TECHNISCHE UNIVERSITÄT DRESDEN

Netzwerkkodierung in Theorie und Praxis

Praktische Anwendungen der Netzwerkkodierung

Professor Dr.-Ing. Dr. h.c. Frank H.P. Fitzek

M.Sc. Juan Cabrera

Deutsche Telekom Chair of Communication Networks (ComNets)

Netzwerkkodierungstheorie

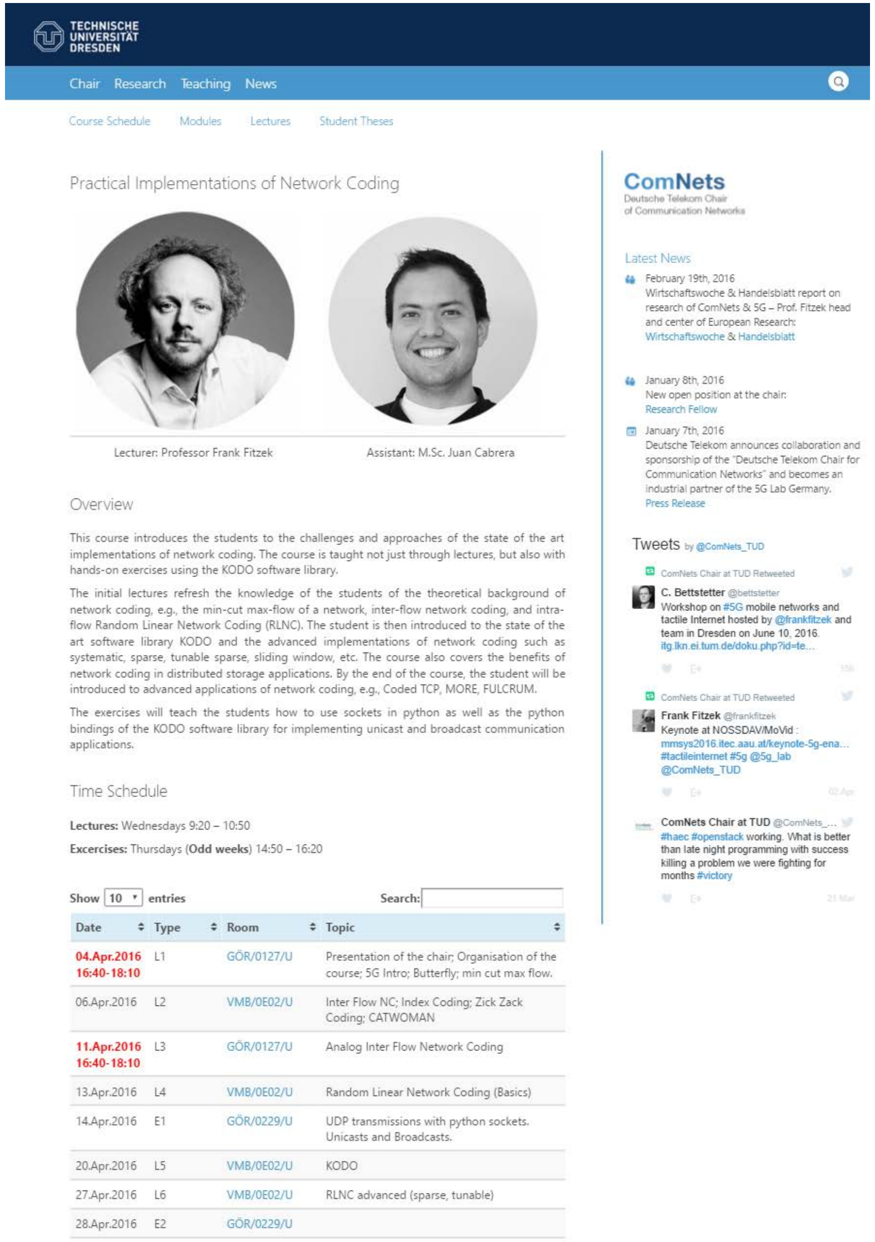
Professor Dr.-Ing. Eduard Jorswieck

Dipl.-Ing. Johannes Richter

Theoretische Nachrichtentechnik




Lecture / Exercise Dates - tinyurl.com/zooafld


TECHNISCHE UNIVERSITÄT DRESDEN

Chair Research Teaching News

Course Schedule Modules Lectures Student Theses

Practical Implementations of Network Coding

Lecturer: Professor Frank Fitzek

Assistant: M.Sc. Juan Cabrera

Overview

This course introduces the students to the challenges and approaches of the state of the art implementations of network coding. The course is taught not just through lectures, but also with hands-on exercises using the KODO software library.

The initial lectures refresh the knowledge of the students of the theoretical background of network coding, e.g., the min-cut max-flow of a network, inter-flow network coding, and intra-flow Random Linear Network Coding (RLNC). The student is then introduced to the state of the art software library KODO and the advanced implementations of network coding such as systematic, sparse, tunable sparse, sliding window, etc. The course also covers the benefits of network coding in distributed storage applications. By the end of the course, the student will be introduced to advanced applications of network coding, e.g., Coded TCP, MORE, FULCRUM.

The exercises will teach the students how to use sockets in python as well as the python bindings of the KODO software library for implementing unicast and broadcast communication applications.

Time Schedule

Lectures: Wednesdays 9:20 – 10:50

Excercises: Thursdays (**Odd weeks**) 14:50 – 16:20

Show 10 entries Search:

Date	Type	Room	Topic
04.Apr.2016 16:40-18:10	L1	GÖR/0127/U	Presentation of the chair; Organisation of the course; 5G Intro; Butterfly; min cut max flow.
06.Apr.2016	L2	VMB/0E02/U	Inter Flow NC; Index Coding; Zick Zack Coding; CATWOMAN
11.Apr.2016 16:40-18:10	L3	GÖR/0127/U	Analog Inter Flow Network Coding
13.Apr.2016	L4	VMB/0E02/U	Random Linear Network Coding (Basics)
14.Apr.2016	E1	GÖR/0229/U	UDP transmissions with python sockets. Unicasts and Broadcasts.
20.Apr.2016	L5	VMB/0E02/U	KODO
27.Apr.2016	L6	VMB/0E02/U	RLNC advanced (sparse, tunable)
28.Apr.2016	E2	GÖR/0229/U	

ComNets
Deutsche Telekom Chair of Communication Networks

Latest News

February 19th, 2016
Wirtschaftswoche & Handelsblatt report on research of ComNets & 5G – Prof. Fitzek head and center of European Research:
Wirtschaftswoche & Handelsblatt

January 8th, 2016
New open position at the chair:
Research Fellow

January 7th, 2016
Deutsche Telekom announces collaboration and sponsorship of the "Deutsche Telekom Chair for Communication Networks" and becomes an industrial partner of the 5G Lab Germany.
Press Release

Tweets by @ComNets_TUD

ComNets Chair at TUD Retweeted
C. Bettstetter @bettstetter
Workshop on #5G mobile networks and tactile Internet hosted by @frankfitzek and team in Dresden on June 10, 2016.
itg.lkn.ei.tum.de/doku.php?id=te...

ComNets Chair at TUD Retweeted
Frank Fitzek @frankfitzek
Keynote at NOSSDAV/MoVid :
mmsys2016.itec.aau.at/keynote-5g-ena...
#tactileinternet #5g @5g_lab
@ComNets_TUD

ComNets Chair at TUD @ComNets_...
#haec #openstack working. What is better than late night programming with success killing a problem we were fighting for months #victory

- Here all information for the lecture and the exercise can be found.
- Slides
- Links
  - Steinwurf
  - Python
  - KODOMARK (google play)

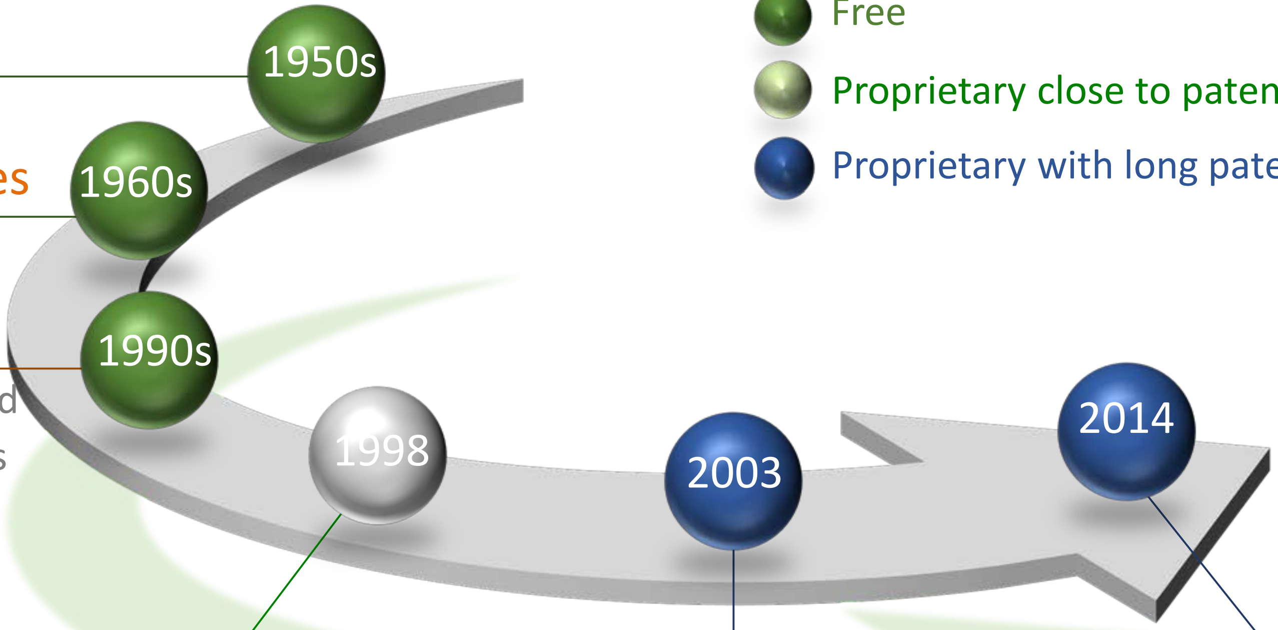
Please check every week!

Random Linear Network Coding

Lecture 4

Coding History

Evolution of Coding



Free

Proprietary close to patent expiry

Proprietary with long patent life

Block Codes

Convolutional Codes

Modern Codes

- LDPCs – patent expired
- Turbo Codes – patents expired or expiring

Rateless Codes

Raptor and related codes

- Rate-less (refinement to free E2E)
- Still E2E, still static

Network Codes

- RLNC enables network coding
- Deterministic construction for special cases
  - Index Coding
  - CATWOMAN (Linux 3.10)

Fulcrum Codes

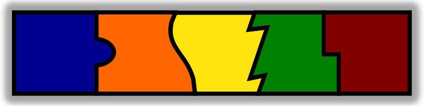
- RLNC-enabled
  - *Fluid complexity* (flexible field size)
  - Breaks performance-overhead trade-off

The Technology: RLNC

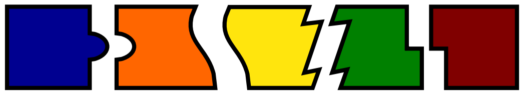
At the heart of many communications problems is a collectors' problem.

Traditional Approach

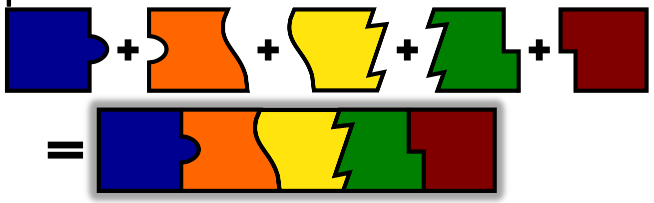
- Data broken into pieces



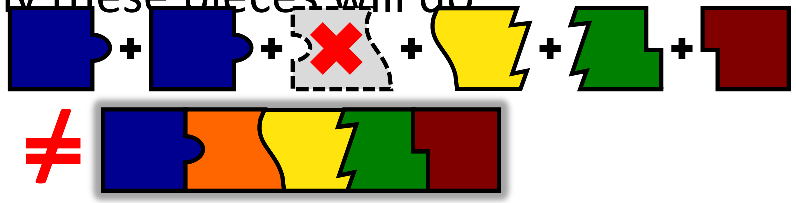
- k-piece data set → k pieces



- All pieces needed

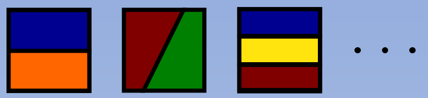


- Only these pieces will do

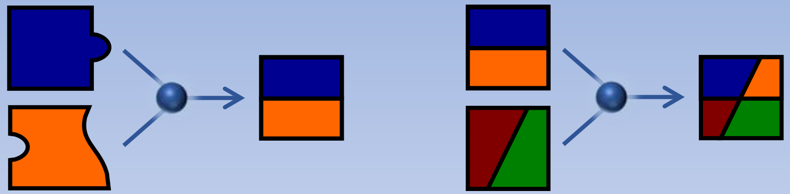


RLNC

- Mixtures created from pieces



- **Any node can create mixtures**

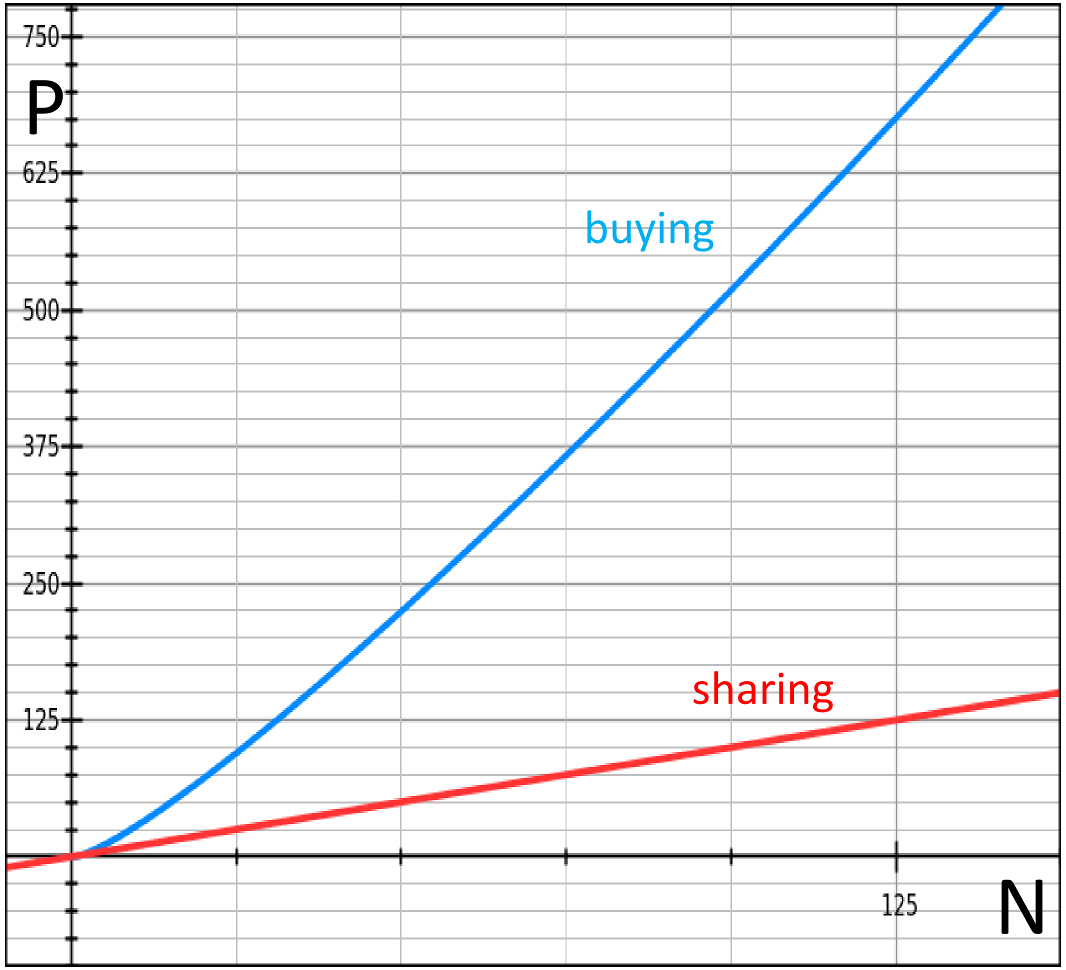


- Many mixtures possible

- Any k mixtures will do

TECHNISCHE UNIVERSITÄT DRESDEN

Collector‘s Problem



$$P = N * (\ln(N) + 0.577)$$

Random Linear Network Coding

Orignal packets

$$\begin{pmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \end{pmatrix} = \begin{pmatrix} \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} & \alpha_{1,4} & \alpha_{1,5} & \alpha_{1,6} \\ \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} & \alpha_{2,4} & \alpha_{2,5} & \alpha_{2,6} \\ \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} & \alpha_{3,4} & \alpha_{3,5} & \alpha_{3,6} \\ \alpha_{4,1} & \alpha_{4,2} & \alpha_{4,3} & \alpha_{4,4} & \alpha_{4,5} & \alpha_{4,6} \\ \alpha_{5,1} & \alpha_{5,2} & \alpha_{5,3} & \alpha_{5,4} & \alpha_{5,5} & \alpha_{5,6} \\ \alpha_{6,1} & \alpha_{6,2} & \alpha_{6,3} & \alpha_{6,4} & \alpha_{6,5} & \alpha_{6,6} \end{pmatrix} \boxed{\begin{pmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{pmatrix}}$$

Gaussian elimination *n* x *n matrix* requires $An^3 + Bn^2 + Cn$ operations.

