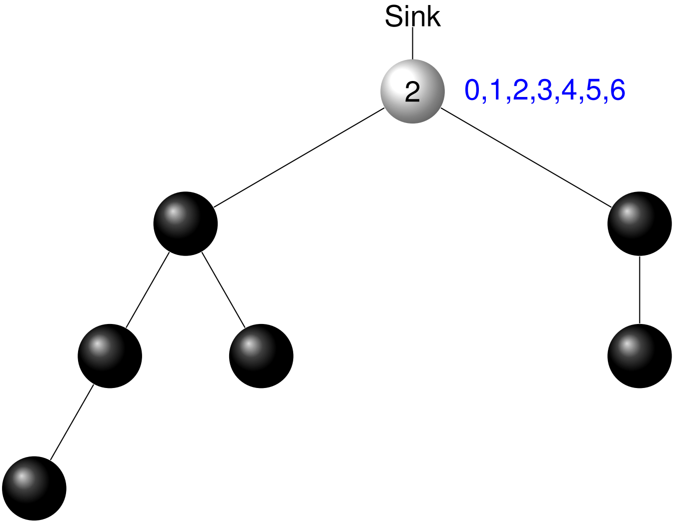
After Round 6

Example: Nodes keep every second slot



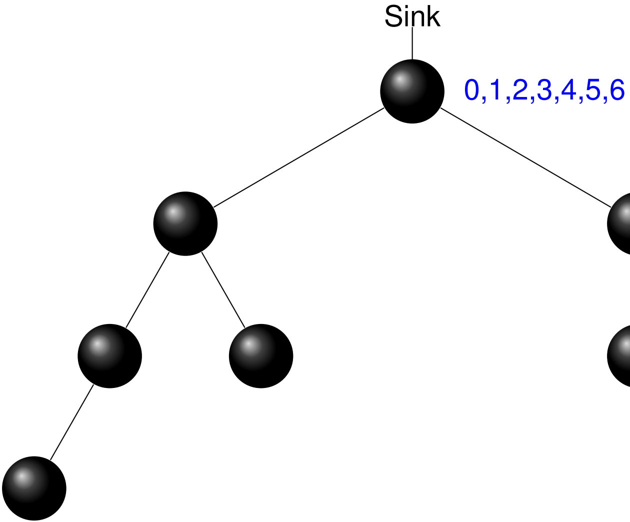
After Round 7

Example: Nodes keep every second slot



After Round 8

Example: Nodes keep every second slot



After Round 9

Sending Slot

Upon entering slot s do
if $s \in sendSlots$ **then**
    if $slotsToSkip > 0$ **then**
        $slotsToSkip = slotsToSkip - 1$
    else if $queue.size > 0$ **then**
        Switch on transceiver
        SENDPACKET
        Wait for acknowledgment
        Switch off transceiver
        Handle acknowledgment
    end if
end if

Sending Packets

- With each packet sent, a node
  - forwards received slots not kept by itself
  - informs parent about new slots used for sending in next round
  - informs parent if this is last packet the node will forward
- With the last packet a node forwards all available slots

Receiving Slot

Upon entering slot *s* do
if $s \in listenSlots$ **then**
  $slotsToSkip\_sender$ = number of slots sending node has to skip
  if $slotsToSkip\_sender > 0$ **then**
    $slotsToSkip\_sender$- -
  else
    Switch on transceiver
    Receive packet
    Switch off transceiver
  end if
end if

Receiving Packets

- Upon receiving a packet, a node
  - adds the data to its queue in EEPROM
  - accepts received new listening slots
  - handles received sending slots according to distribution scheme
  - sends acknowledgment including
    - number of future slots to skip in case buffer is full

Handling Acknowledgments

- Upon receiving an acknowledgment, a node
  - removes the first packet from its queue
  - clears the set of slots to forward
  - uses the new sending slots in future rounds
  - sets the number of slots to skip (if buffer of parent is full)

Reliability

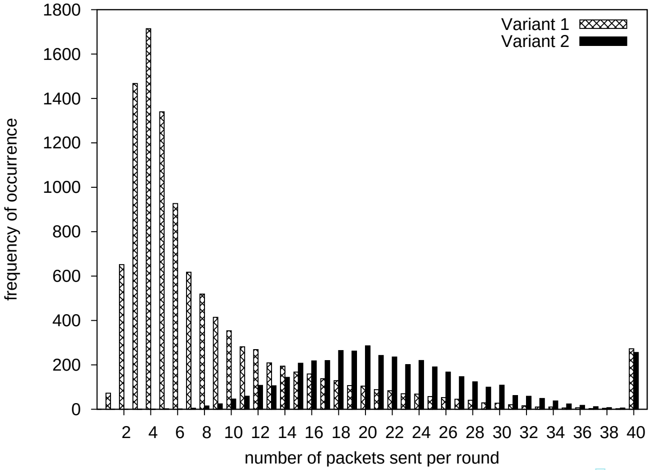
- Data is keep in EEPROM at any time
- Loss of packet is not a problem
- Loss of acknowledgment leads packet duplication (store hash values)
- Ensure that distribution of free slots is invariant under loss of acknowledgment
- Special treatment for loss of acknowledgment of last packet

Simulation

- 40 nodes
- 500 packets stored during flood tide
- Available time window: 8 hours
- Length of time slot 100 ms
- Length of round 4 s
- Buffer limit 1,000 packets

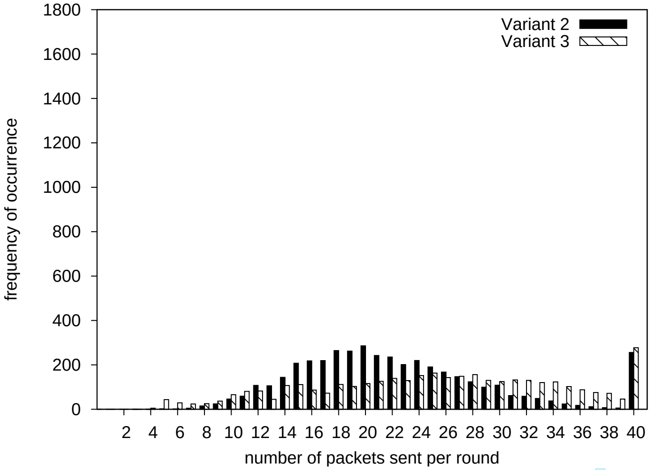
Simulation

Variant 1: All forwarded slots remain with the parent
Variant 2: Every second forwarded slot remains with the parent



Simulation

Variant 2: Every second forwarded slot remains with the parent
Variant 3: A node keeps every $d + 1$ th forwarded slot



Conclusion

- Novel protocol for reliable transport in data intensive applications that addresses all major sources of energy waste
- Simulations indicate good performance
- Future work:
  - Blocking function
  - Other distribution schemes

Scheduling Transmission of Bulk Data in Sensor Networks using a Dynamic TDMA Protocol

Volker Turau and Christoph Weyer

Institute of Telematics
Hamburg University of Technology

Data Intensive Sensor Networks
Mannheim, Germany, May 11, 2007

TUHH
Technische Universität Hamburg-Harburg