Low-delay compression for sensor networks

Alexandre Guitton

University of Oxford, Computing Laboratory

Joint work with Niki Trigoni and Sven Helmer

MSN 2007, 12th-13th of July 2007

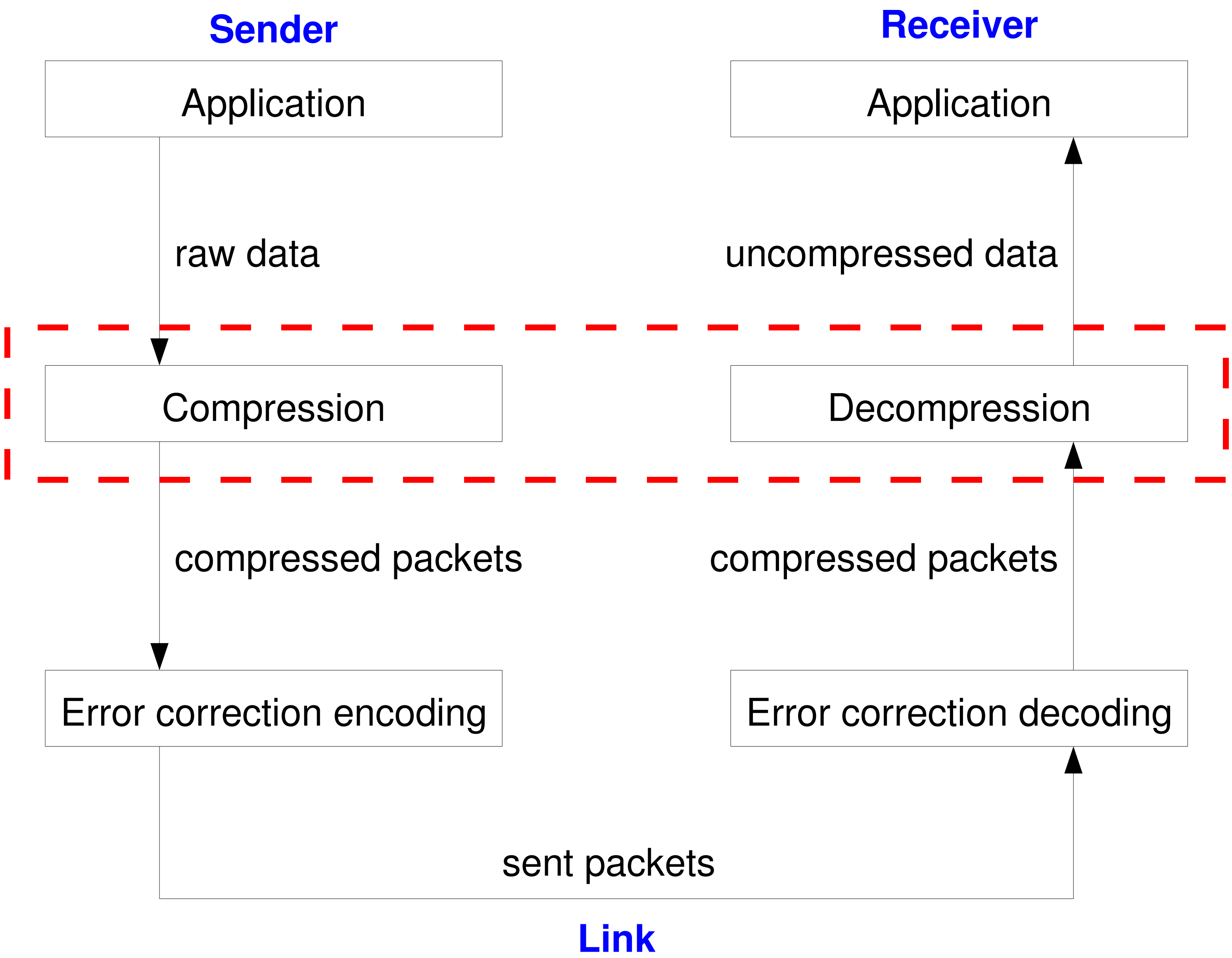
Outline

- Motivation
- Existing compression techniques
  - Standard compression (LZW, Adaptive Huffman)
  - Compression with packet retransmissions (RT)
- Proposed fault-tolerant compression (FT)
- Evaluation and conclusions

Motivation

- Sensor nodes are battery powered
  - To save energy: compressing data before transmitting it
- Challenge: lossy communication channels
- Performance metrics
  - Energy-efficiency: $\frac{\text{Bytes of uncompressed data at the receiver}}{\text{Bytes transmitted by the sender}}$
  - Delay between first transmission and decoding

Our focus


Sender
Receiver
Application
Application
raw data
uncompressed data
Compression
Decompression
compressed packets
compressed packets
Error correction encoding
Error correction decoding
sent packets
Link

Existing approaches

Standard compression

Zebra sensing

"Zebra seen at 8:00 in (20,30)"

LZW

0010
1011
1100
1010

Hamming encoding

0010(01) 1011(10)
1100(00) 1010(11)

0010(01) **0**01**0**(**01**)
1100(0**1**) 101**1**(11)

Hamming decoding

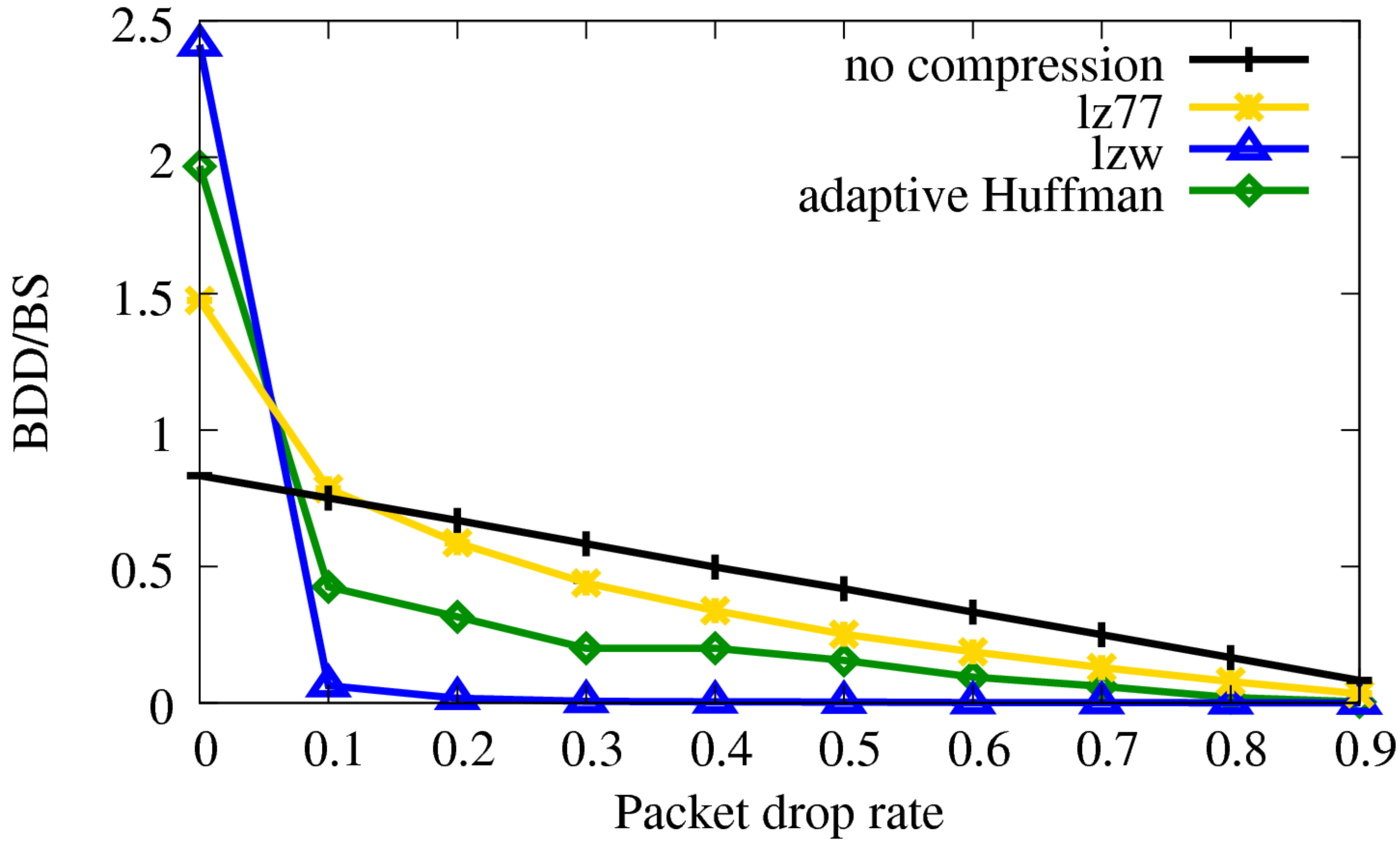
0010
X
1100
1010

LZW

"Zebra "

Zebra sensing

Existing approaches
Standard compression



Compressing data with dynamic dictionaries **is less energy-efficient** than not compressing it when the packet drop rate exceeds 10%