Random Linear Network Coding

$$\begin{pmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \end{pmatrix} = \overbrace{\begin{pmatrix} \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} & \alpha_{1,4} & \alpha_{1,5} & \alpha_{1,6} \\ \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} & \alpha_{2,4} & \alpha_{2,5} & \alpha_{2,6} \\ \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} & \alpha_{3,4} & \alpha_{3,5} & \alpha_{3,6} \\ \alpha_{4,1} & \alpha_{4,2} & \alpha_{4,3} & \alpha_{4,4} & \alpha_{4,5} & \alpha_{4,6} \\ \alpha_{5,1} & \alpha_{5,2} & \alpha_{5,3} & \alpha_{5,4} & \alpha_{5,5} & \alpha_{5,6} \\ \alpha_{6,1} & \alpha_{6,2} & \alpha_{6,3} & \alpha_{6,4} & \alpha_{6,5} & \alpha_{6,6} \end{pmatrix}}^{\text{coding coefficients}} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{pmatrix}$$

Gaussian elimination *n* x *n matrix* requires $An^3 + Bn^2 + Cn$ operations.

Random Linear Network Coding

$$\underbrace{\begin{pmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \end{pmatrix}}_{\text{coded packets}} = \begin{pmatrix} \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} & \alpha_{1,4} & \alpha_{1,5} & \alpha_{1,6} \\ \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} & \alpha_{2,4} & \alpha_{2,5} & \alpha_{2,6} \\ \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} & \alpha_{3,4} & \alpha_{3,5} & \alpha_{3,6} \\ \alpha_{4,1} & \alpha_{4,2} & \alpha_{4,3} & \alpha_{4,4} & \alpha_{4,5} & \alpha_{4,6} \\ \alpha_{5,1} & \alpha_{5,2} & \alpha_{5,3} & \alpha_{5,4} & \alpha_{5,5} & \alpha_{5,6} \\ \alpha_{6,1} & \alpha_{6,2} & \alpha_{6,3} & \alpha_{6,4} & \alpha_{6,5} & \alpha_{6,6} \end{pmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{pmatrix}$$

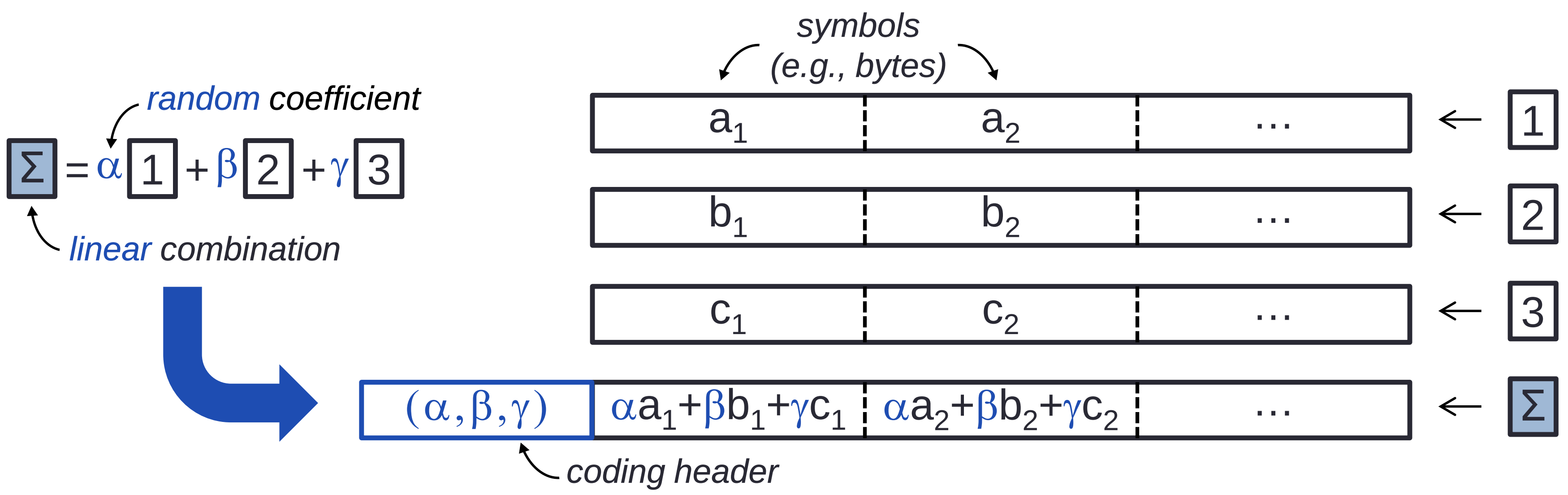
Gaussian elimination *n* x *n matrix* requires $An^3 + Bn^2 + Cn$ operations.

Random Linear Network Coding

$$\begin{pmatrix} C_1 \\ \vdots \\ C_G \\ C_{G+1} \\ \vdots \\ C_K \end{pmatrix} = \begin{pmatrix} \alpha_{1,1} & \cdots & \alpha_{1,G} \\ \vdots & \ddots & \vdots \\ \alpha_{G,1} & \cdots & \alpha_{G,G} \\ \alpha_{G+1,1} & \cdots & \alpha_{G+1,G} \\ \vdots & \ddots & \vdots \\ \alpha_{K,1} & \cdots & \alpha_{K,G} \end{pmatrix} \begin{pmatrix} P_1 \\ \vdots \\ P_G \end{pmatrix}$$

Rateless code: can output any number of coded packets.
(such as Fountain codes, but better than RS)

How does RLNC work?

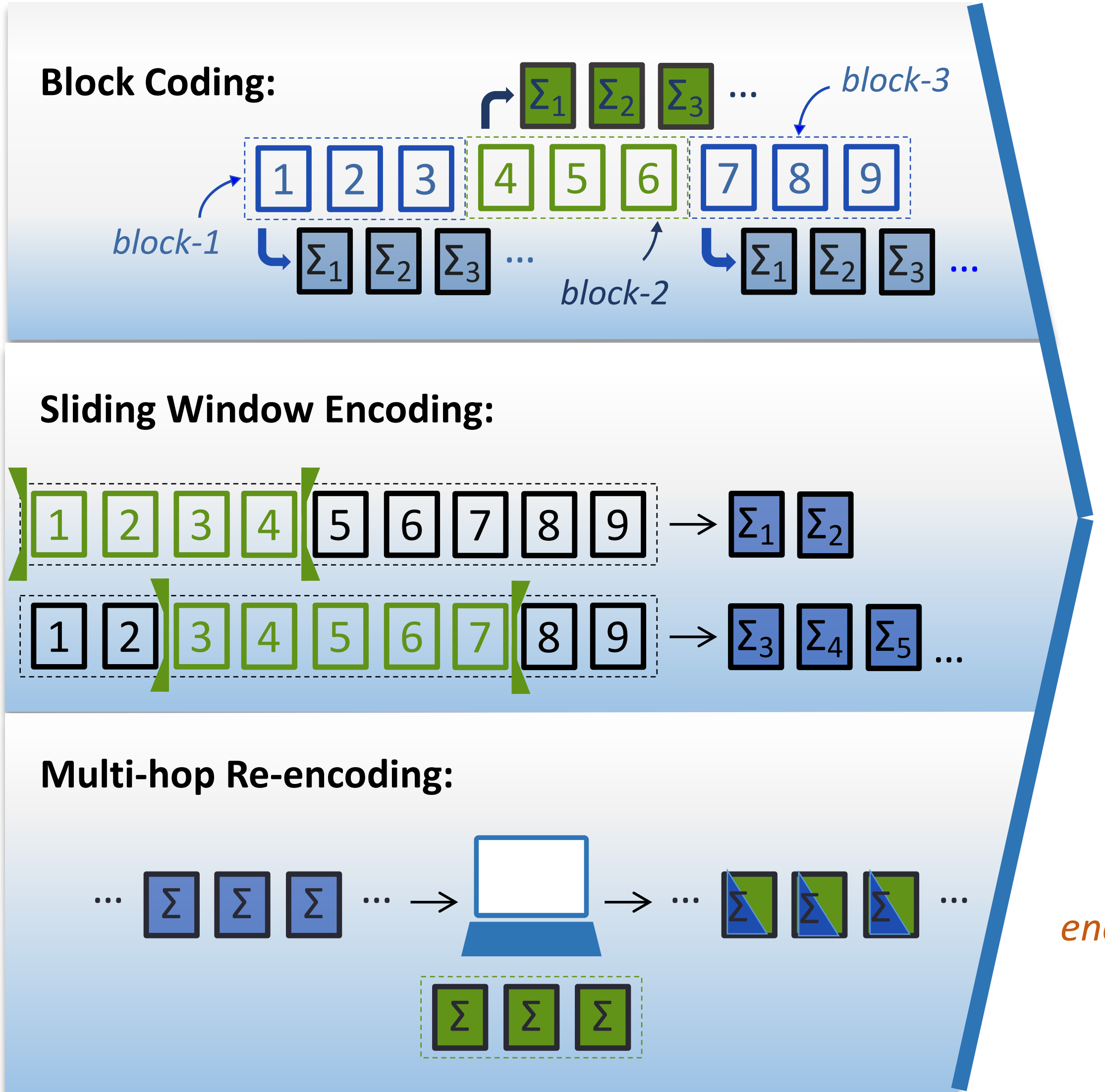


- Random Generation of Coefficients
- Code Embedded within Data

- No State Tracking
- New IT Paradigms

RLNC Use Case

Multiple encoding schemes



One decoding scheme

(simple equation-solving)

$$\boxed{1} = a'_1\,\boxed{\Sigma_1} + b'_1\,\boxed{\Sigma_2} + g'_1\,\boxed{\Sigma_3}$$

$$\boxed{2} = a'_2\,\boxed{\Sigma_1} + b'_2\,\boxed{\Sigma_2} + g'_2\,\boxed{\Sigma_3}$$

$$\boxed{3} = \boxed{1} + b'_3\,\boxed{\Sigma_2} + g'_3\,\boxed{\Sigma_3}$$

obtained through Gaussian Elimination

Can decode using both encoded and un-encoded packets

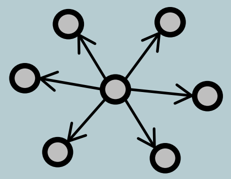
RLNC: The Technology

Coding Today

(all End-to-End)

Classical

Multicast

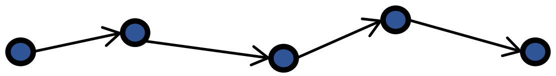


Coding Tomorrow with RLNC

Classical + Sliding Window Encoding

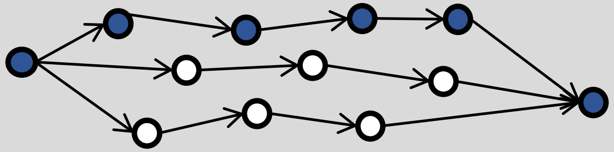
Real time video streaming, TCP, SDN…

Multihop



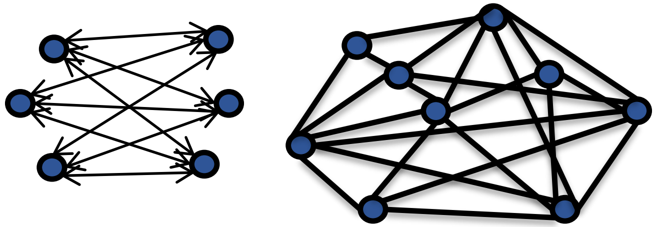
Edge caches, wireless mesh, reliable multicast, satellites, small relay topologies, SDN…

Multipath



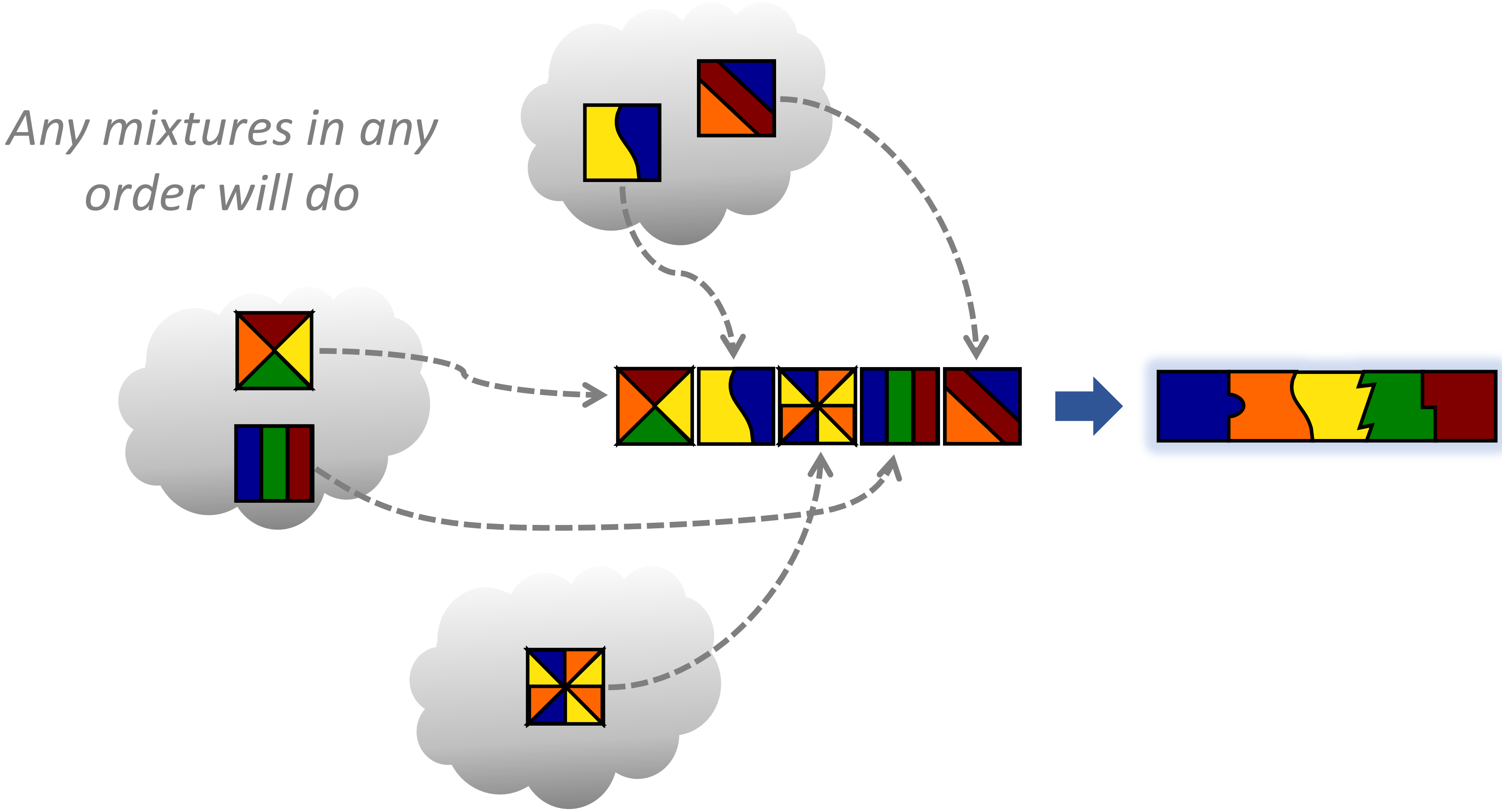
Multi-source streaming Multipath TCP, channel bundling, heterogeneous network combining, SDN…

Multisource – Multi-destination / Mesh



Distributed cloud, SDN, advanced mesh (IoT, car2car, M2M, smart grid) …

Multipath – Multicloud

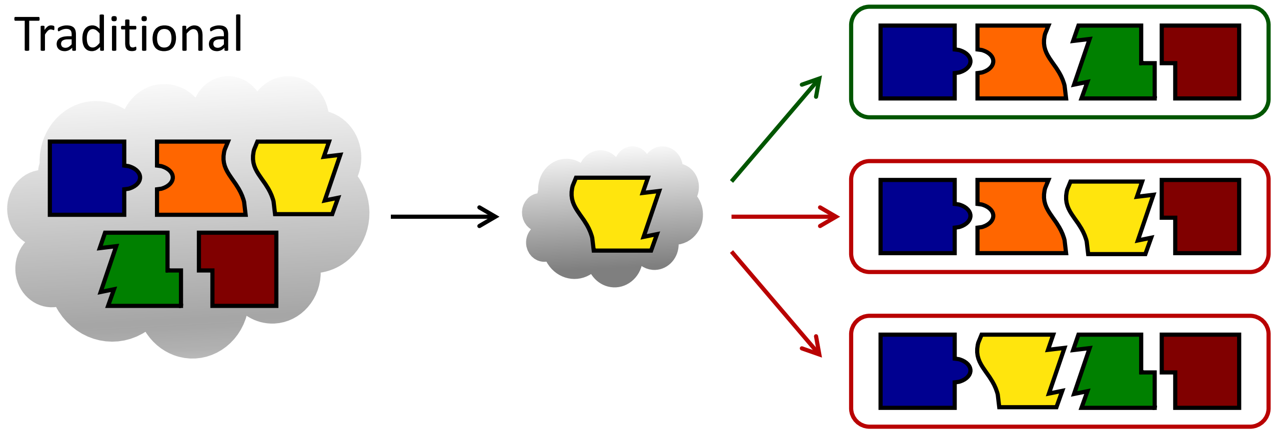


- RLNC Enables "Stateless Communications"

TECHNISCHE UNIVERSITÄT DRESDEN

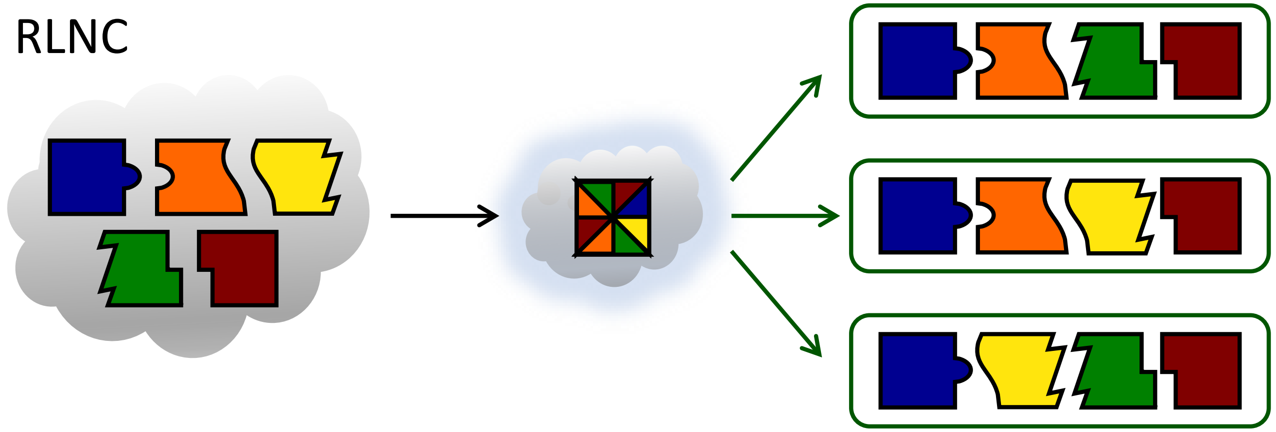
Coded Edge Caching

Traditional



- One small cache can't satisfy everyone
- Only the device missing the piece in the edge cache can reconstruct

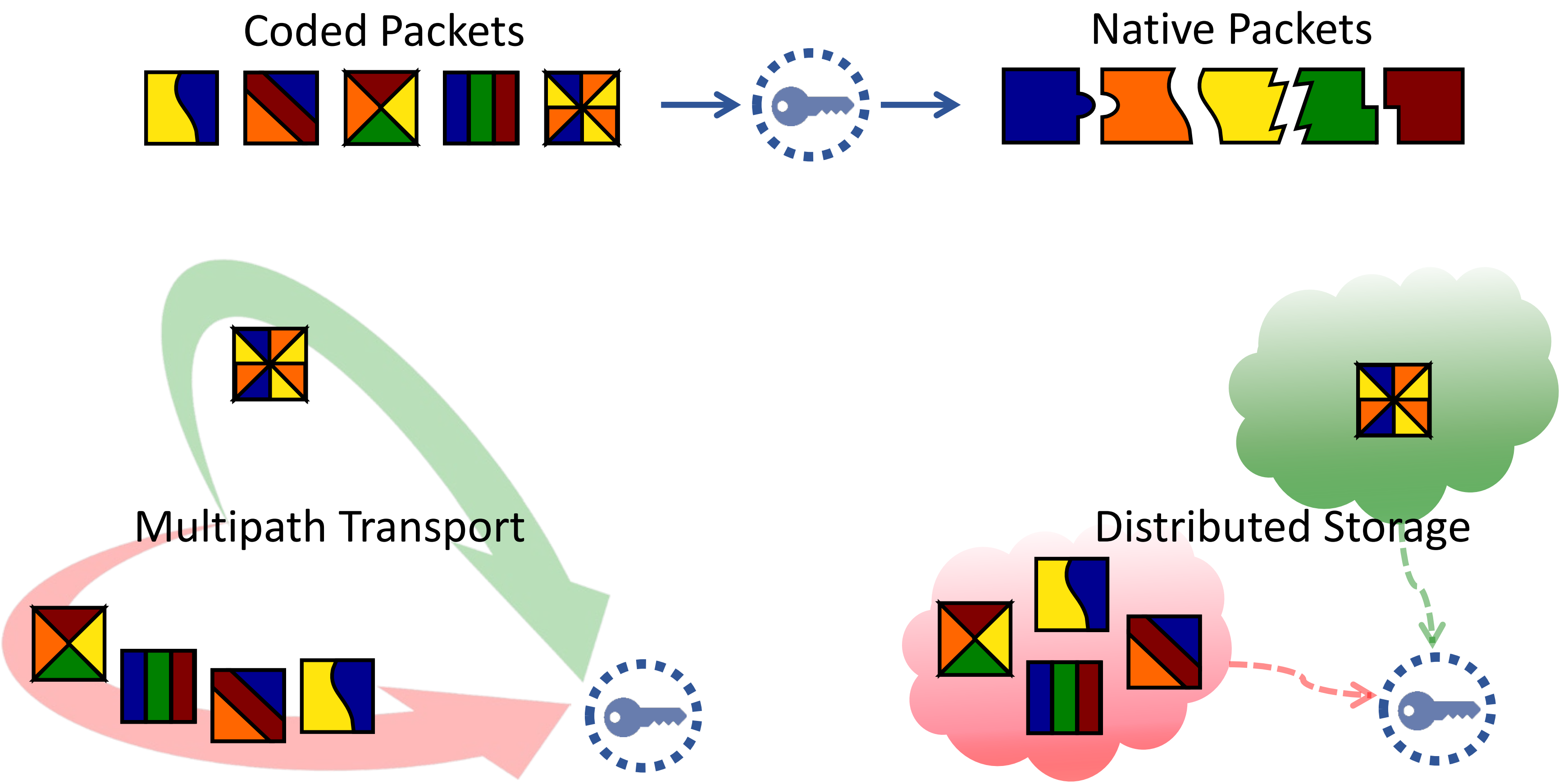
RLNC



- Mixtures enable all nodes to reconstruct
- An example of RLNC's enhancement of non-coded systems

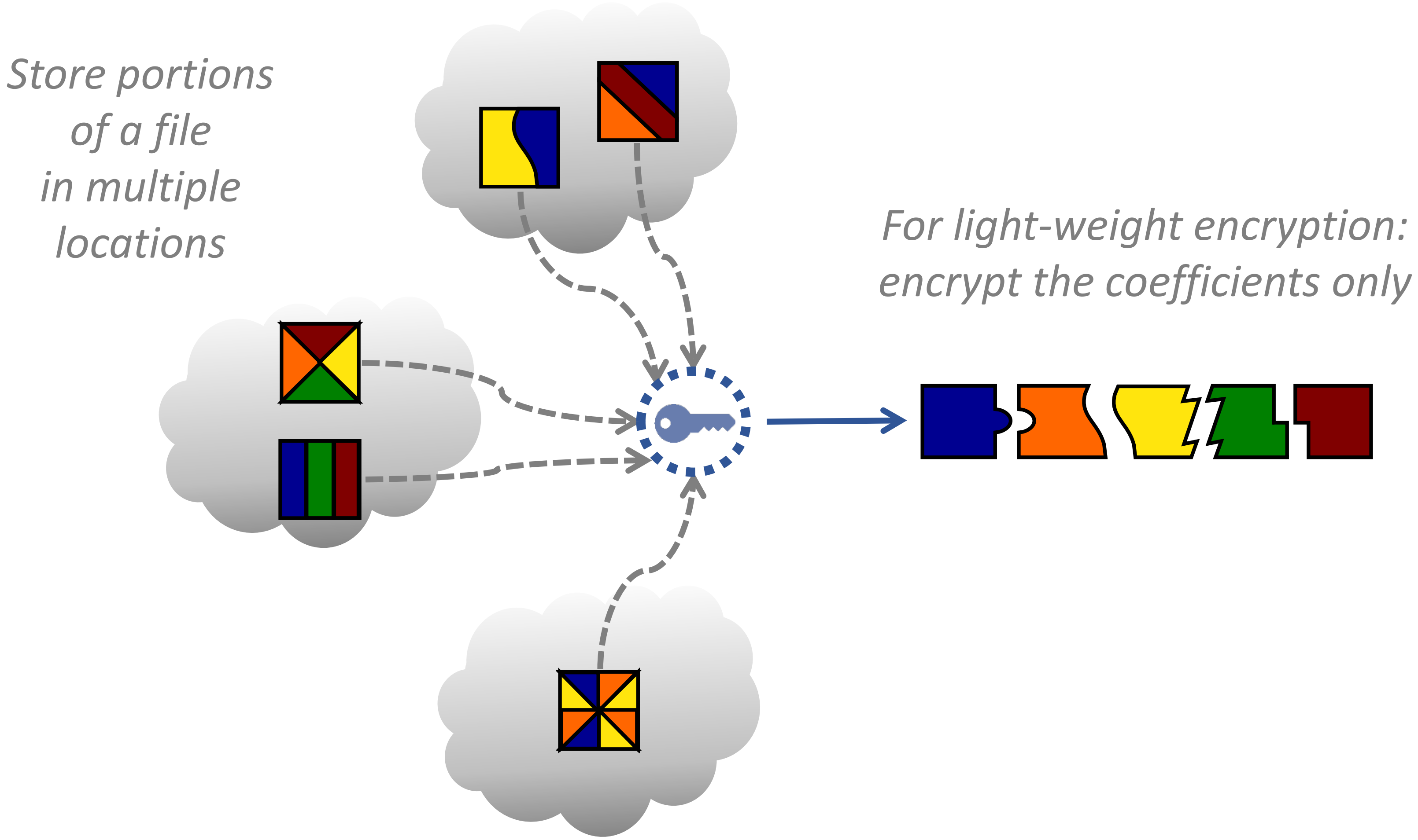
▪ A Little RLNC Can Go a Long Way

Coding as an Additional Security Measure



- Data on a given path/cloud acts as a cypher

RLNC Improves Cloud Security



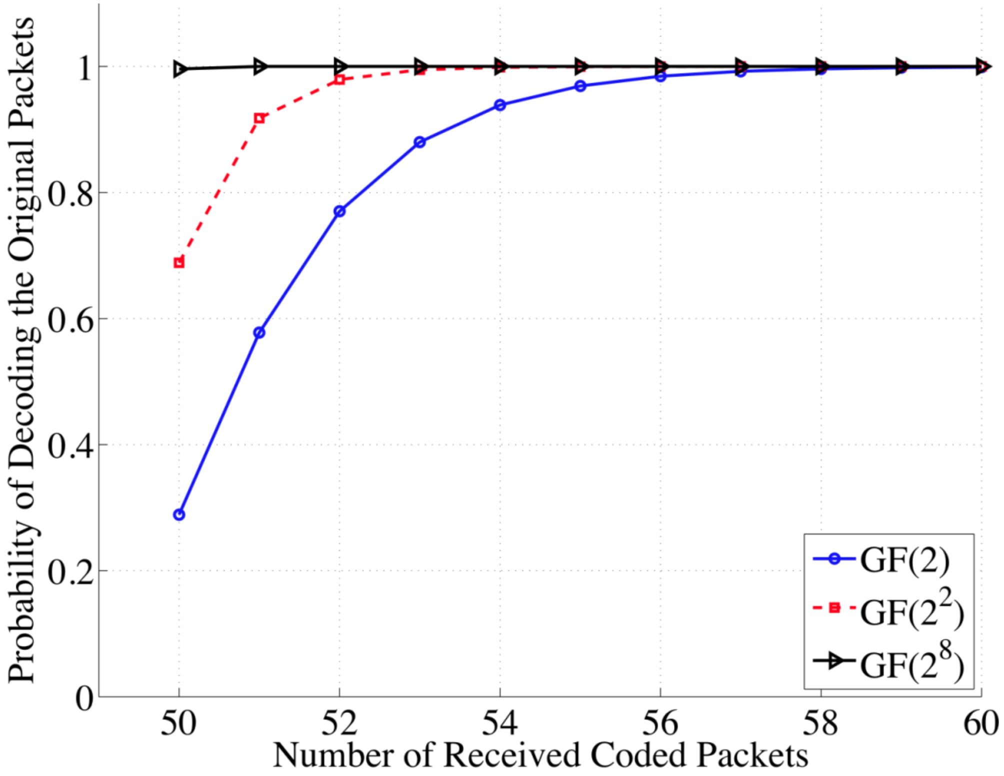
- Mixture equations "unlock" data from mixtures

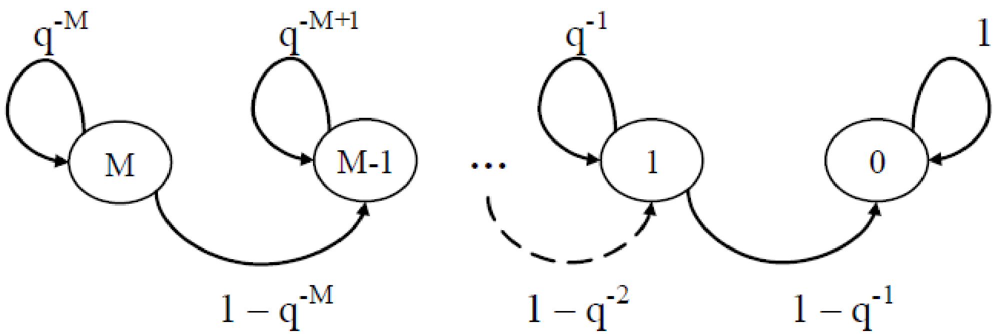
Key Parameters of RLNC

- Generation size: number of packets that are (currently) coded together.
- Field size: number of elements in the finite field

- Both have an impact on:
  - Performance
  - Complexity

Field and Generation Size





Small field sizes are resulting in linear dependent coded packets.