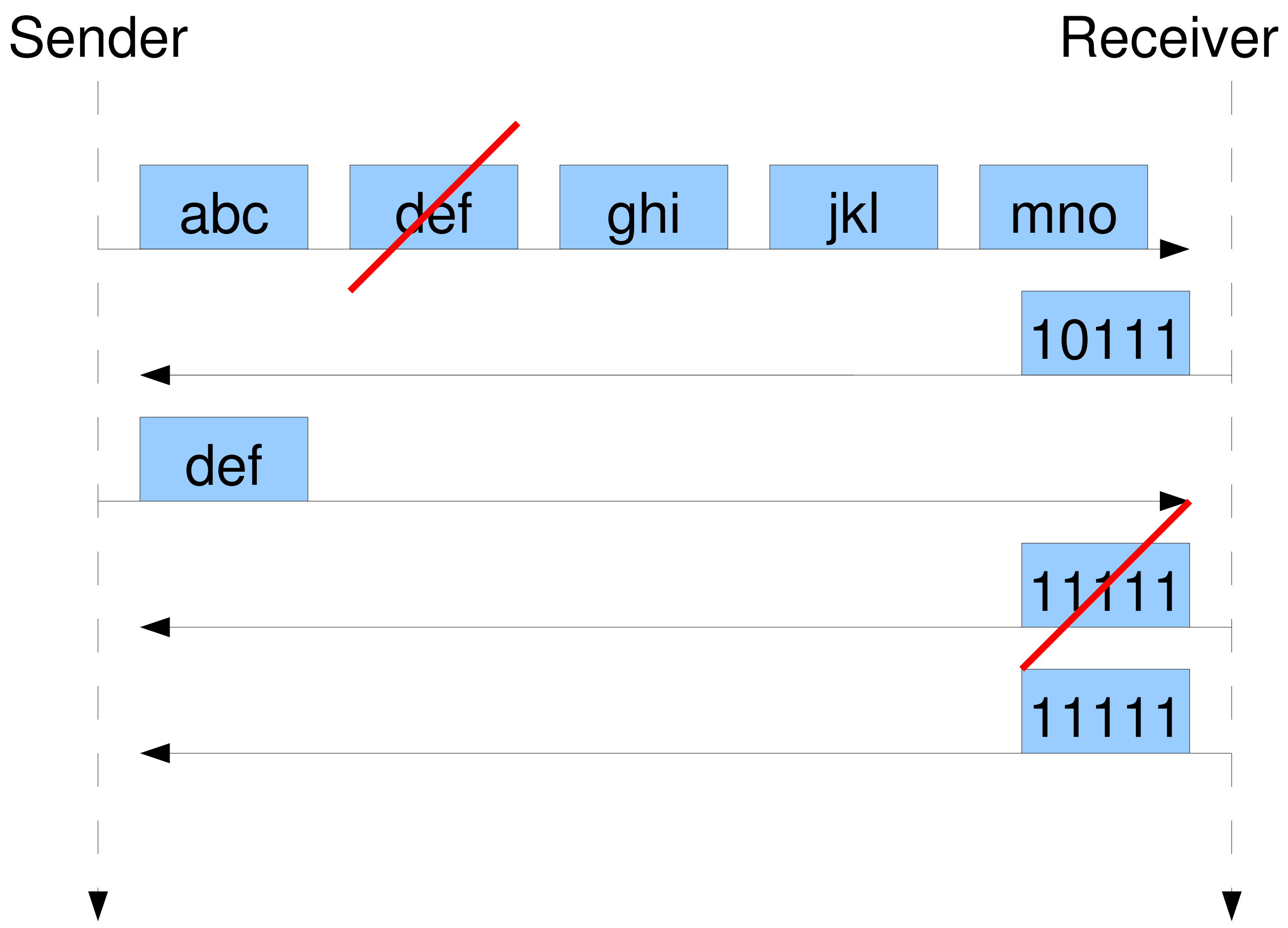
Existing approaches

Compression with retransmissions

- Retransmission (RT) mechanism [Sadler and Martonosi, 2006] to cope with packet losses
  - Packets are grouped in blocks
  - Receiver sends block ACKs
  - Sender retransmits dropped packets
  - Compression is restarted at each block
- RT is applied to LZW (RT-LZW)

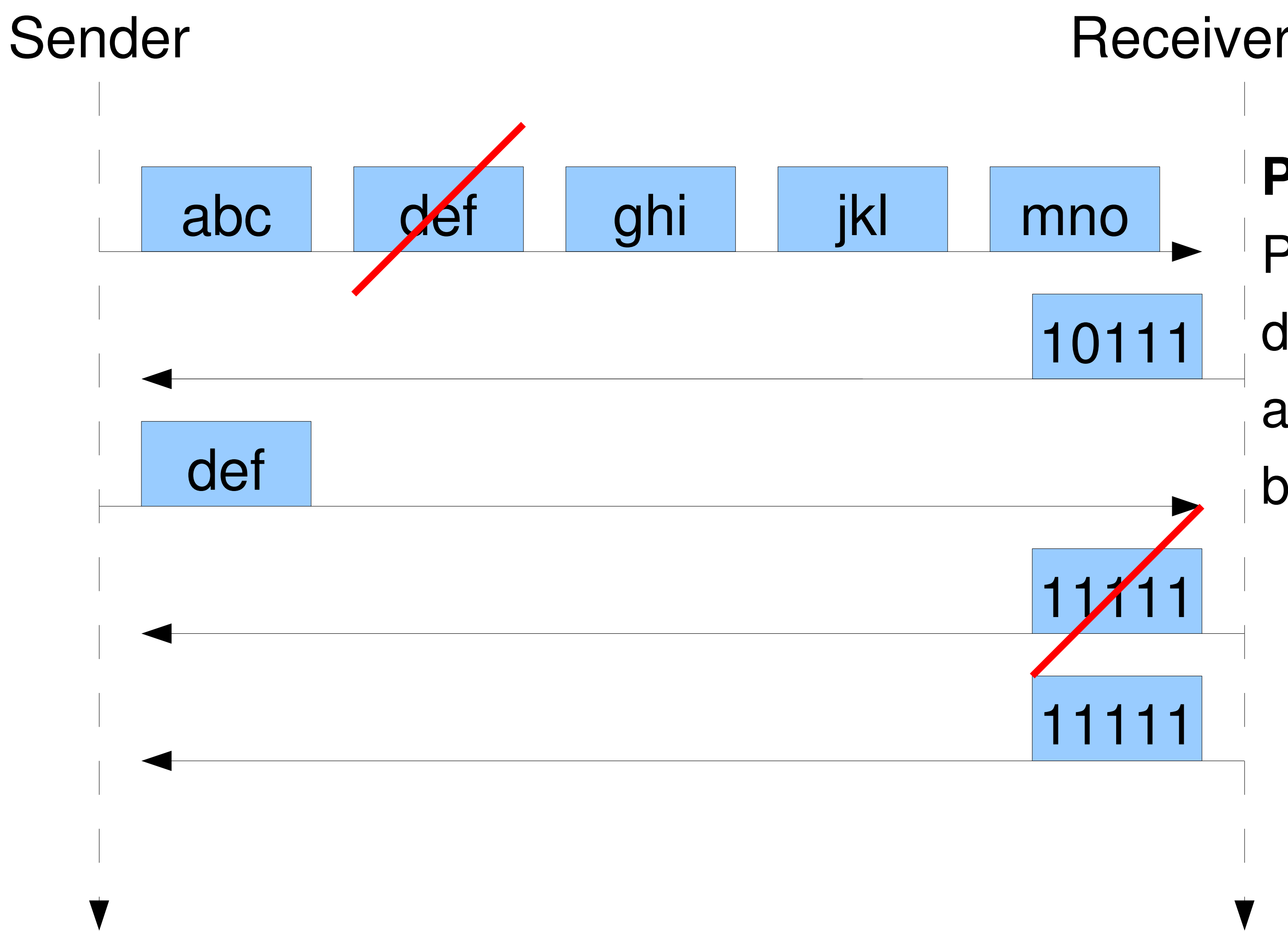
Existing approaches

Compression with retransmissions



Existing approaches

Compression with retransmissions



Problem 1

Packets cannot be decoded as they arrive, they have to be decoded in order

Existing approaches
Compression with retransmissions

Problem 1

Packets cannot be decoded as they arrive, they have to be decoded in order

Problem 2

In the event of a disconnection, the effort of the sender is wasted

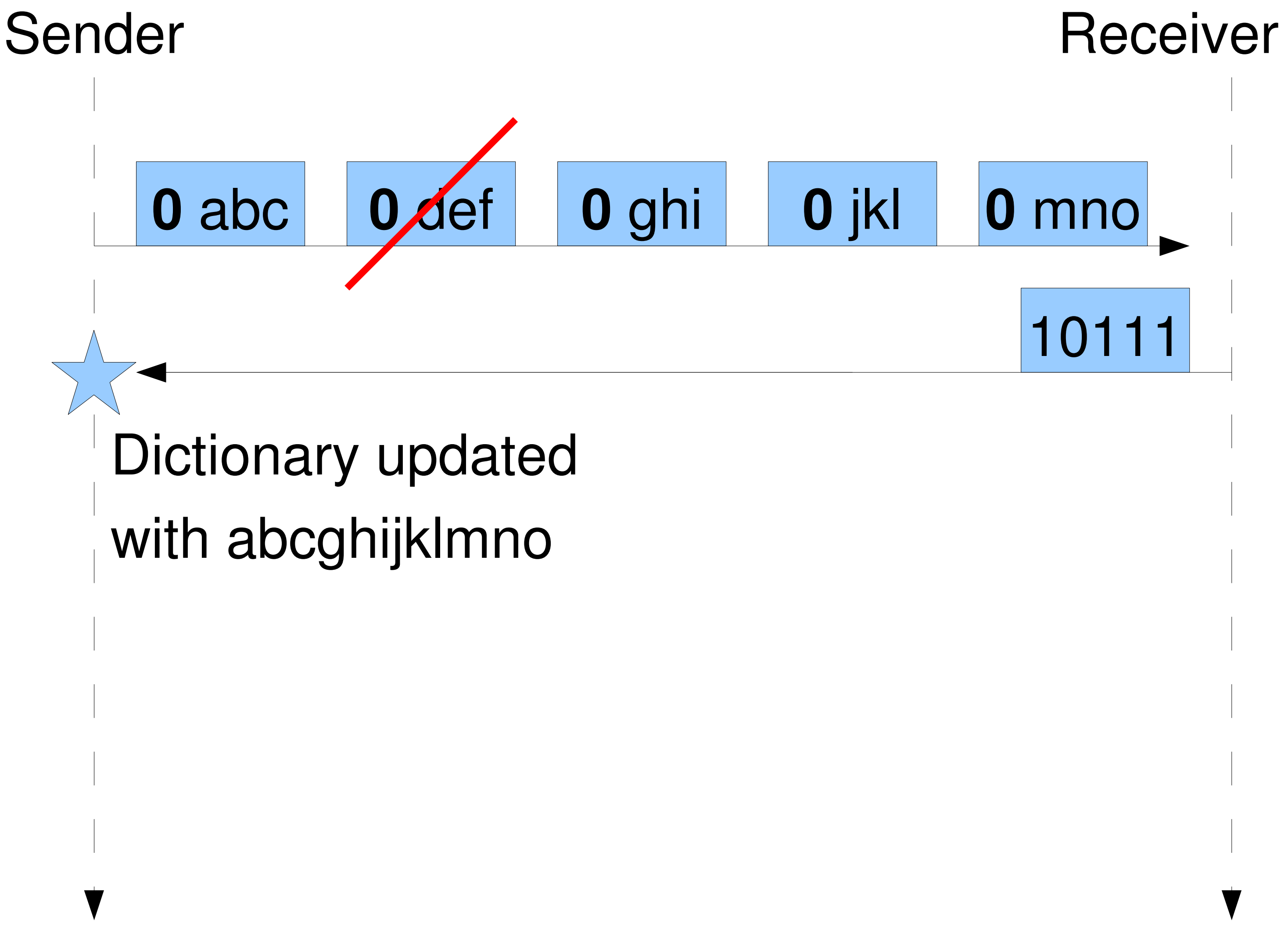
Existing approaches
Compression with retransmissions

- To address these two problems
  - Small blocks are used (the delay is not too large, and energy is not wasted in case of a disconnection)
  - The dictionary is restarted at the beginning of each block (blocks are independent of each other)
- But
  - Small blocks reduce the potential for compression

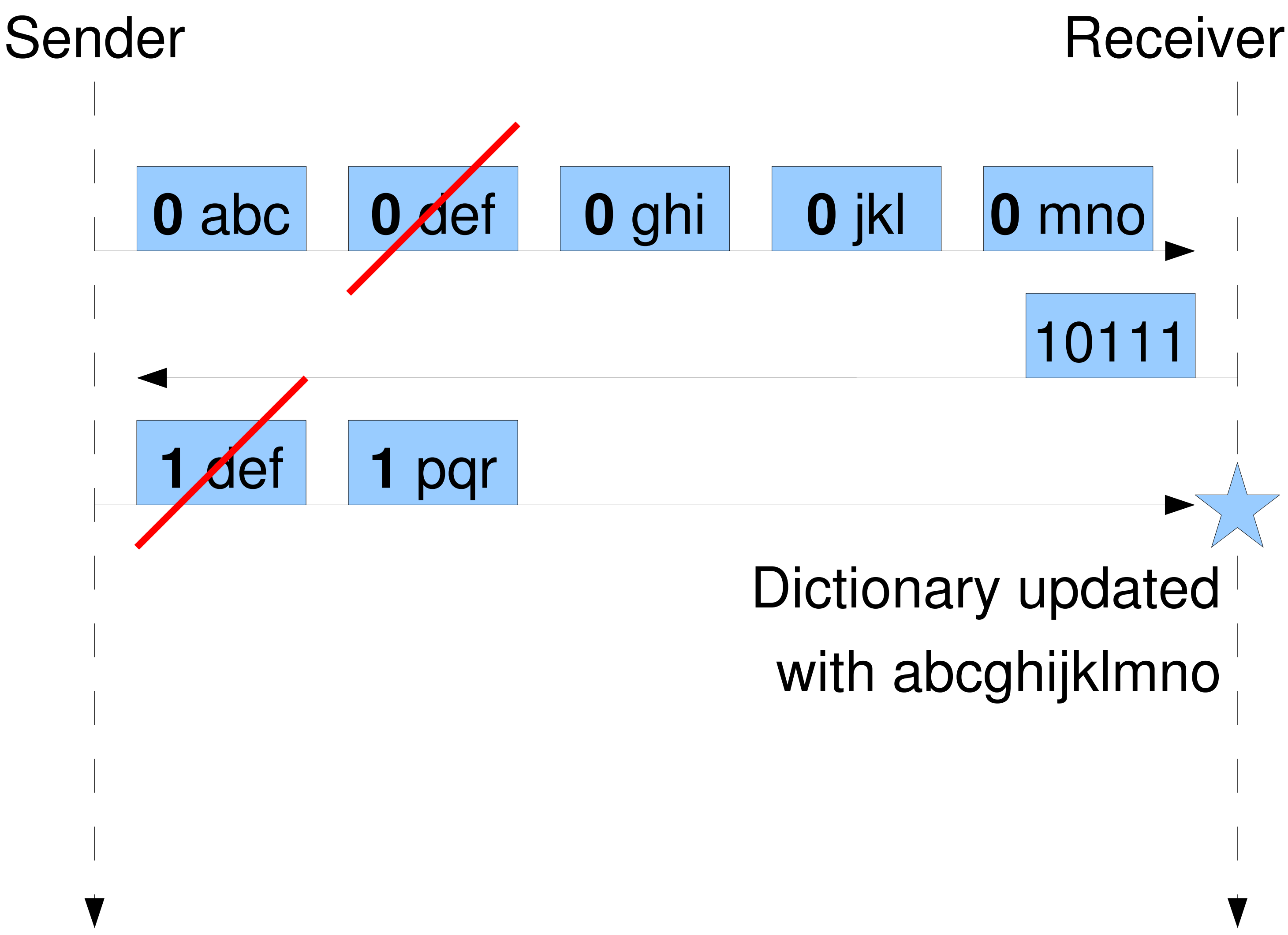
Proposed fault-tolerant mechanism

- Fault-Tolerant (FT) mechanism
  - Packets are grouped in blocks (as in RT)
  - Block ACKs (as in RT)
  - Dictionary is updated after each block (NOT after each symbol)
  - Each packet of a block can be decoded independently of the other packets of the block