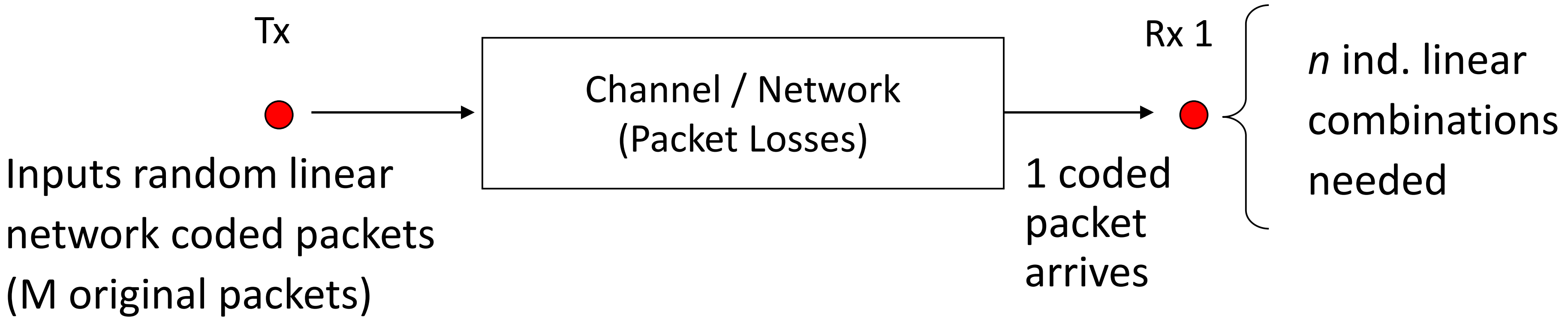
1.6 packets extra per generation in case of binary field sizes.

Large fields sizes (2^8 or higher) have nearly no linear dependency.

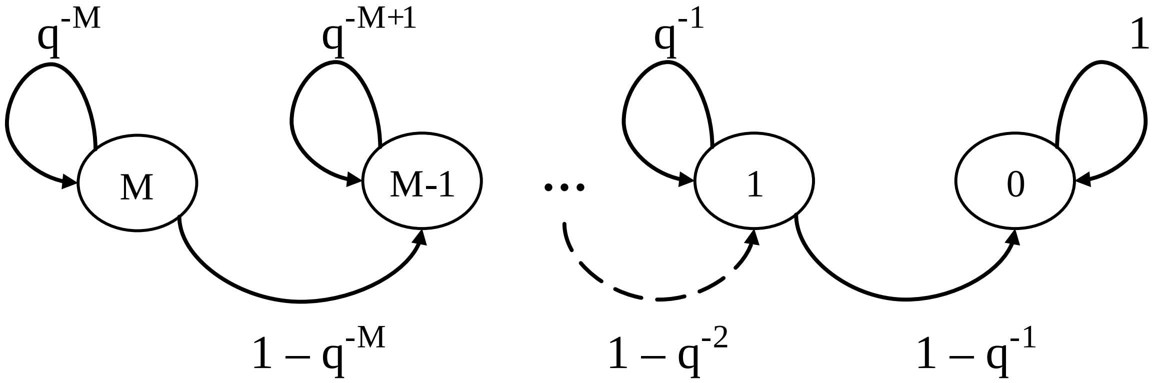
Theory is aiming for large field sizes and large generation sizes!

Field Size Analysis

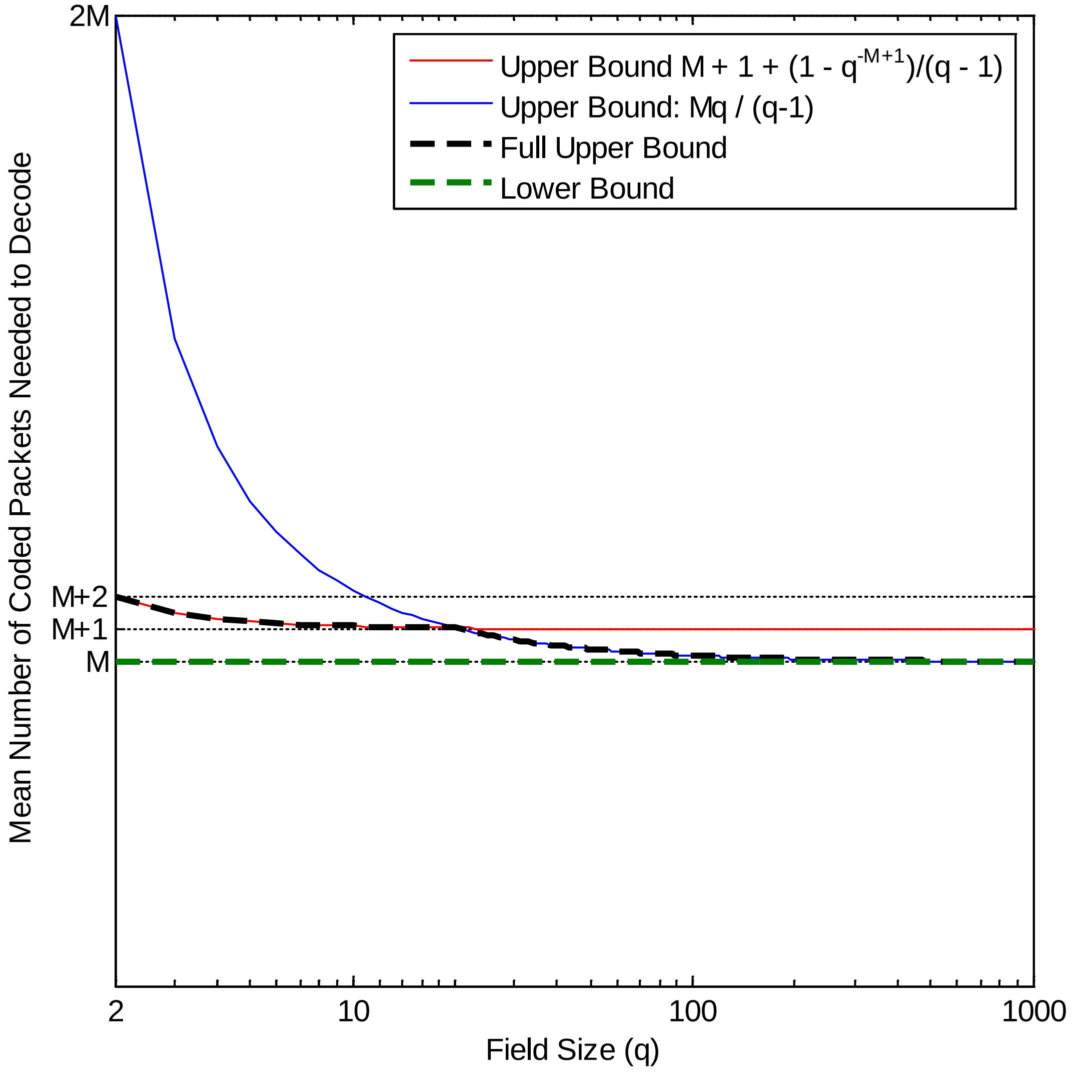
Field Size Analysis for RLNC



–Modeled as a Markov chain



Field Size Analysis



$$E[N_c] = \sum_{k=1}^{M} \frac{1}{1-q^{-k}}$$

$$\leq \min\left(M\frac{q}{q-1}, M+1+\frac{1+q^{-M+1}}{q-1} \right)$$

If *M* is large:

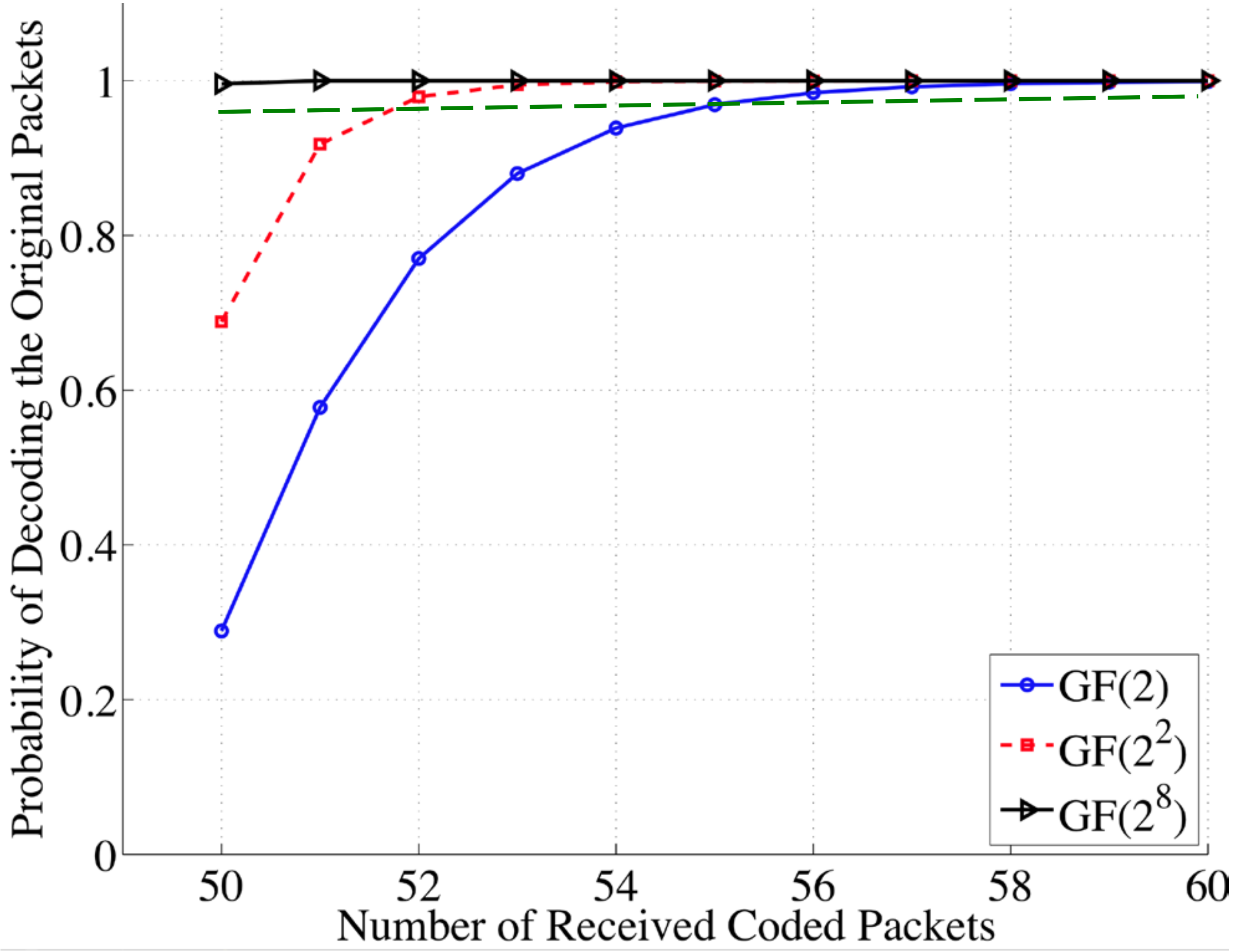
–Little overhead

–Small performance degradation

Later today you will simulate

–Task 7: single link, GF(2), generation size 8

Field Size Analysis: Distribution



DEMO

Recoding Potential

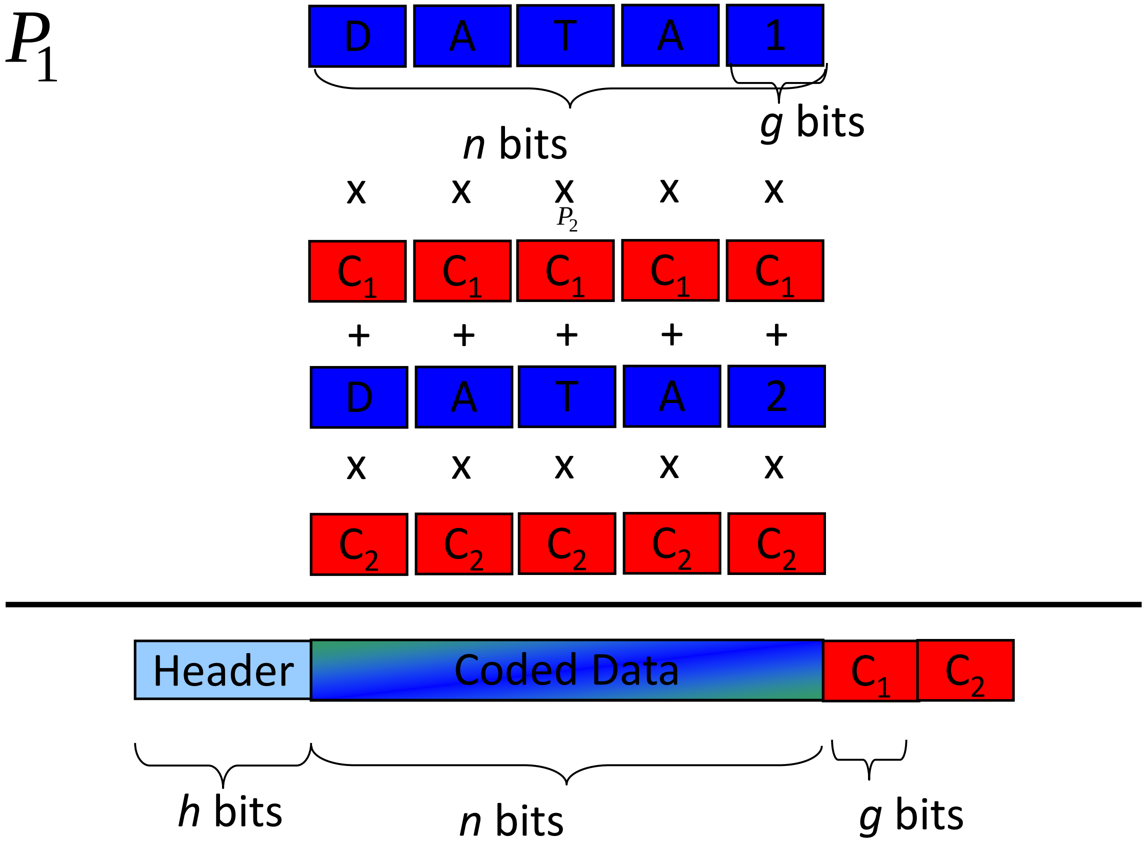
Generating a Coded Packet

- Generating a linear network coded packet (CP)