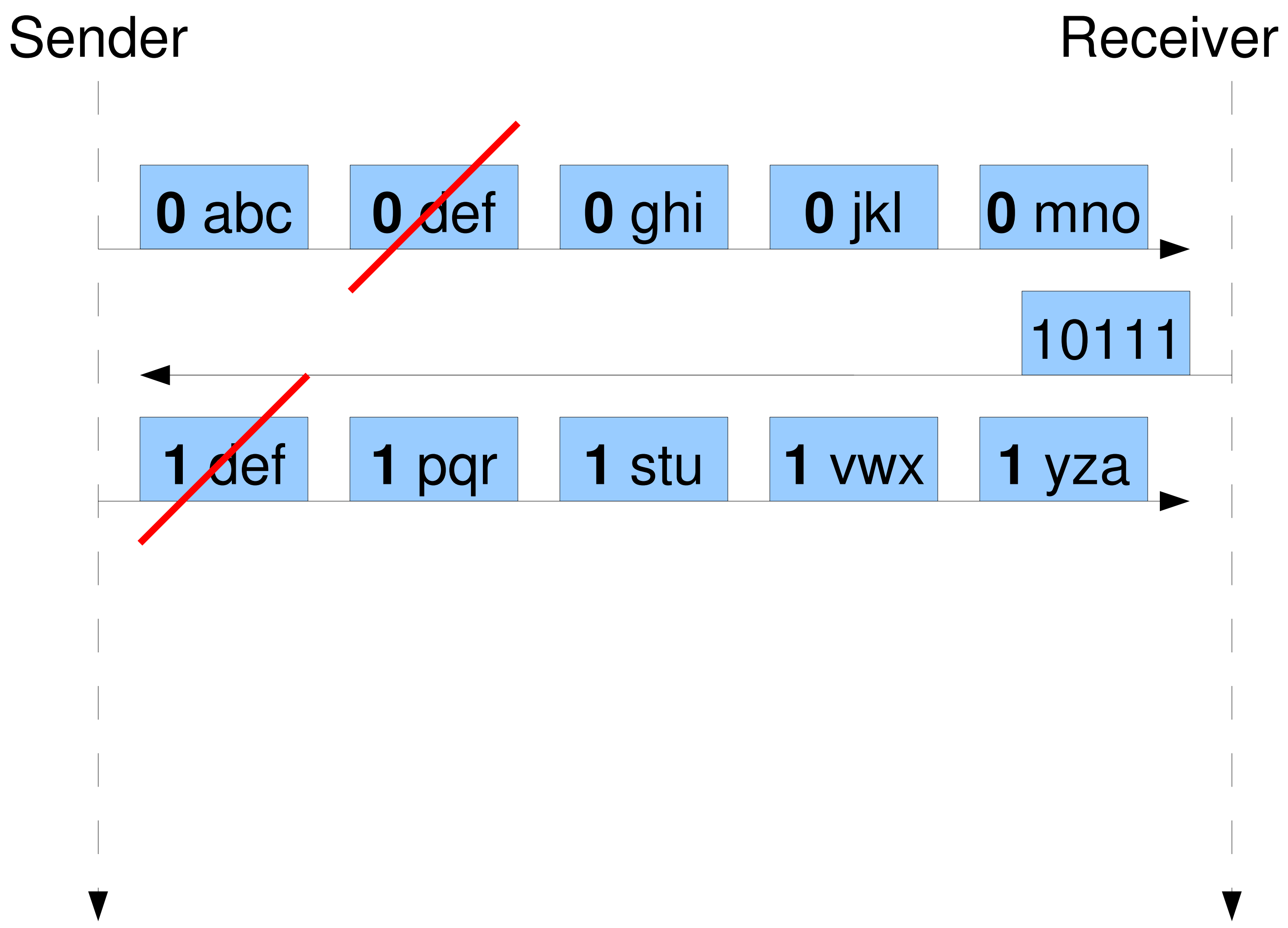
Proposed fault-tolerant mechanism



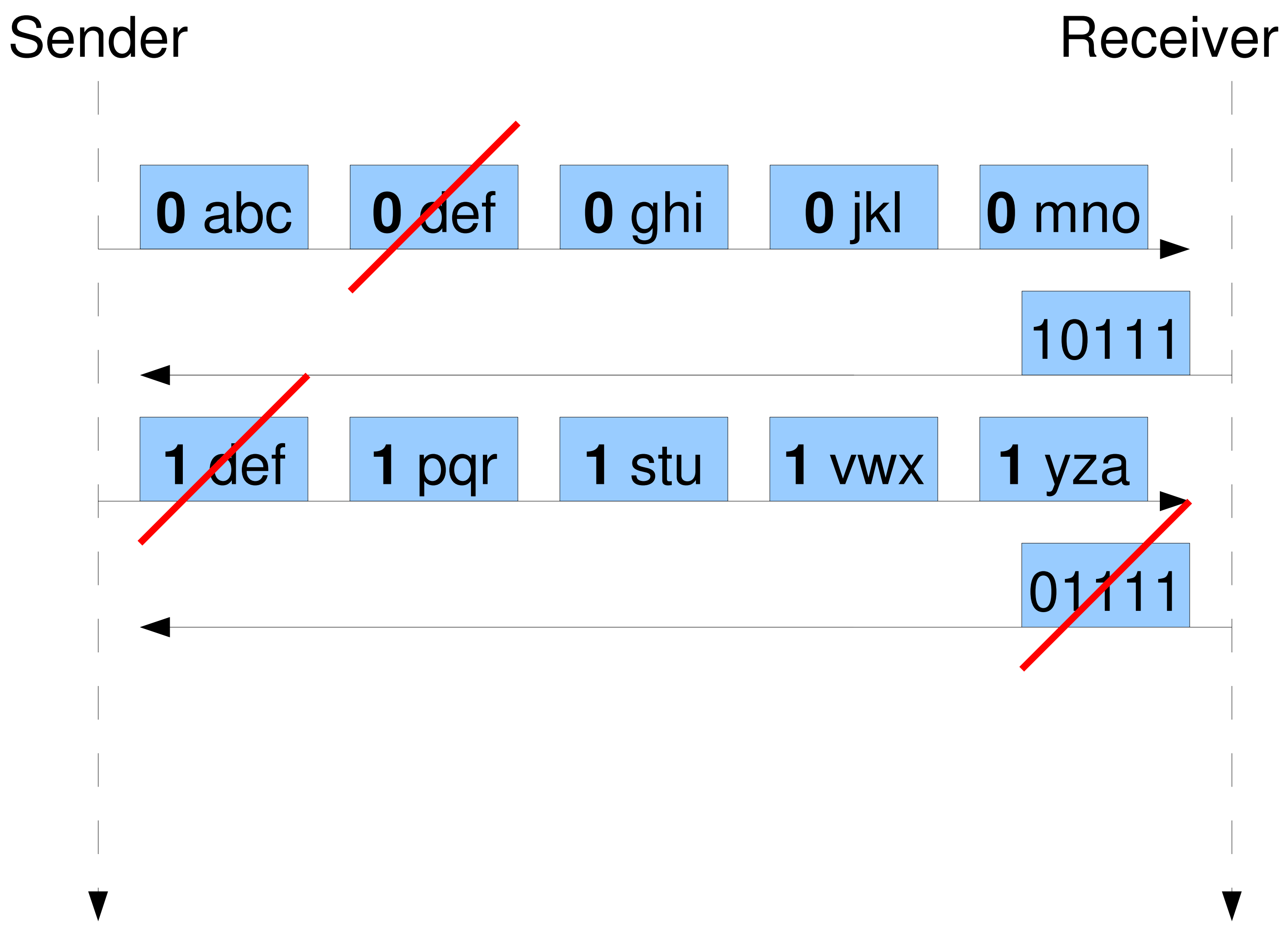
Proposed fault-tolerant mechanism



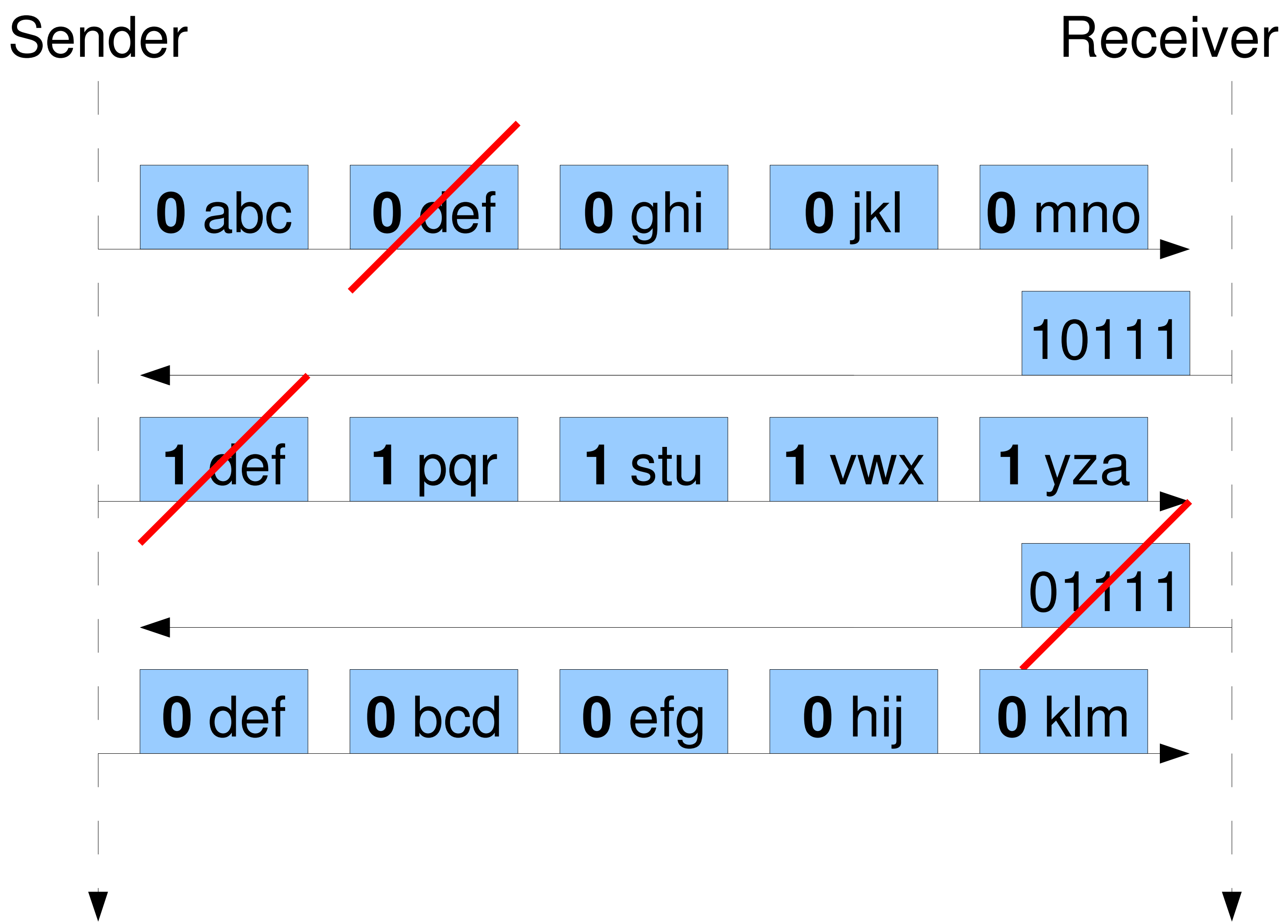
Proposed fault-tolerant mechanism



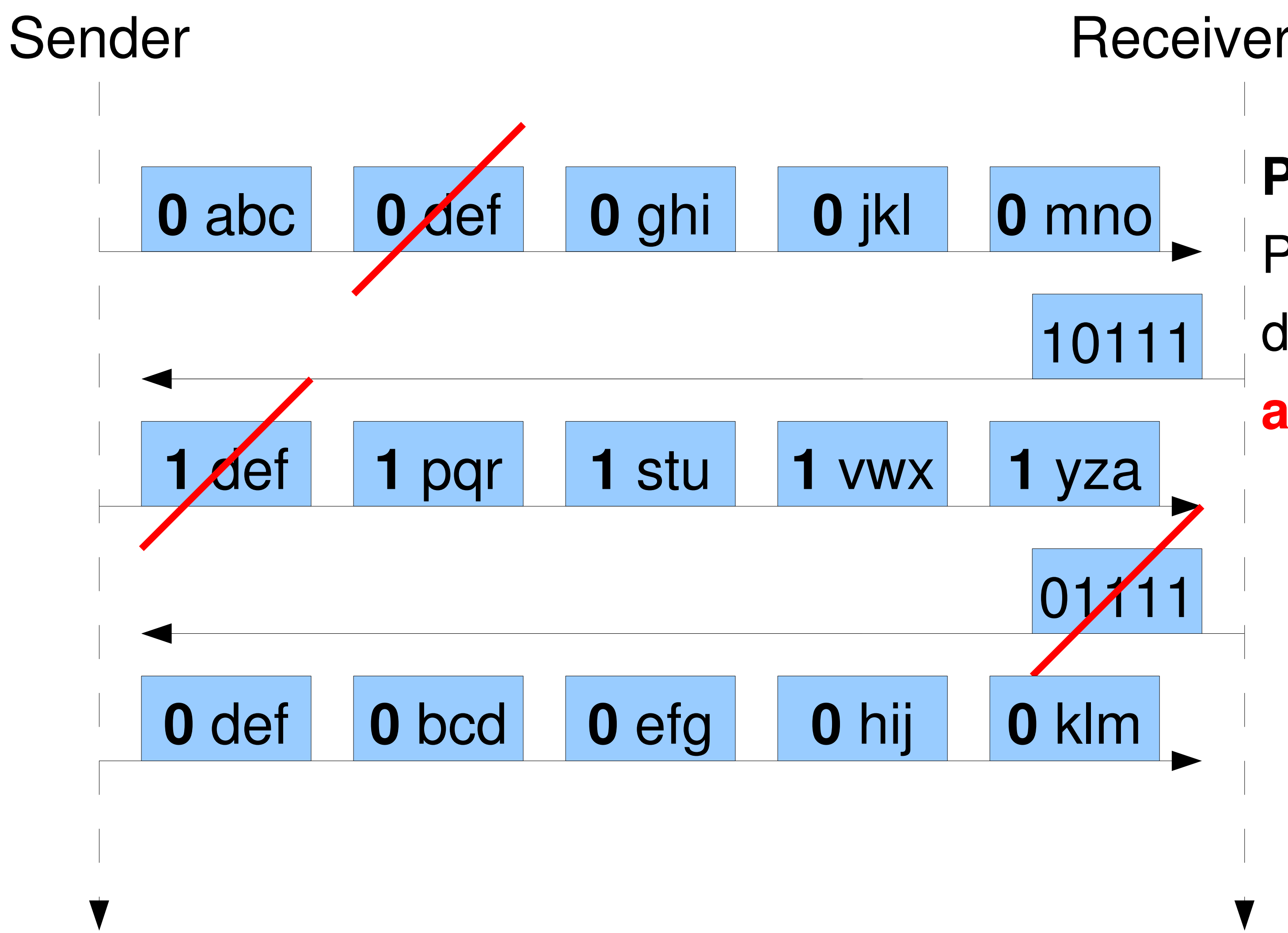
Proposed fault-tolerant mechanism



Proposed fault-tolerant mechanism



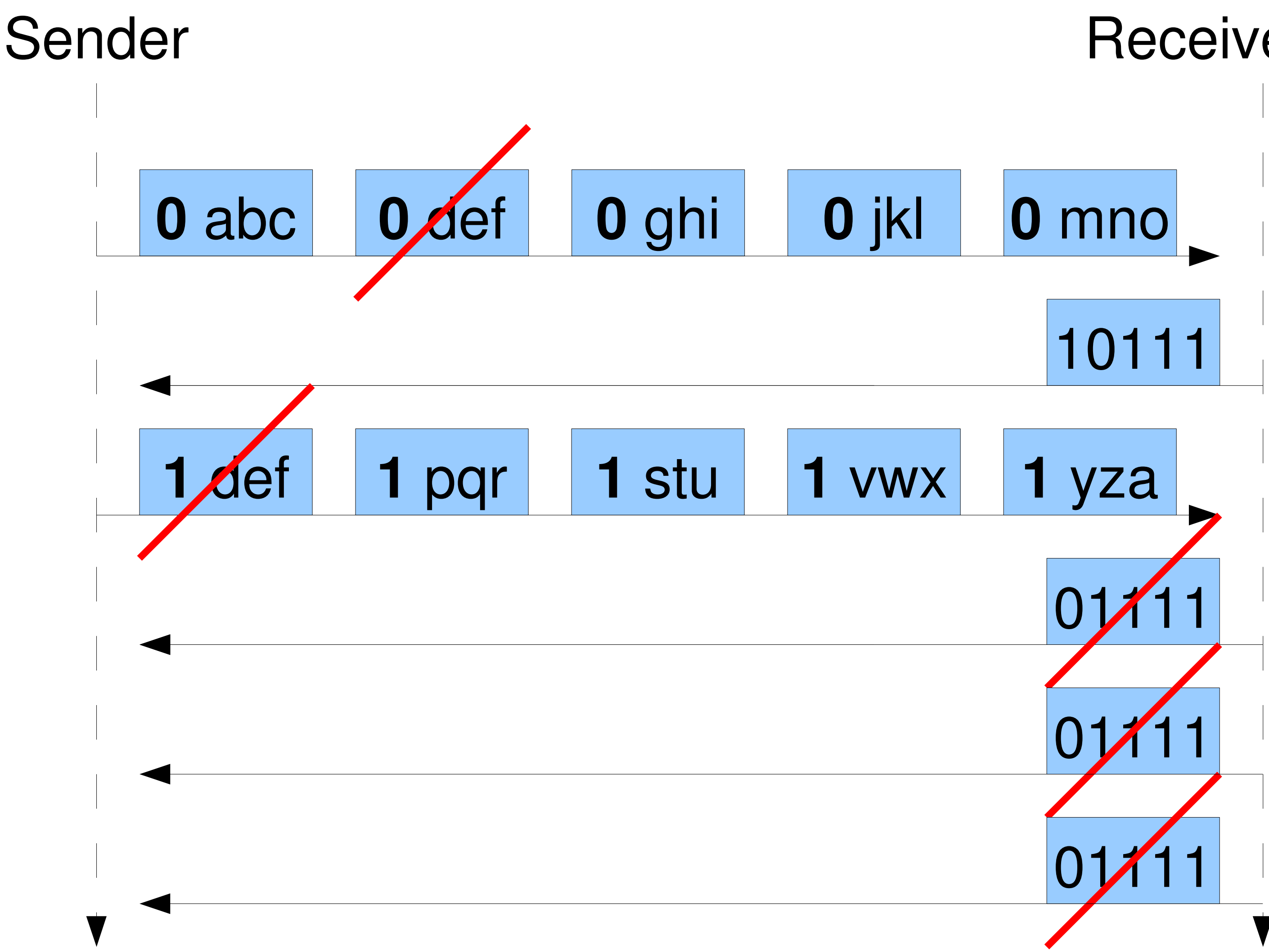
Proposed fault-tolerant mechanism



Problem 1?

Packets can be decoded **as soon as they arrive**

Proposed fault-tolerant mechanism



Problem 1?

Packets can be decoded **as soon as they arrive**

Problem 2?

In the event of a disconnection, all the packets that have been received are useful