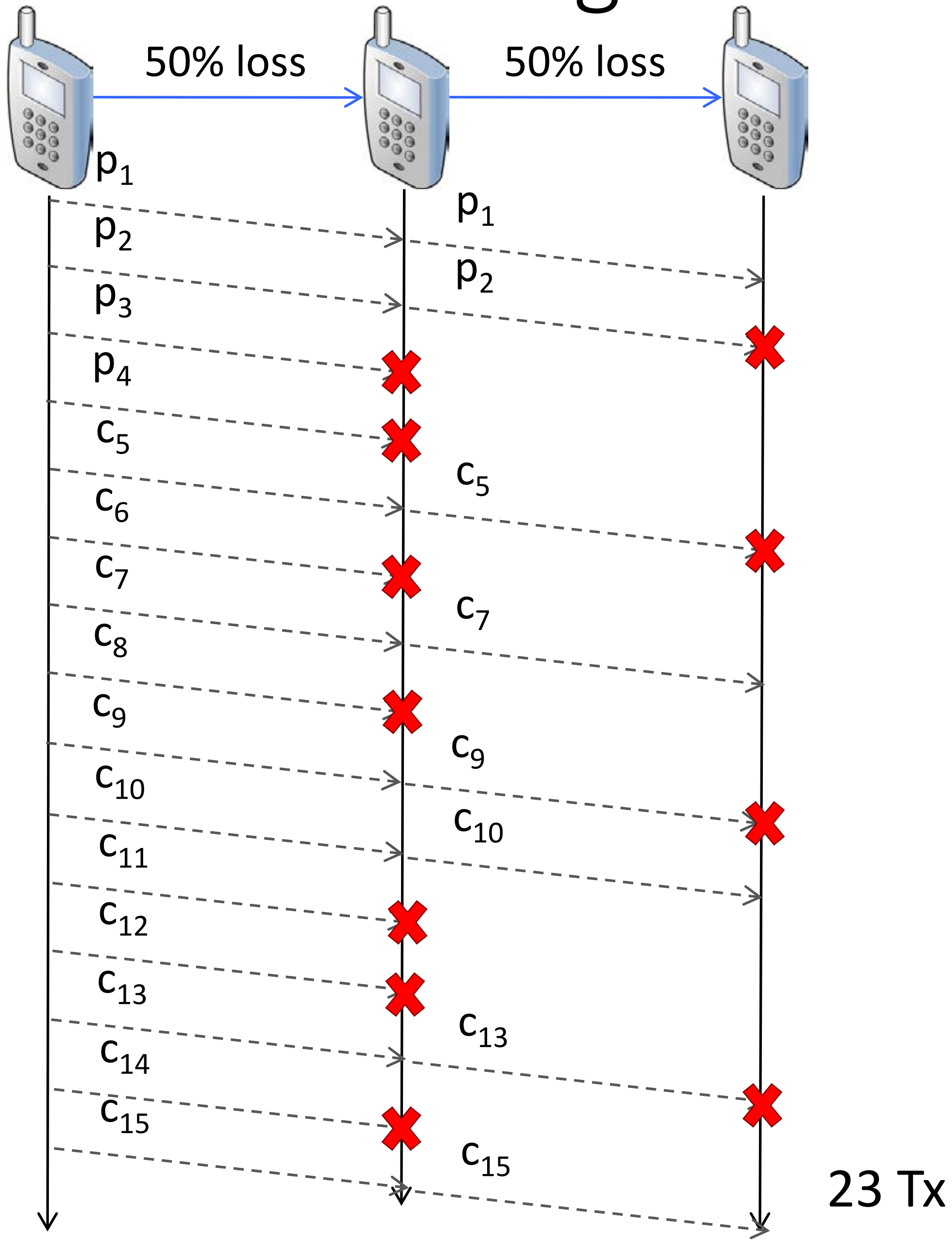
$$CP_j = \sum_i C_i P_i$$

- Operations over finite field of size.

e.g. g = 8 bits, q = 256



No Recoding



- Simple operation: forward
- Structure of code is preserved
- Issues
  - Delay per batch of packets
  - Missing transmission opportunities
  - Equivalent loss probability: compounding each channel's loss

Loss of channel *i*: e_i

Equivalent success prob of a packet:

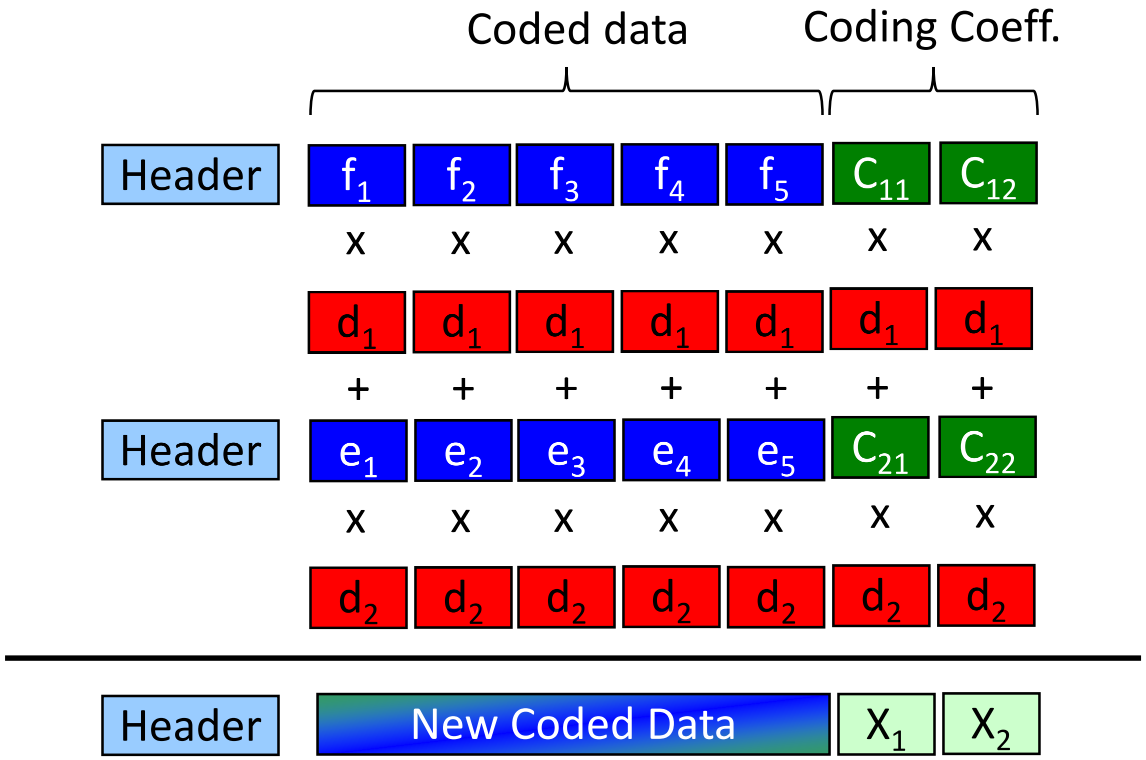
$(1-e_1)(1-e_2)$

$c_i = \sum \alpha_{ij} p_j$

Packet loss

Recoding Packets

- Generating a new linear network coded packet (CP)

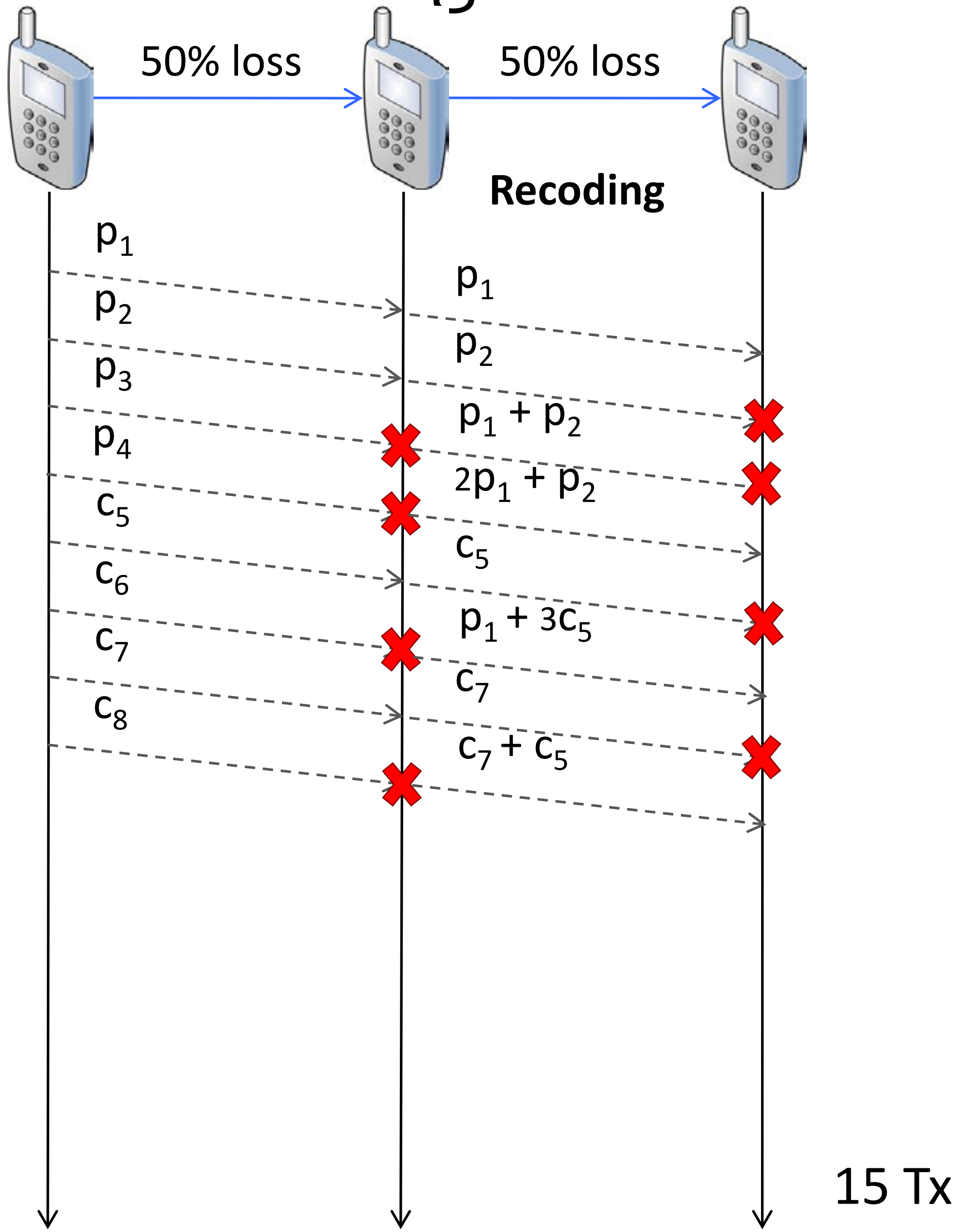


$X_1 = d_1\, C_{11} + d_2\, C_{21}$ and $X_2 = d_1\, C_{12} + d_2\, C_{22}$

Recall that: $f = C_{11}P_1 + C_{12}\, P_2$ and $e = C_{21}P_1 + C_{22}\, P_2$

Thus, $d_1f + d_2e = d_1C_{11}P_1 + d_1C_{12}\, P_2 + d_2C_{21}P_1 + d_2C_{22}\, P_2 = X_1P_1 + X_2P_2$

Recoding



- A bit more complex: recode
  - Equivalent to encoding in worst case
  - Not so bad
- Structure of code may be changed
  - Some exceptions
- Issues
  - If left unchecked, can have unnecessary transmissions, e.g., c_8
  - Additional processing needed

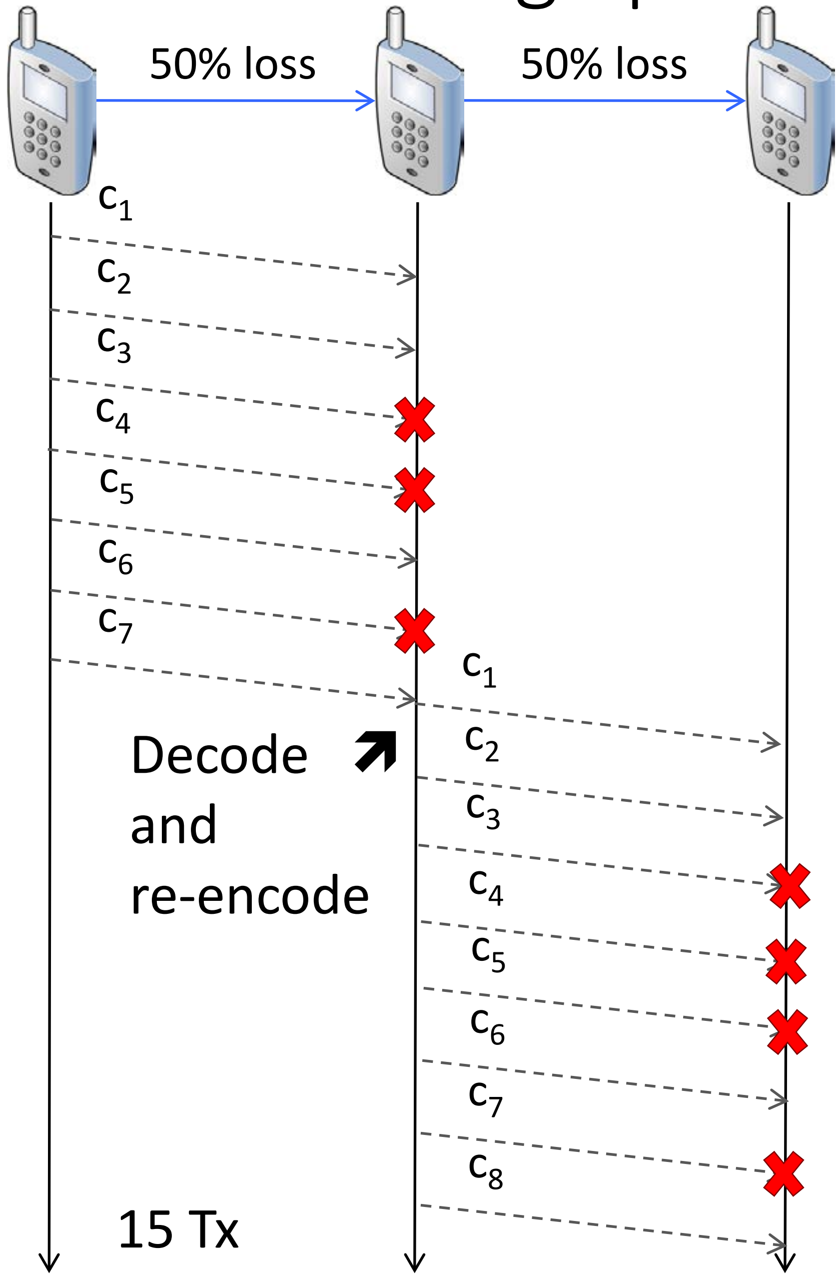
Advantage: equivalent success prob of a *linear combination*:

$$\min\{1-e_1, 1-e_2\}$$

$$c_i = \Sigma\, \alpha_{ij}\, p_j$$

Packet loss

Is “recoding” possible with other linear codes?



Consider Reed-Solomon, LT, etc

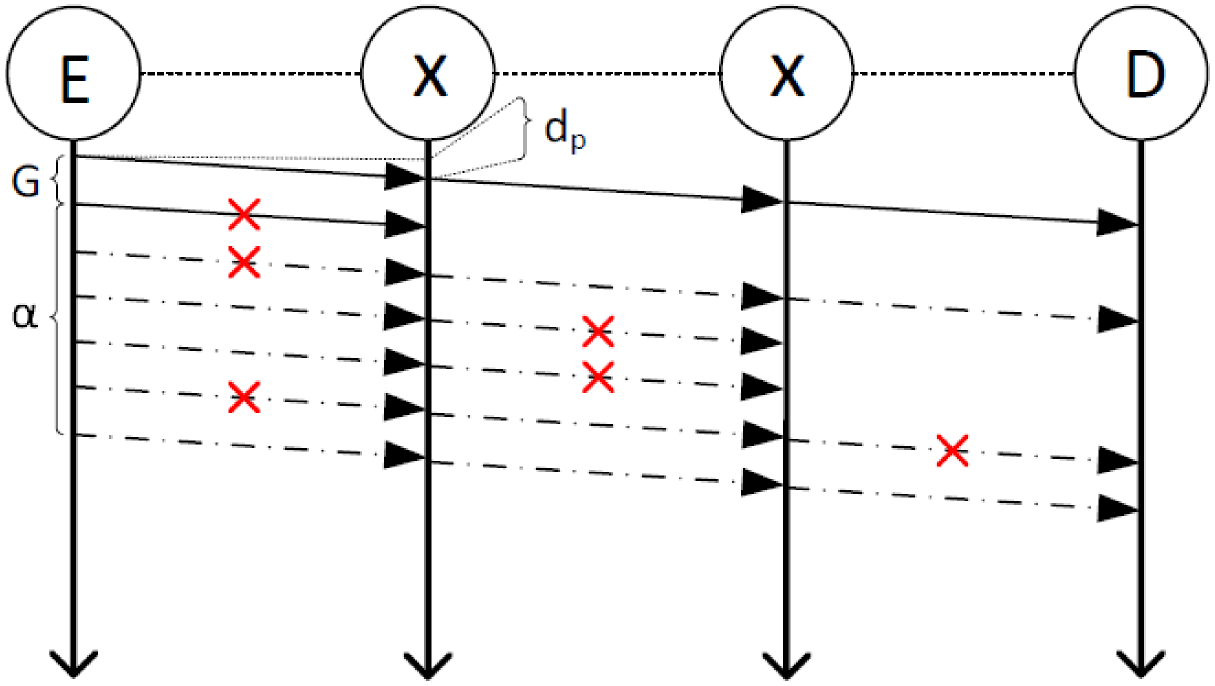
- Structure is not composable
  - Mixing coded packets does not produce a “valid” coded packet
  - Different structure, properties are lost
- Recoding means receiving enough coded packets, decode, and *then* re-encode
- Issues
  - Delay per batch
  - Computational effort
  - Can also have unnecessary Tx
- Success prob of a *linear combination*: $\min\{1-e_1, 1-e_2\}$

Software Defined Network (SDN) Example

Software Defined Networks

- End to End Coding Schemes: Store and Forward

$$P_{E2E} = \sum_{h=1}^{H} G \cdot \left(\frac{1}{(1-\epsilon)^h} \right) \qquad D_{E2E} = \left(G \cdot \left(\frac{1}{(1-\epsilon)^H} \right) + (H-1)\right) \cdot d_p$$

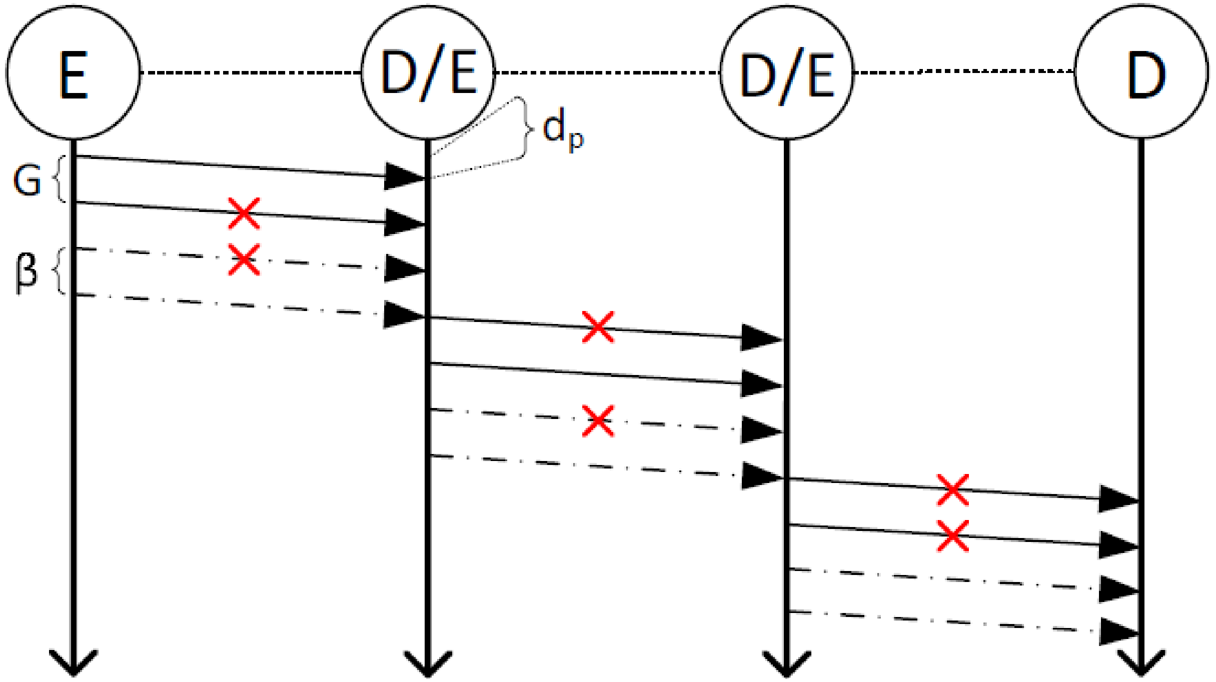


Software Defined Networks

- Hop by Hop Coding Scheme: Store and Forward

$$P_{HbH} = G \cdot H \cdot \left(\frac{1}{1-\epsilon}\right)$$

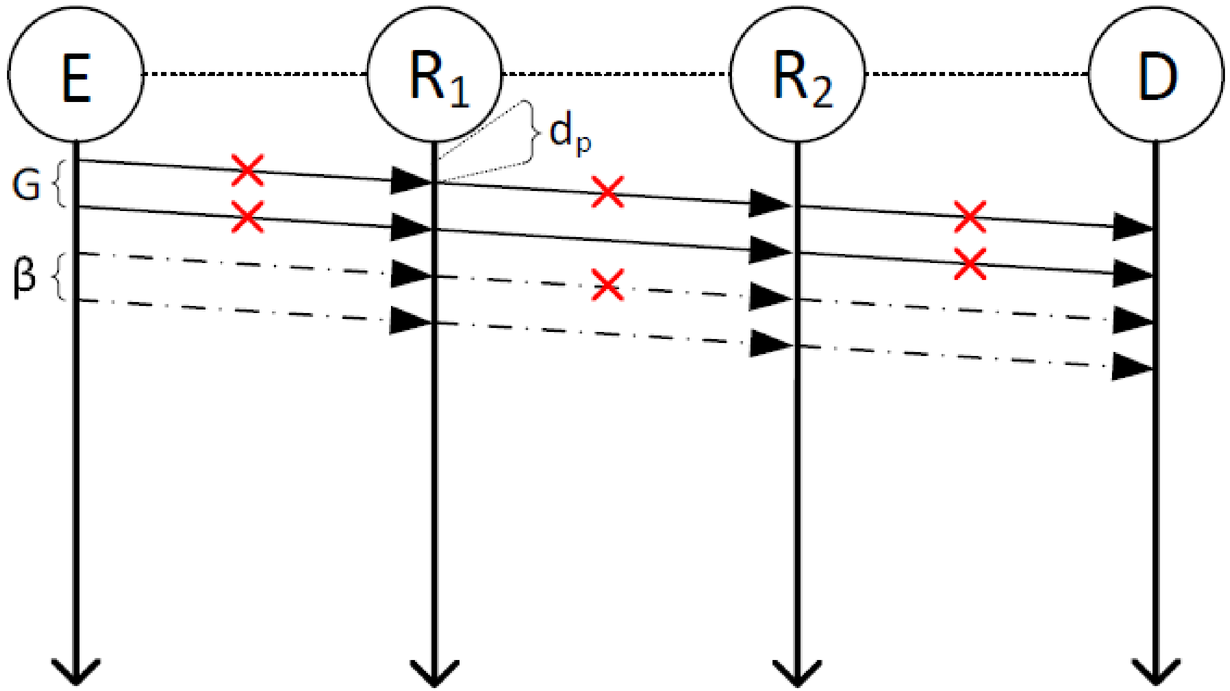
$$D_{HbH} = G \cdot \left(\frac{1}{1-\epsilon}\right) \cdot H \cdot d_p$$



Software Defined Networks

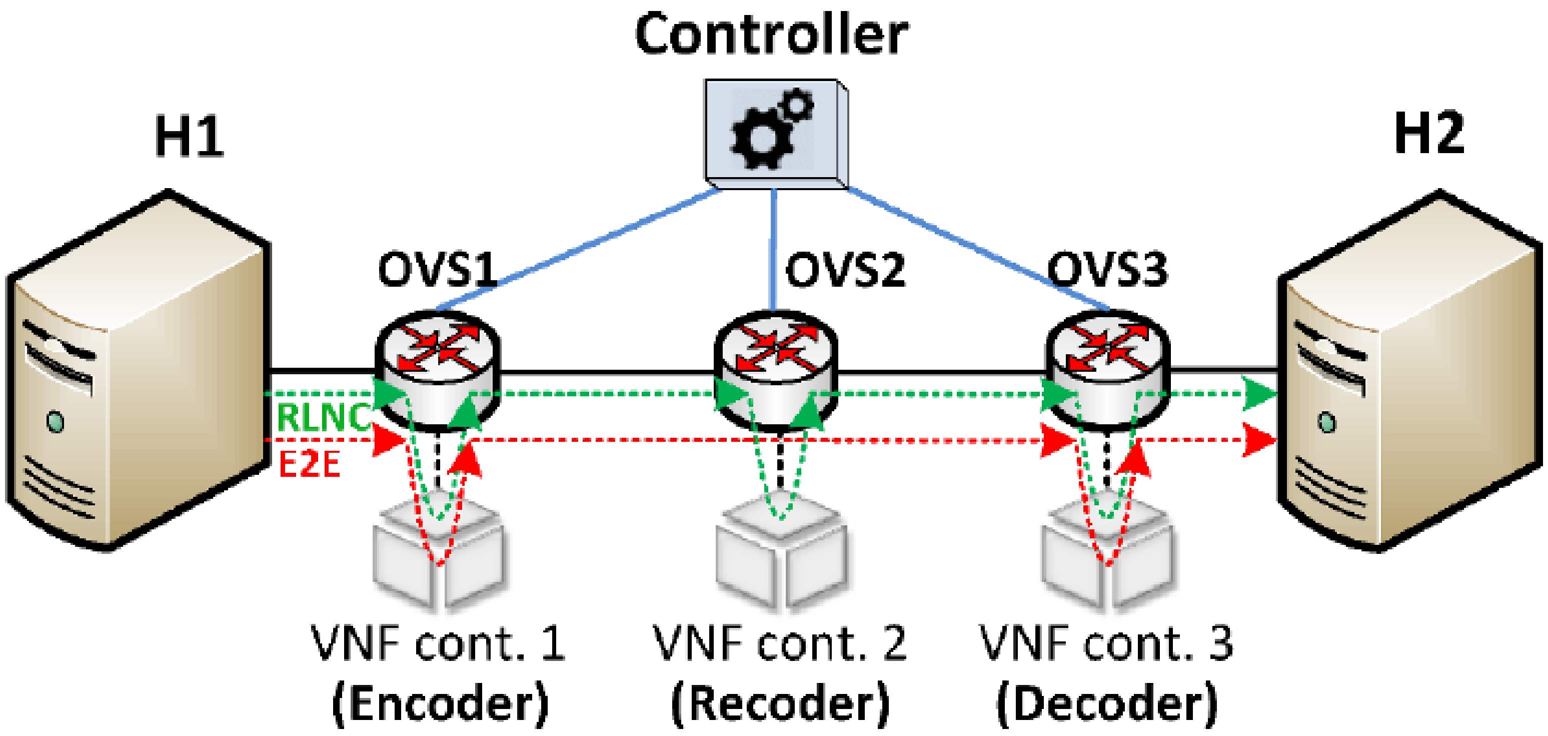
- Network Coding Scheme: Compute and Forward

$$P_{RLNC} = G \cdot H \cdot \left(\frac{1}{1-\epsilon}\right) \qquad D_{RLNC} = \left(G \cdot \left(\frac{1}{1-\epsilon}\right) + (H-1)\right) \cdot d_p$$