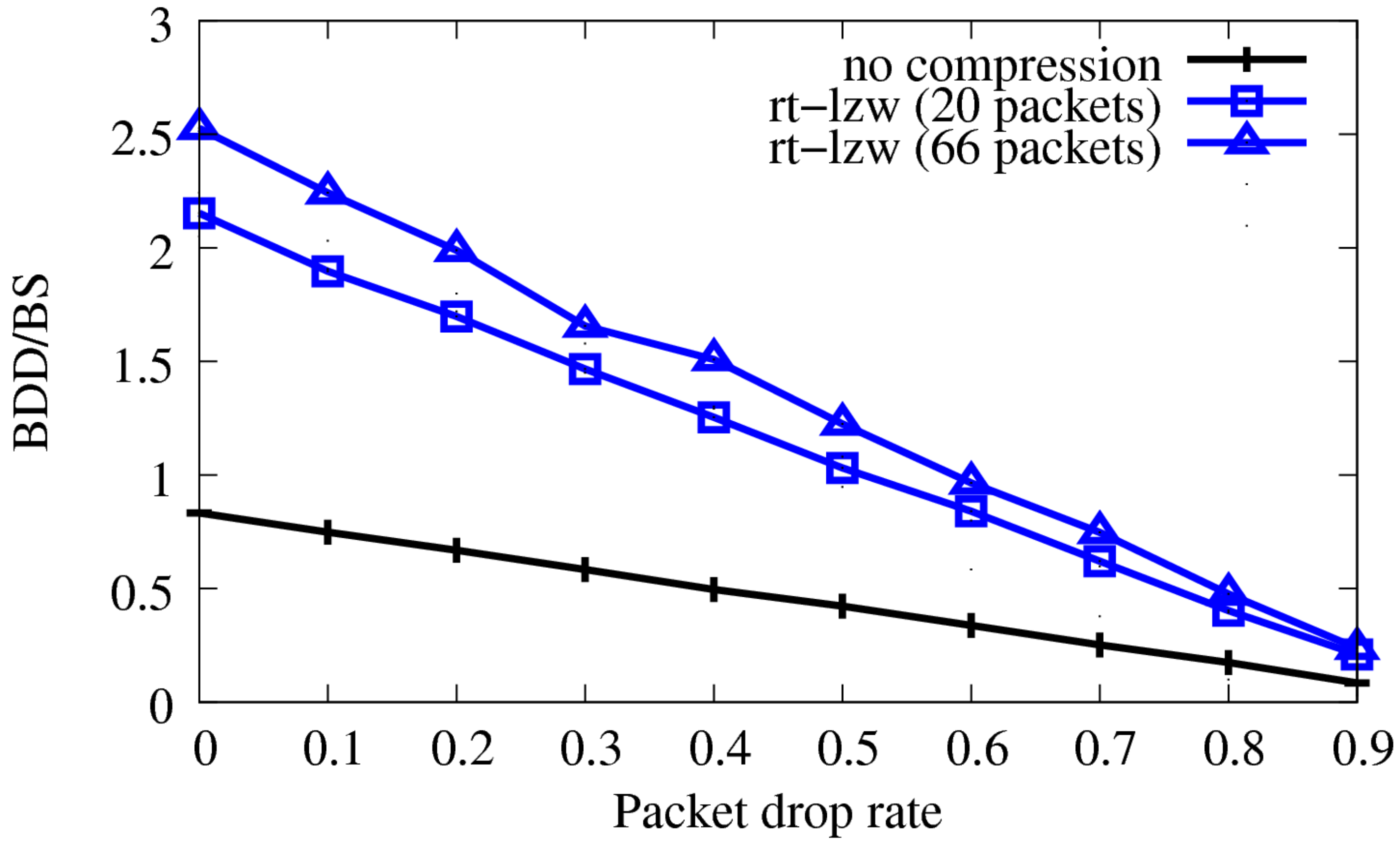
Proposed fault-tolerant mechanism

- Advantages of FT over the RT mechanism
  - Packet can be decoded when they arrive
  - Dictionaries are not reinitialized at each block
  - Availability of the backward link is not mandatory
- Disadvantage
  - Compression is conservative because the dictionary is only updated at the end of each block

Experimental setup

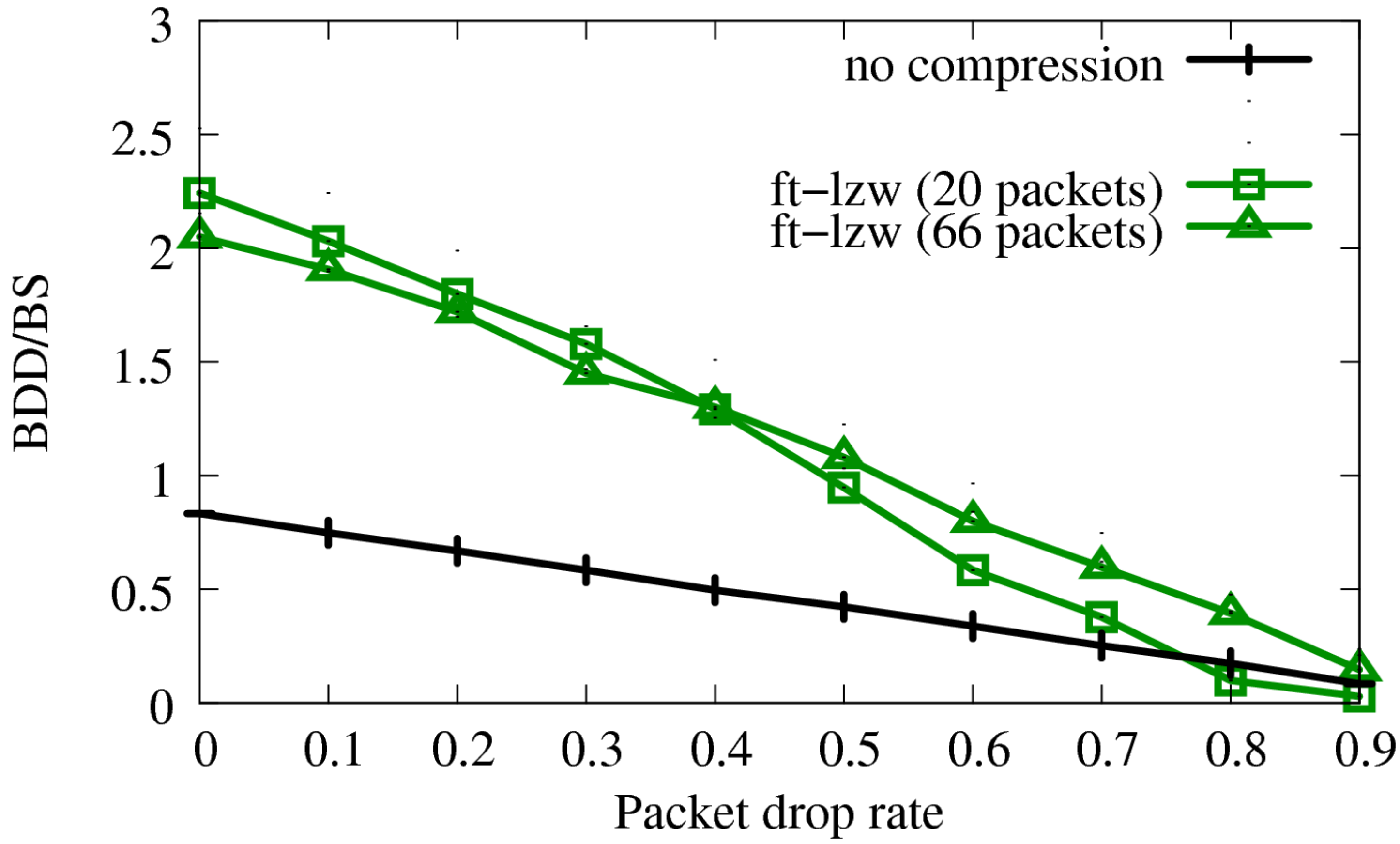
- We applied the RT and FT mechanism to LZW (RT-LZW and FT-LZW)
  - Real road traffic dataset (Scoot)
  - Block sizes of 20 and 66 packets
  - Varied packet loss rate on a link from 0% to 90%
- We measure
  - Energy-efficiency
  - Delay

Evaluation - Energy-efficiency



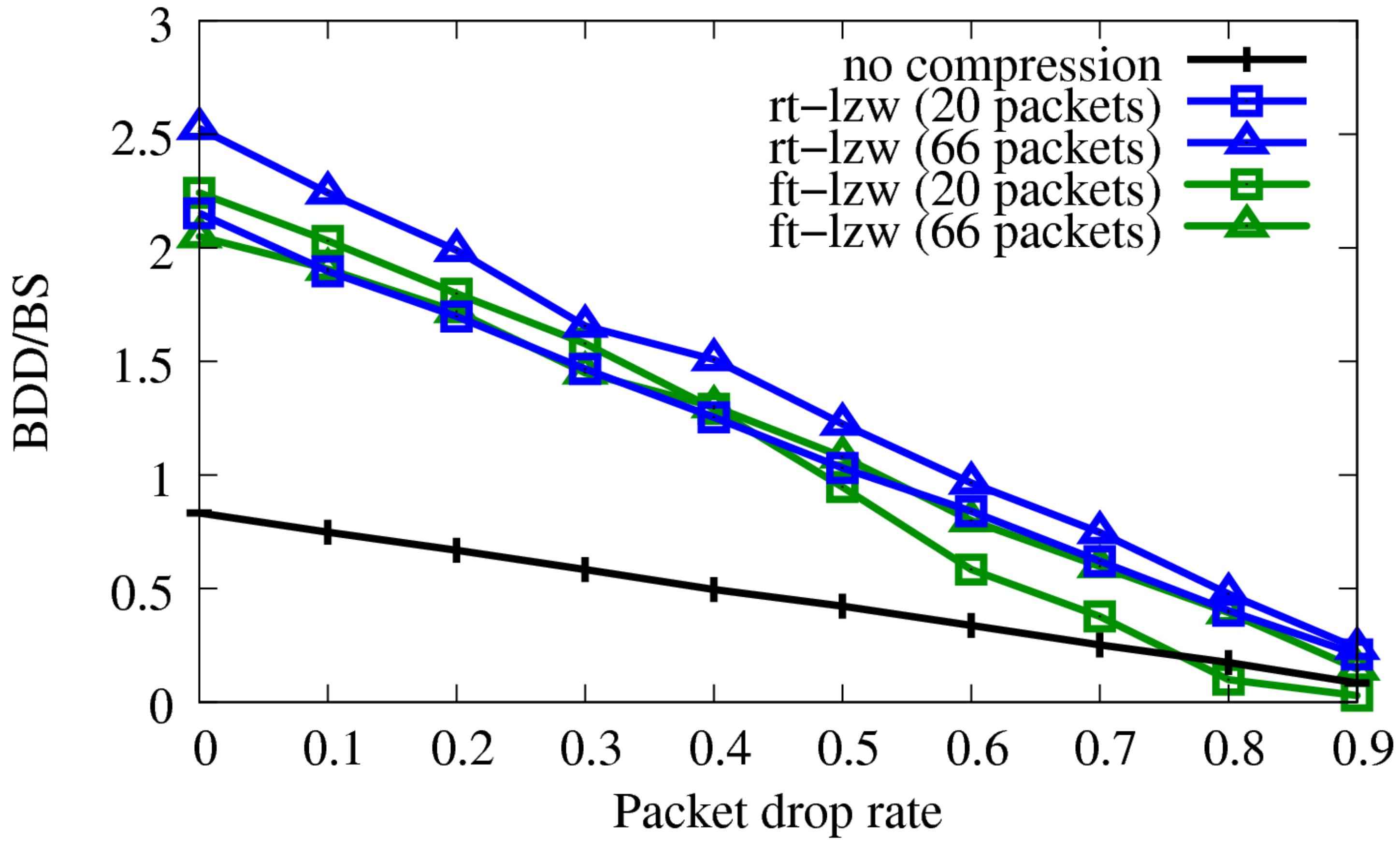
The energy-efficiency of RT-LZW increases with the block size