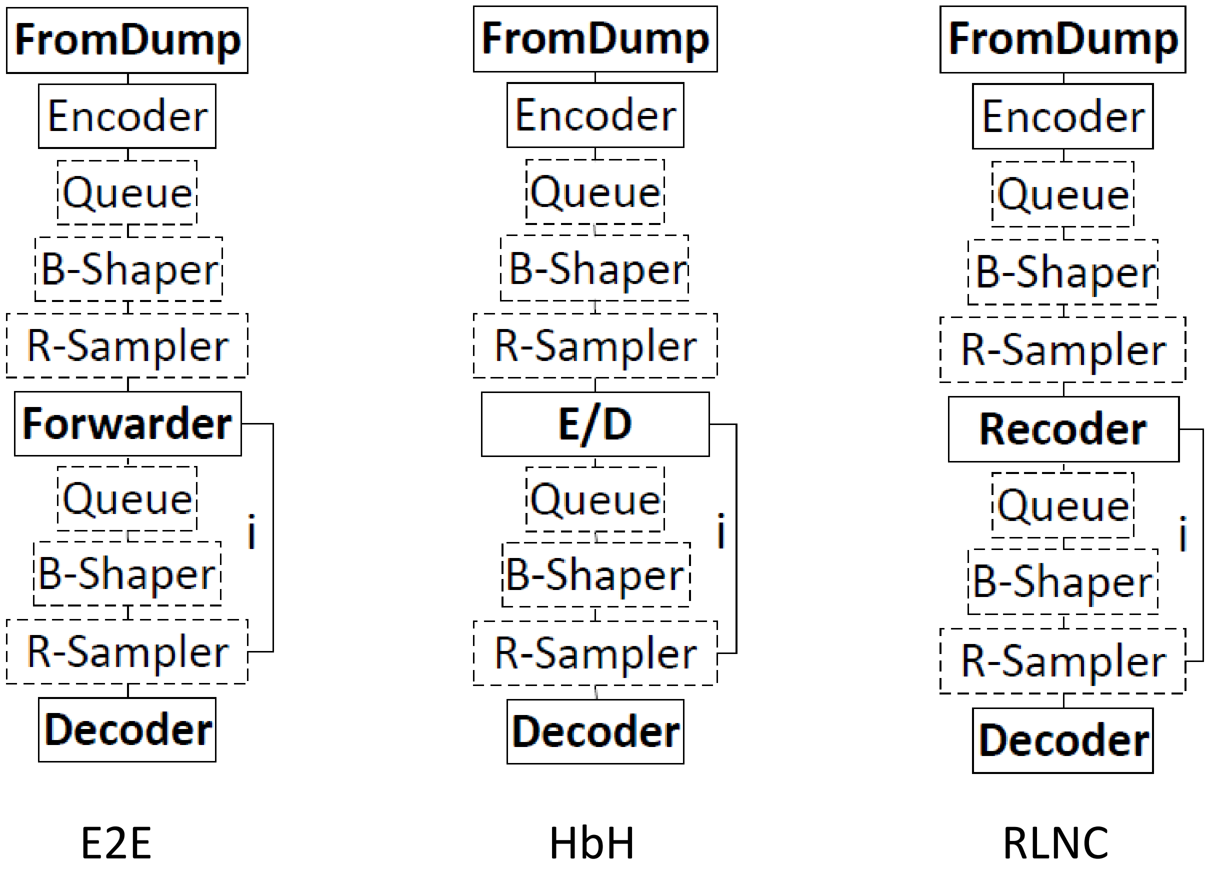
SDN testbed

- Example with ESCAPE prototyping environment



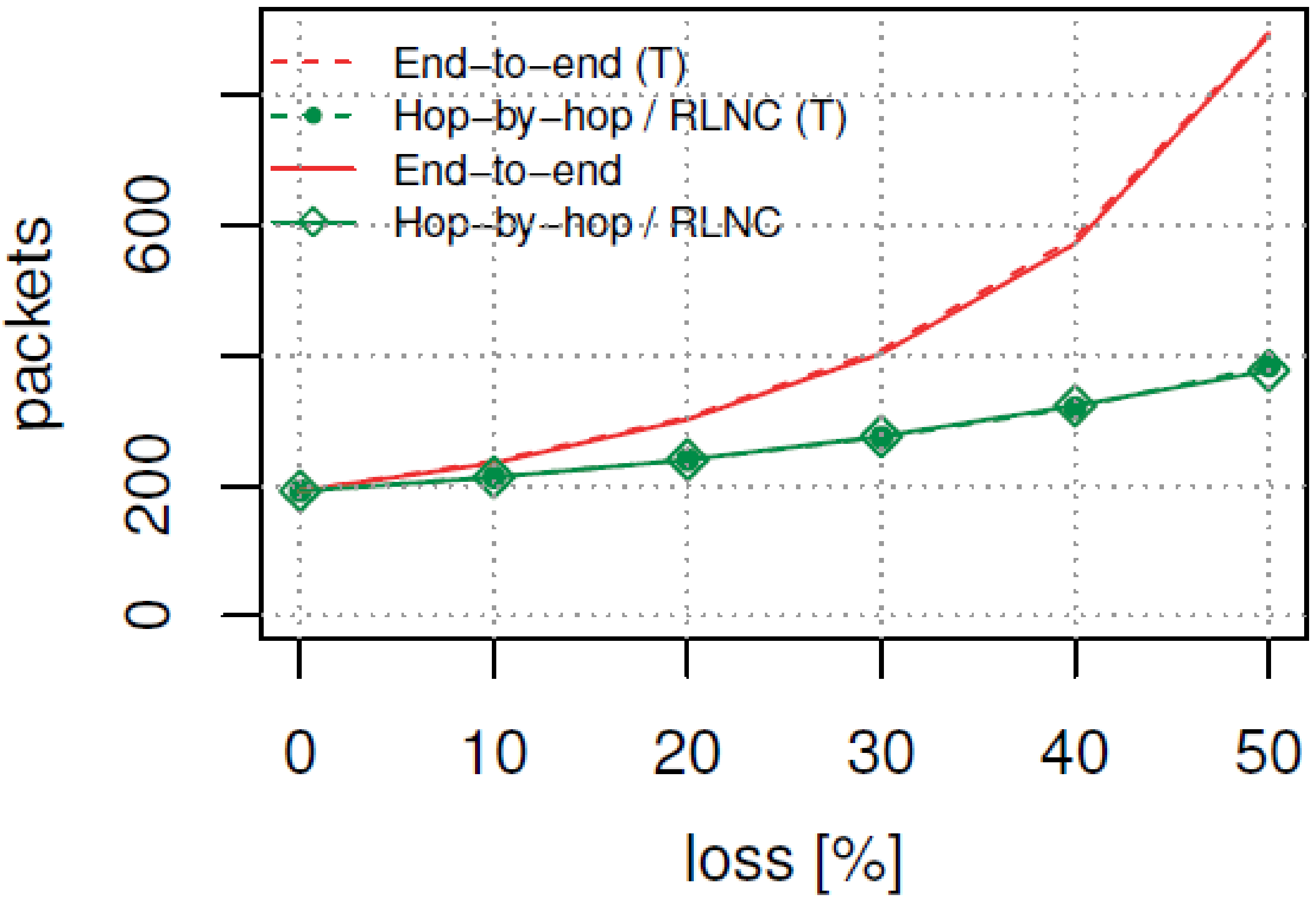
Software Defined Networks

- Click configuration



Software Defined Networks

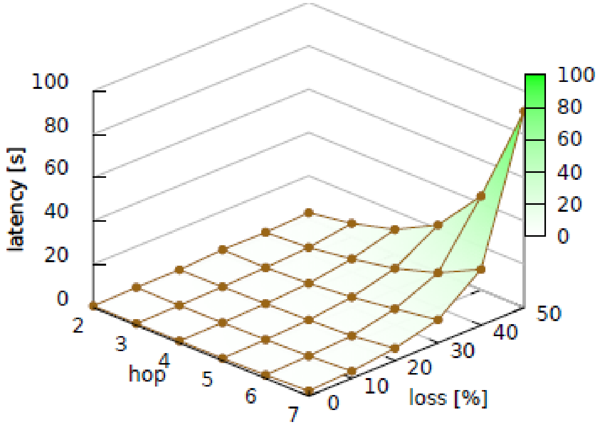
- Packets injected into network for the three approaches



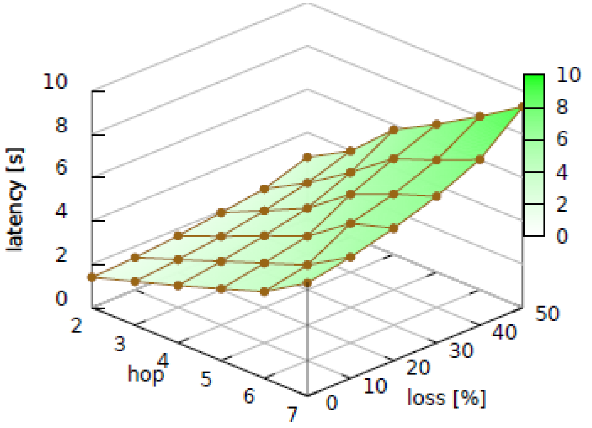
TECHNISCHE UNIVERSITÄT DRESDEN

Results

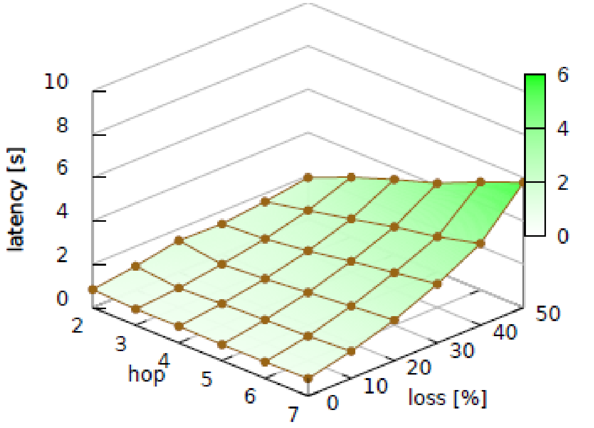
- Packets 64 – Size 250 B – Bitrate 0.25 Mb/s



end-to-end



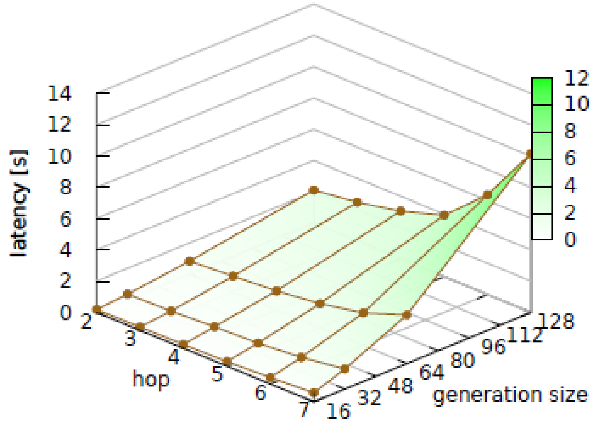
hop-by-hop



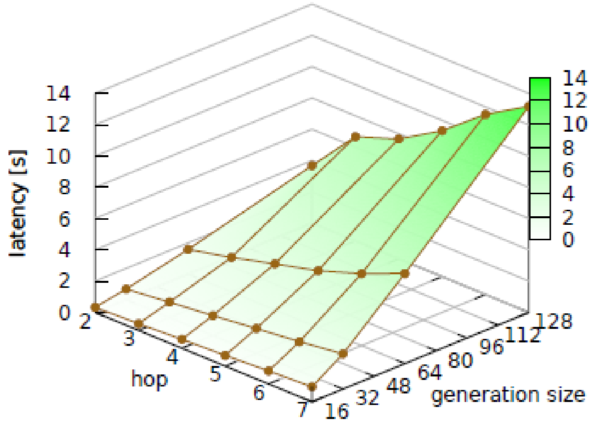
RLNC

Results

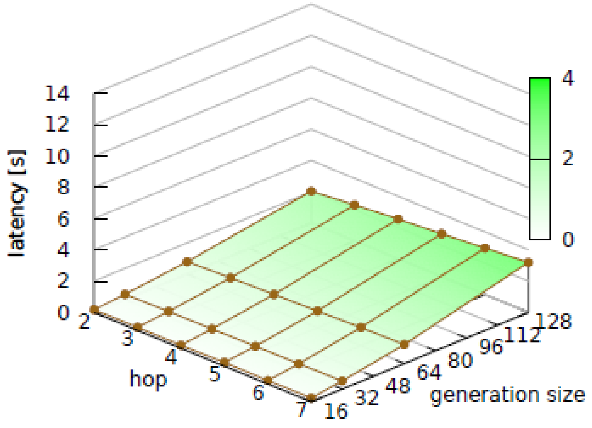
- Size 250 B – Loss 10% – Bitrate 0.25 Mb/s



end-to-end



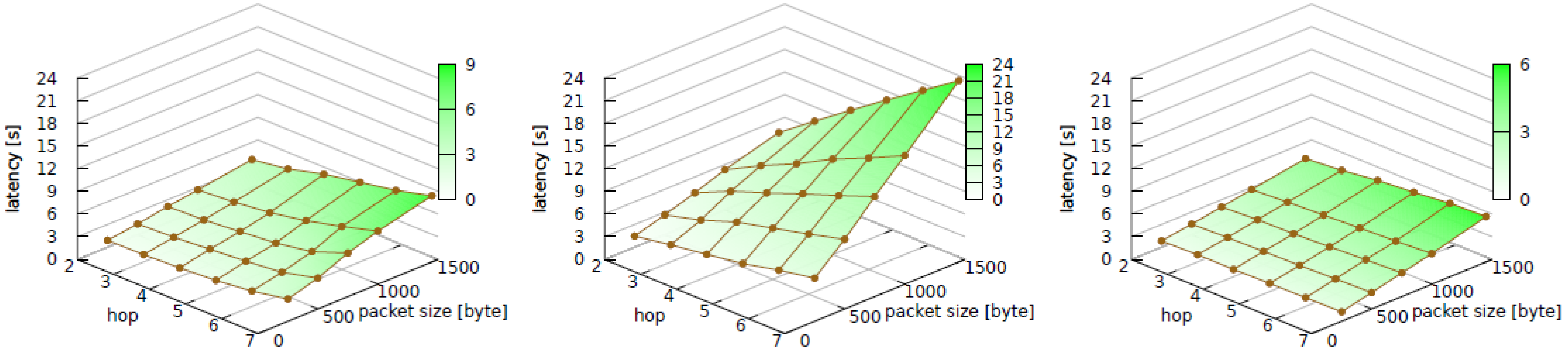
hop-by-hop



RLNC

Results

- Packets 64 – Loss 10% – Bitrate 0.25 Mb/s



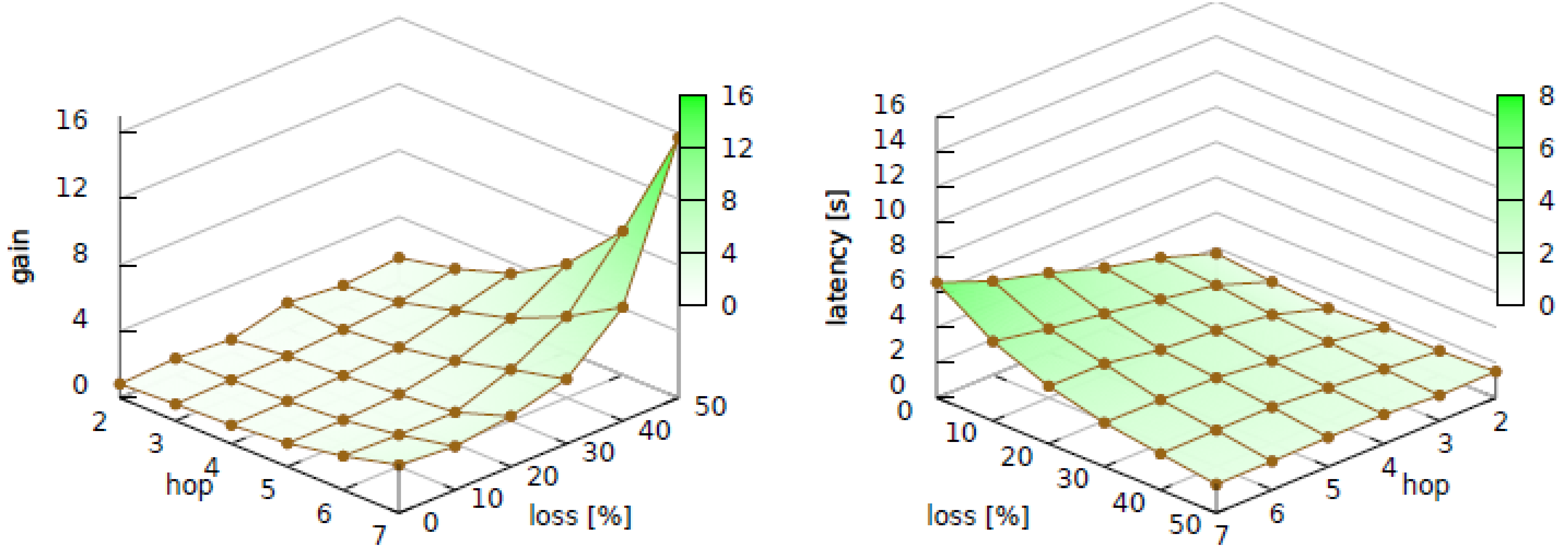
end-to-end

hop-by-hop

RLNC

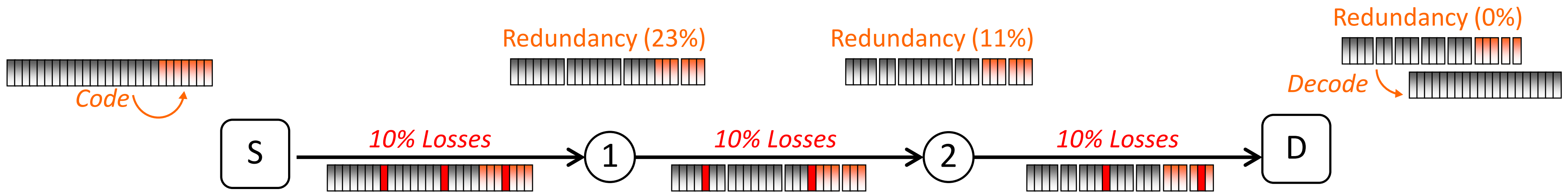
Software Defined Networks

- Latency gain of e2e vs RLNC (left) and hbh vs RLNC(right)