Evaluation - Energy-efficiency



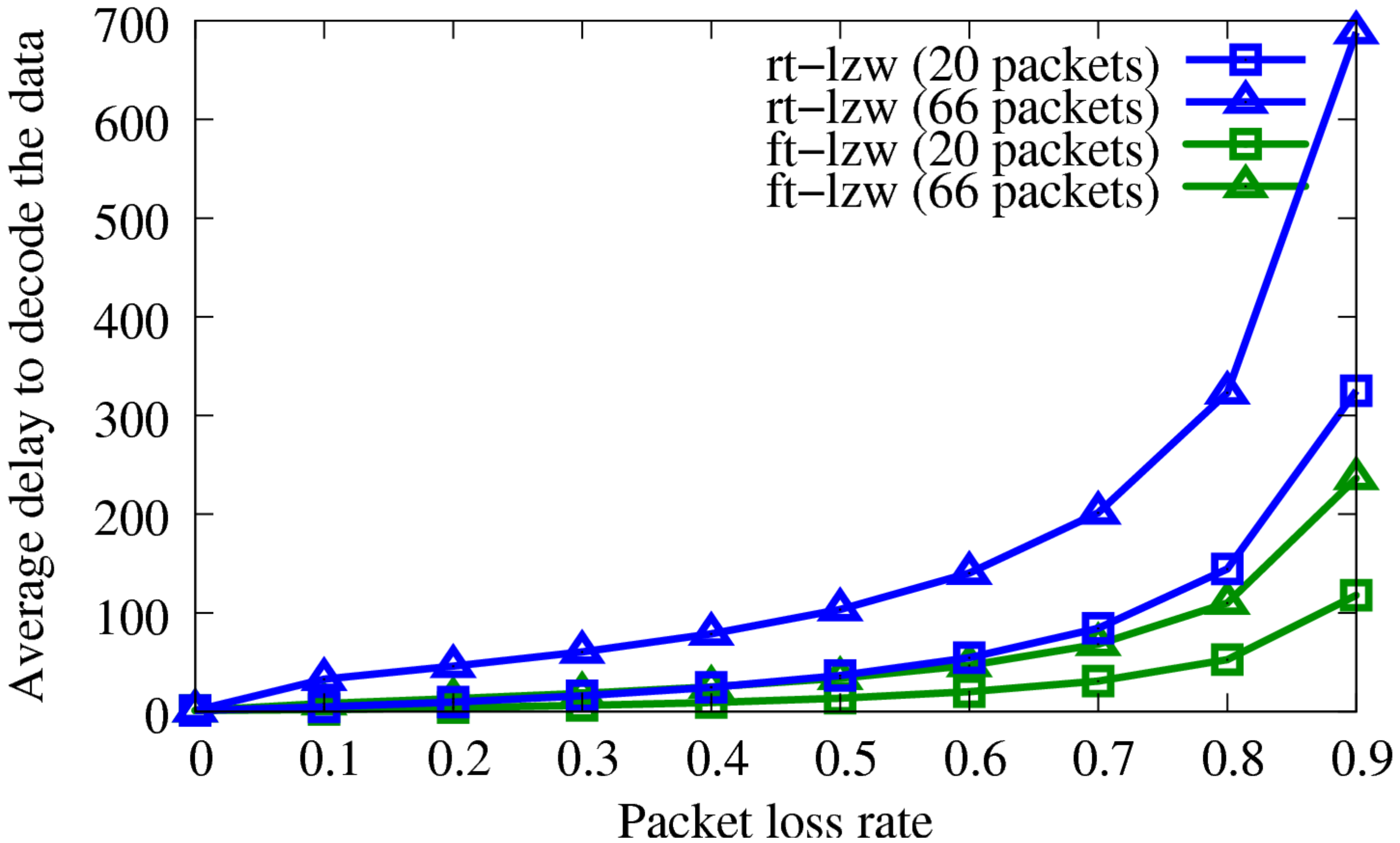
(1) The energy-efficiency of FT-LZW decreases as the block size increases and (2) small block sizes cannot be used in highly lossy environments

Evaluation - Energy-efficiency



RT and FT mechanisms (1) degrade linearly as the packet drop rate increases, and (2) **are comparable**

Evaluation - Delay

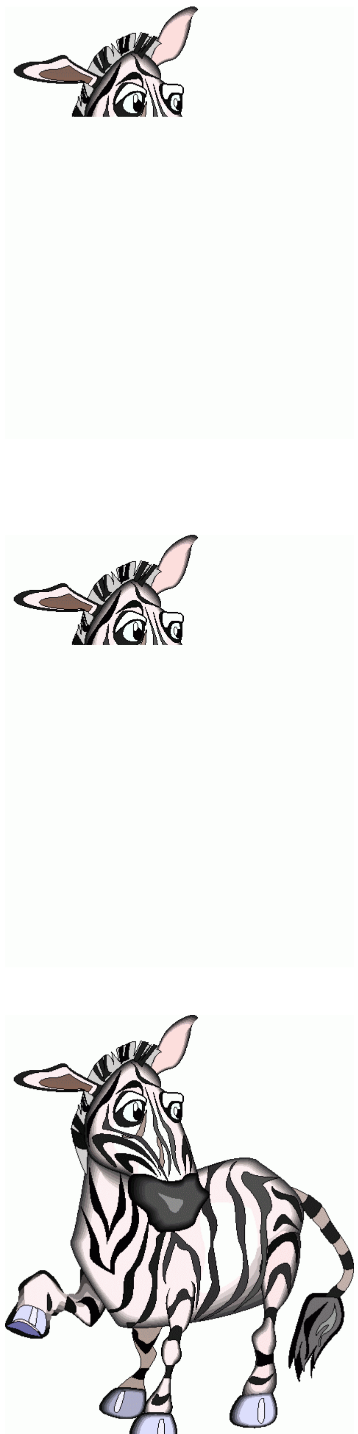


FT is 2-3 times faster than RT for all block sizes

Conclusions

- Standard compression algorithms fail in lossy environments
- FT is comparable to RT in terms of energy-efficiency in static networks, and better in dynamic networks
- FT is 2 to 3 times faster than RT

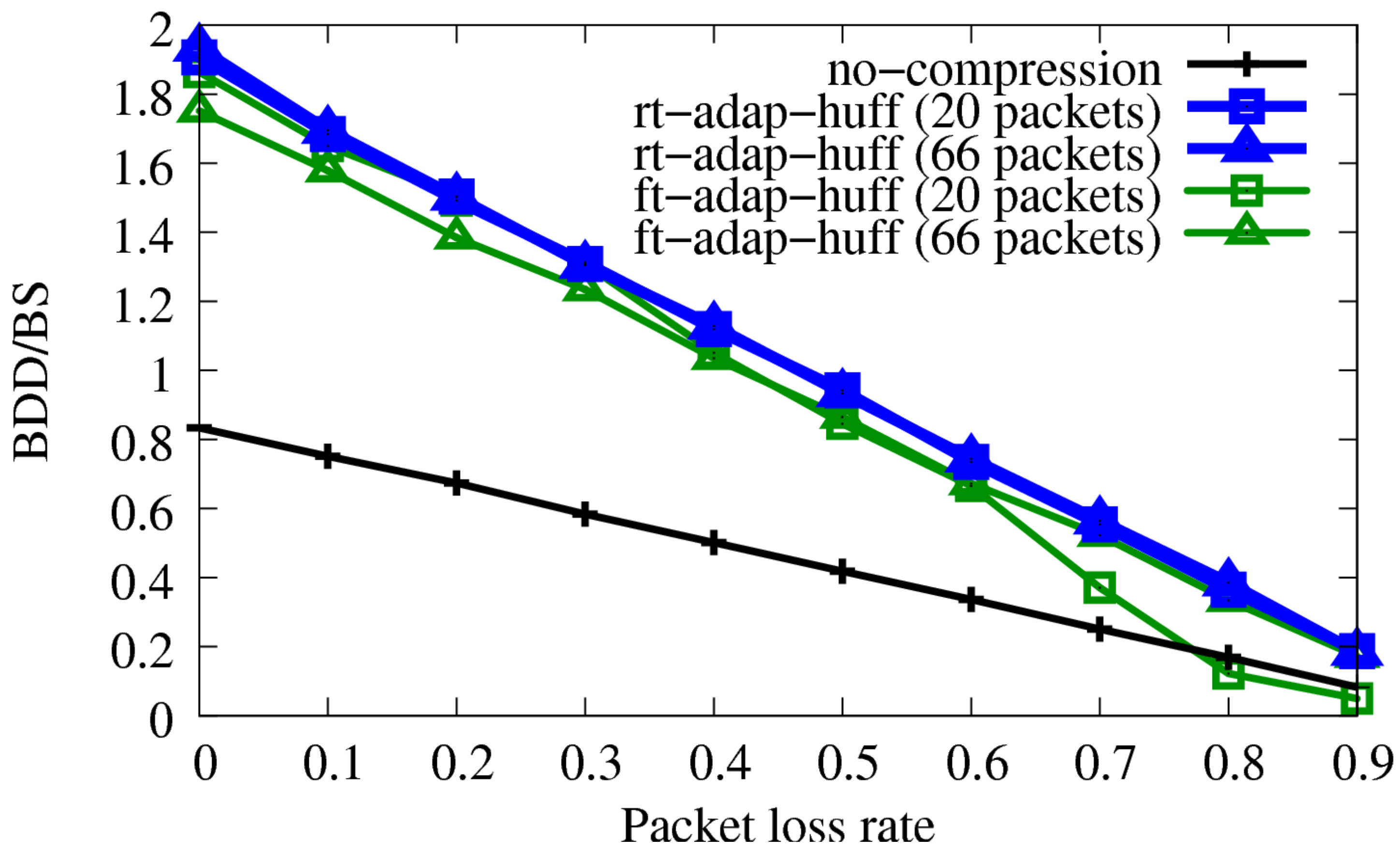
RT



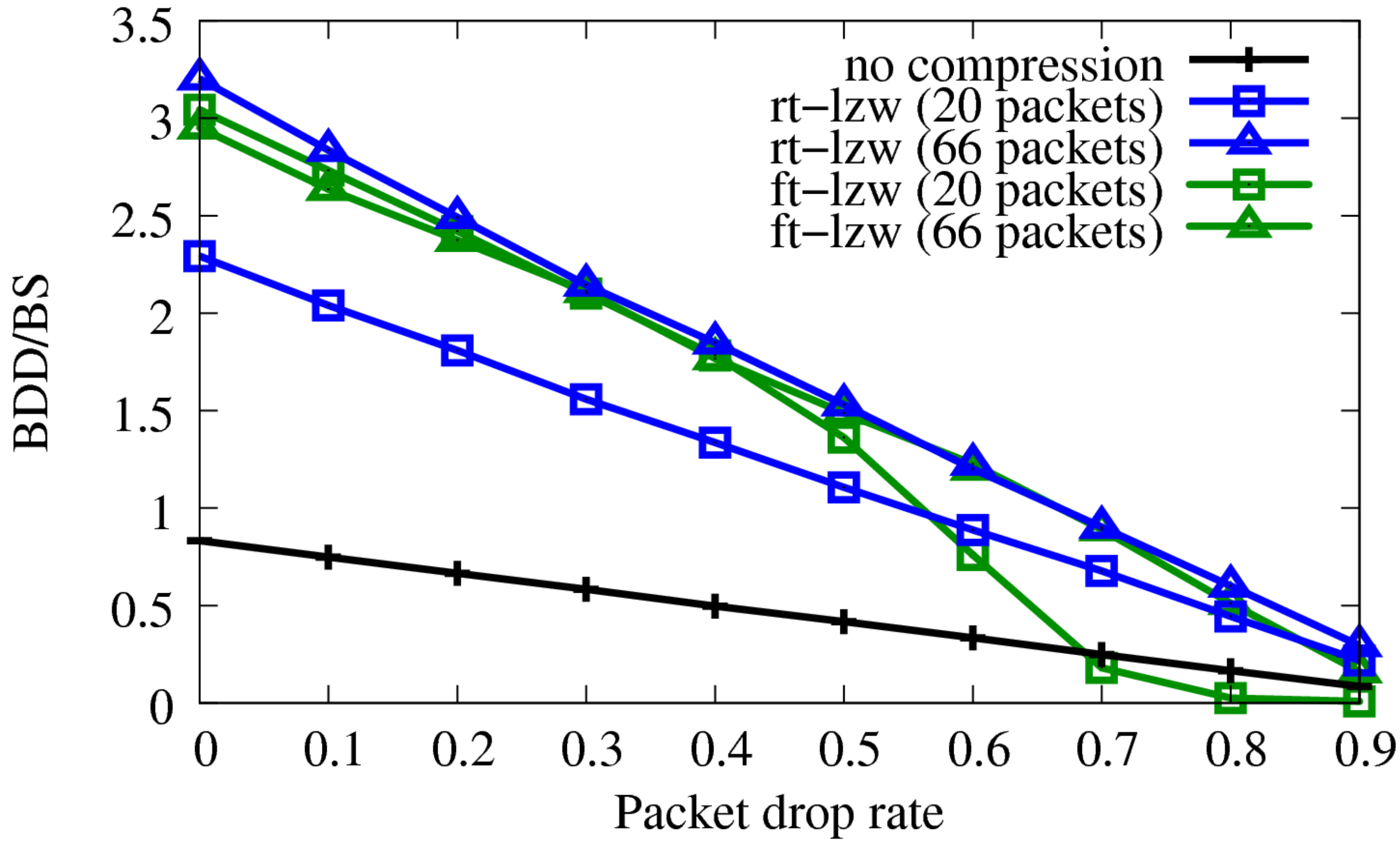
Thank you

FT

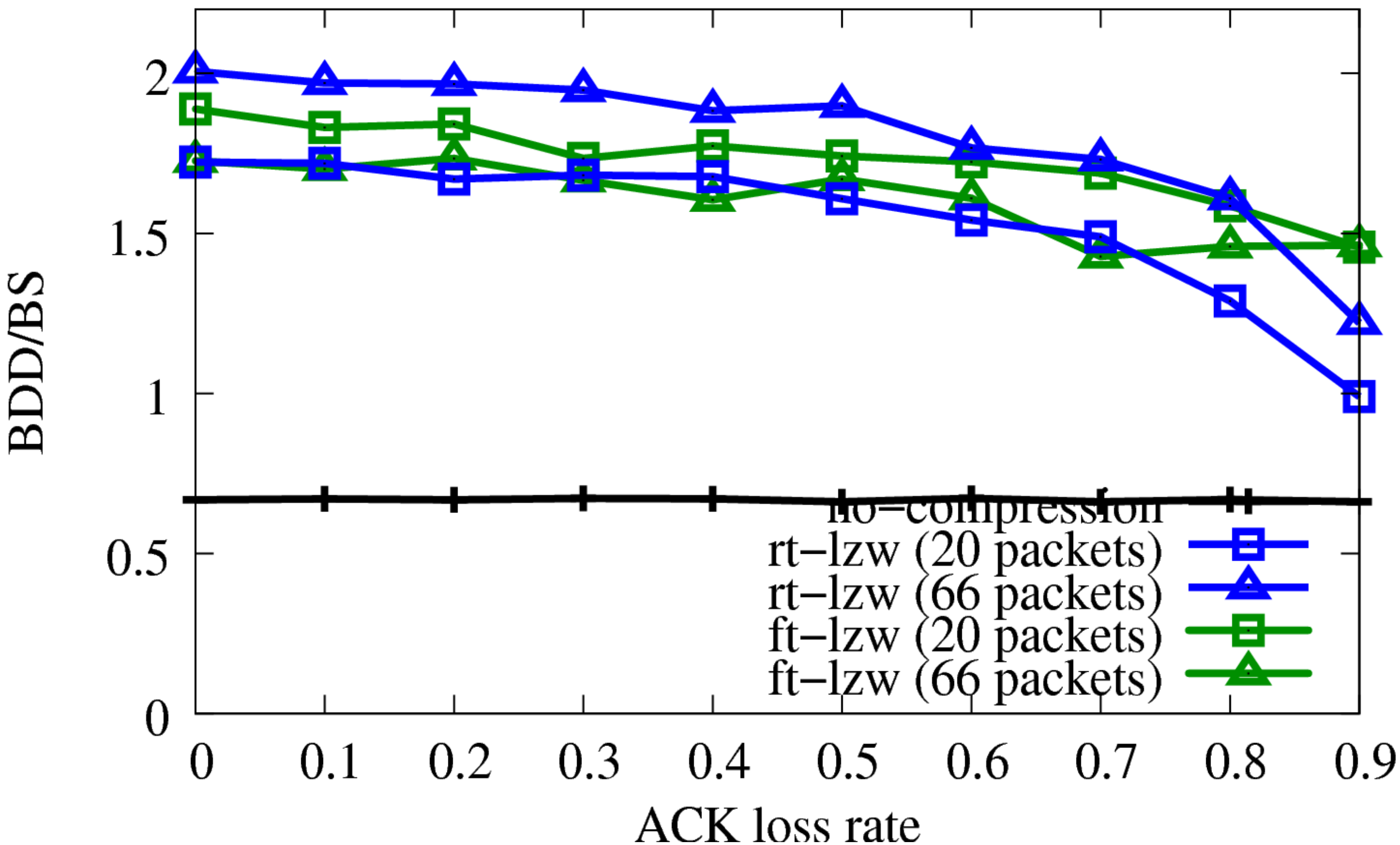
Evaluation - Energy-efficiency



Evaluation - Energy-efficiency



Evaluation - Energy-efficiency



Evaluation - Delay