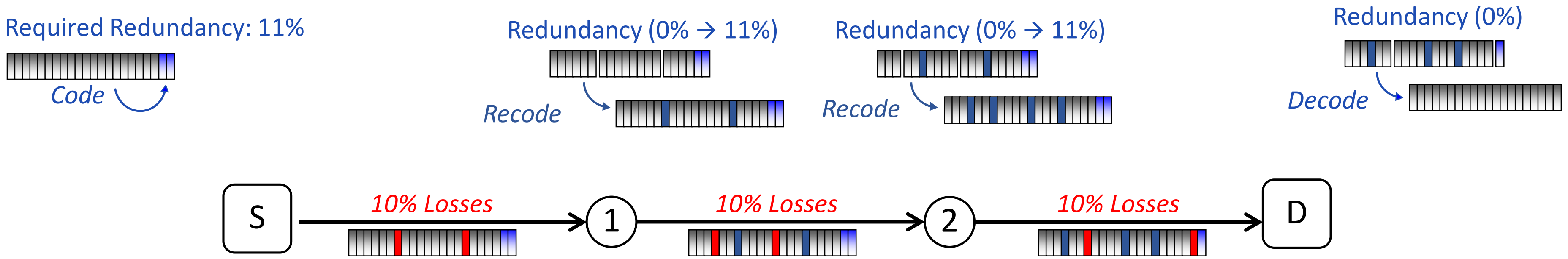


The Recoding Advantage

Coding End-to-End Overhead = Cumulative Losses (37%)

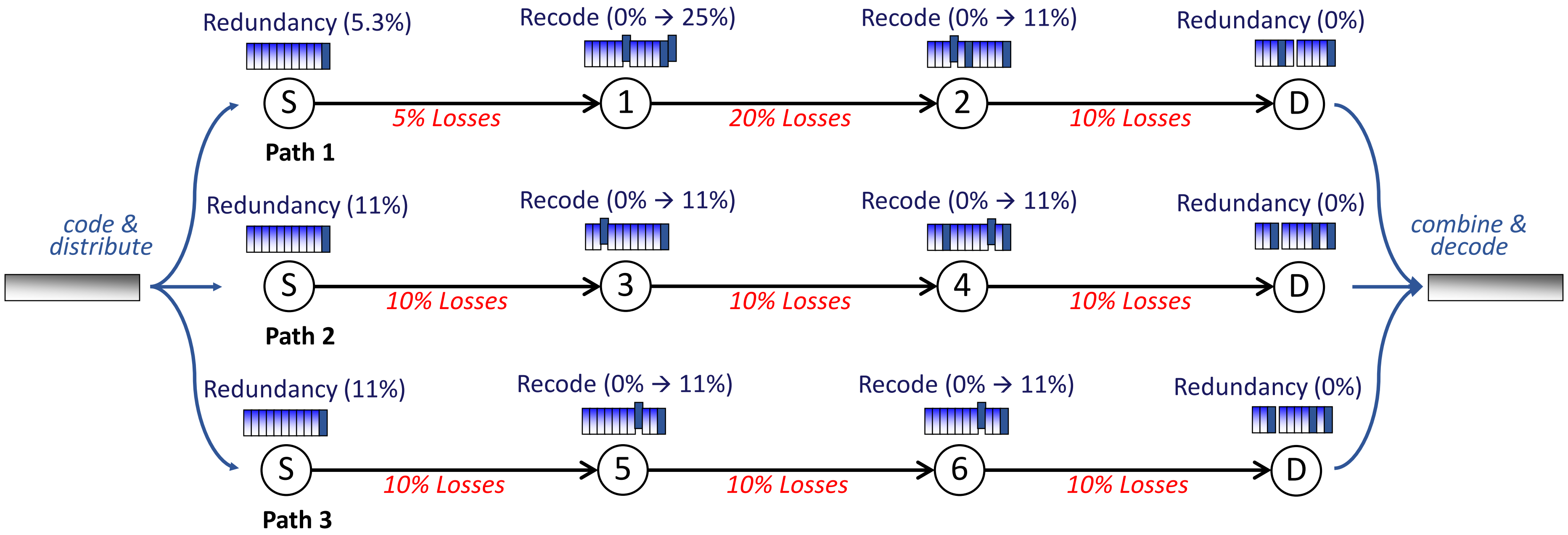


Re-Coding Overhead = Single Worst-Case Loss (11%)



- Optimal and Dynamic Loss Compensation

Recoding + Multipath Advantage

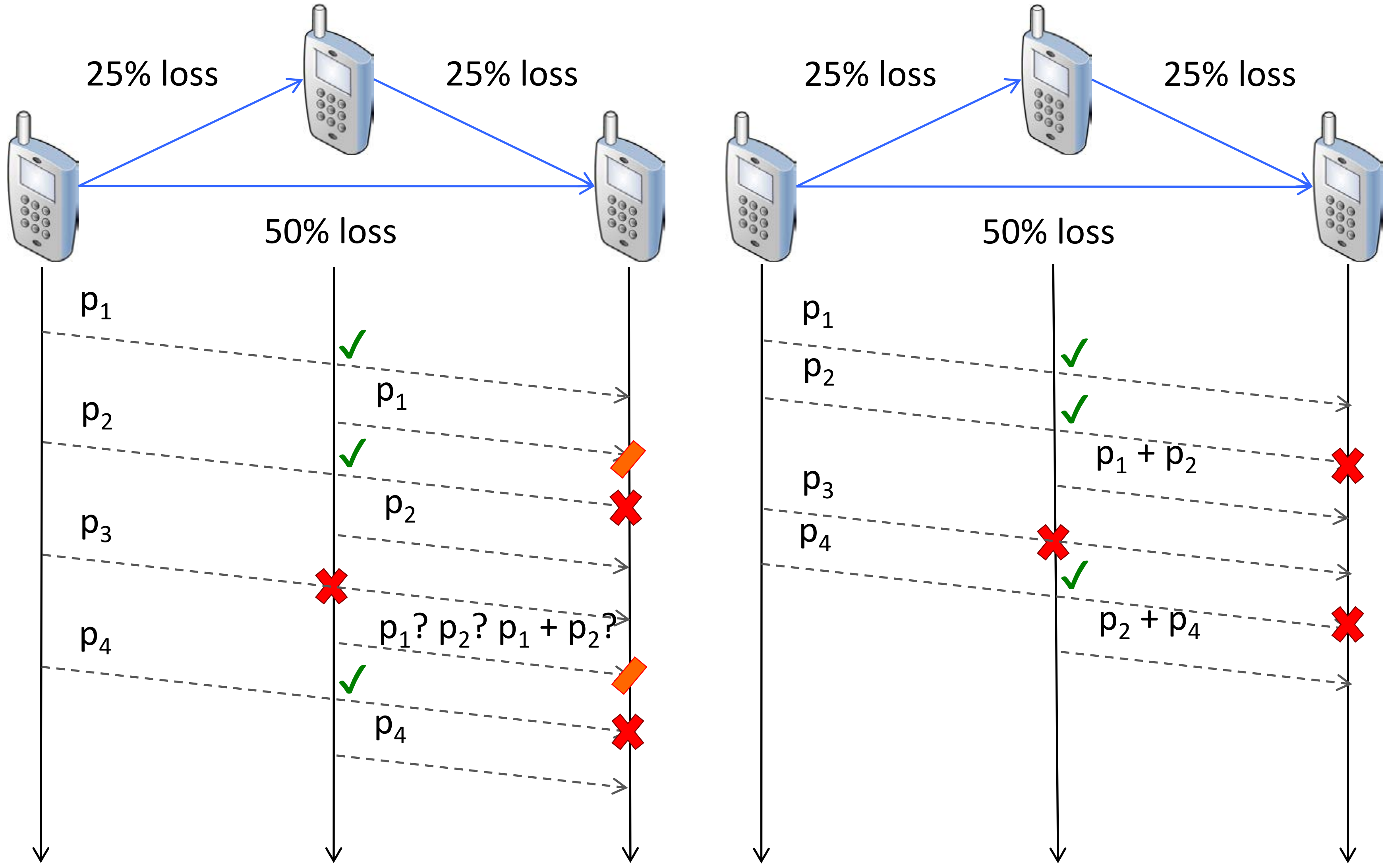


Summing without scheduling

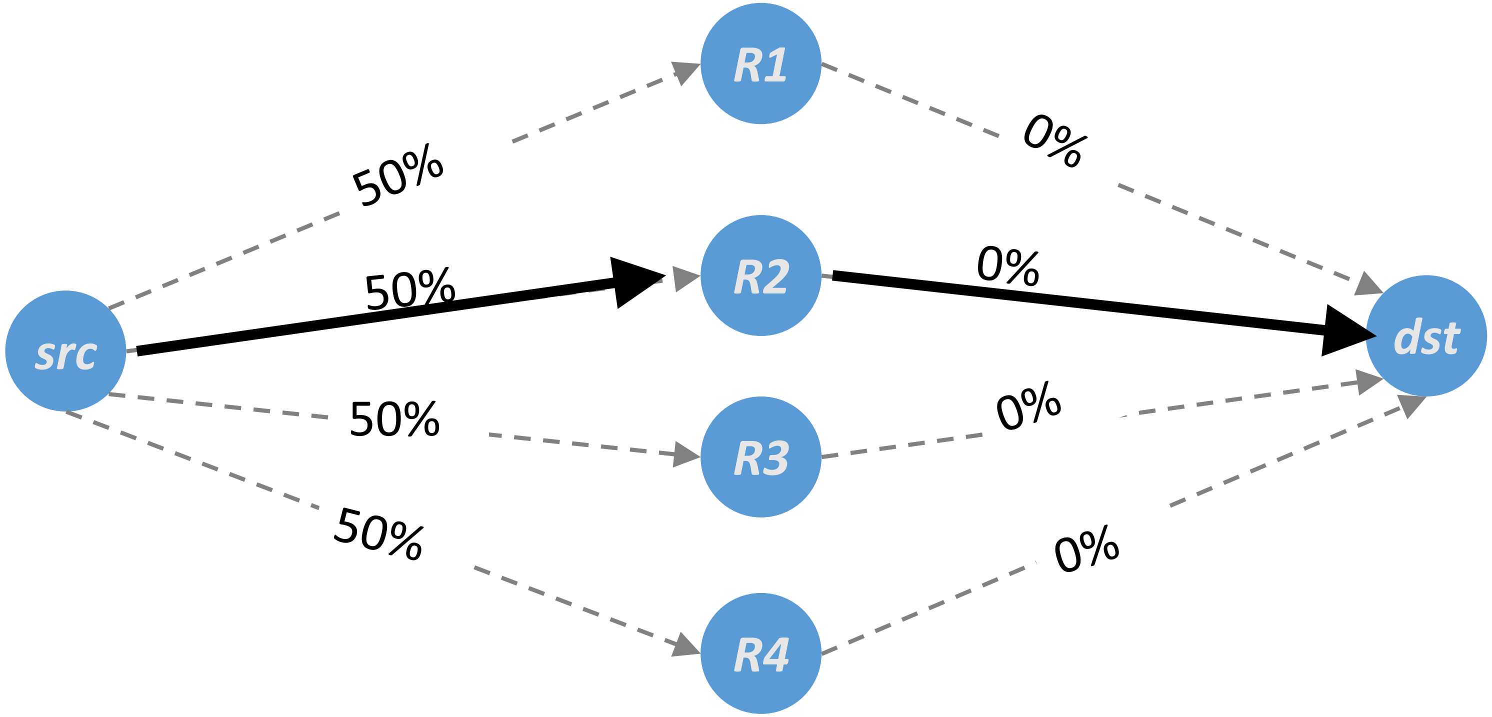
- Total **Throughput** = $\sum_{multipath}$ (path rate – worst link)
- Inject 10Mpbs at each path → Obtain 26Mbps seamlessly

▪ Native Bandwidth Aggregation

Other Recoding Issues

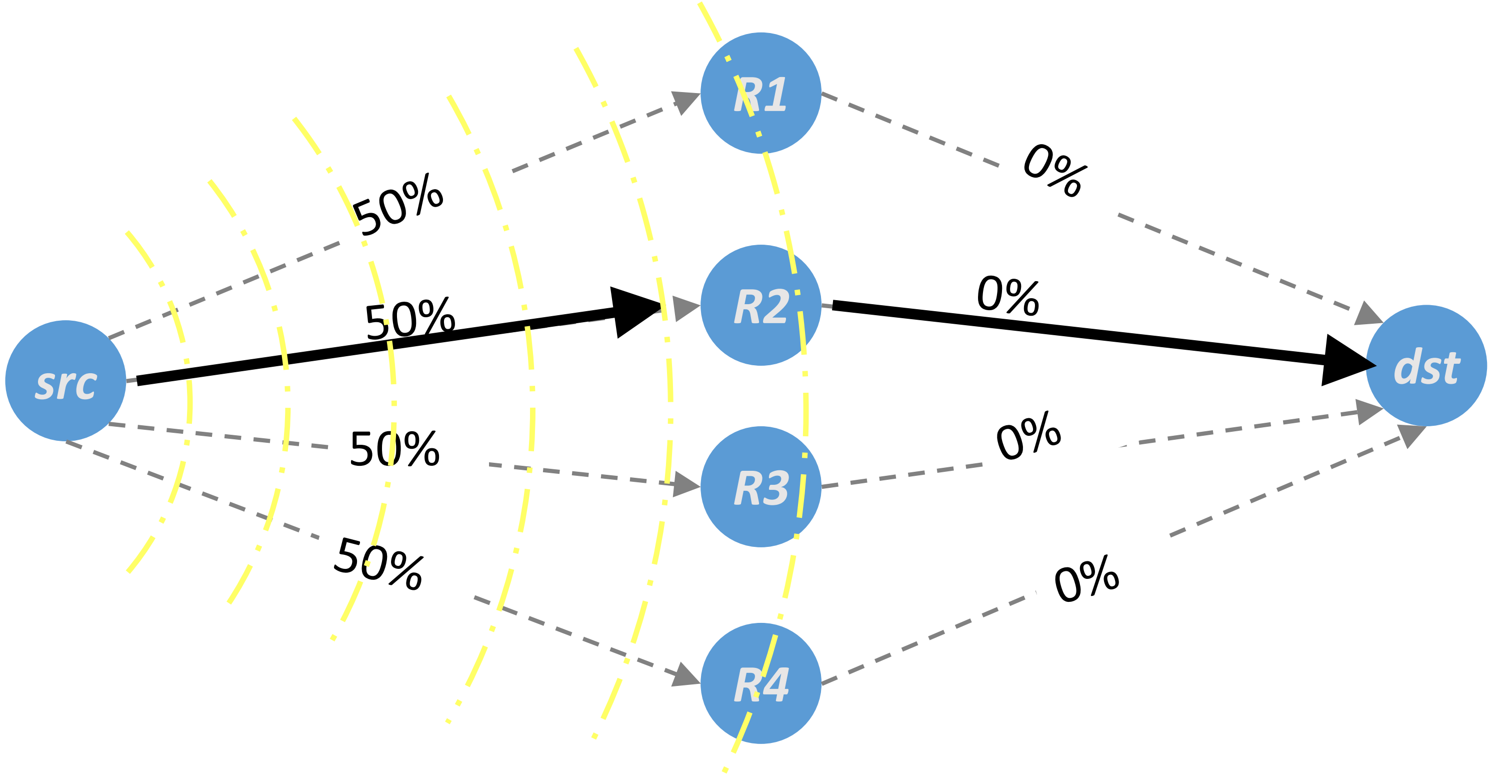


Sender Is in a Bad Spot



- Best single path → Prob. of loss 50%

Sender Is in a Bad Spot



- Best single path → Prob. of loss 50%

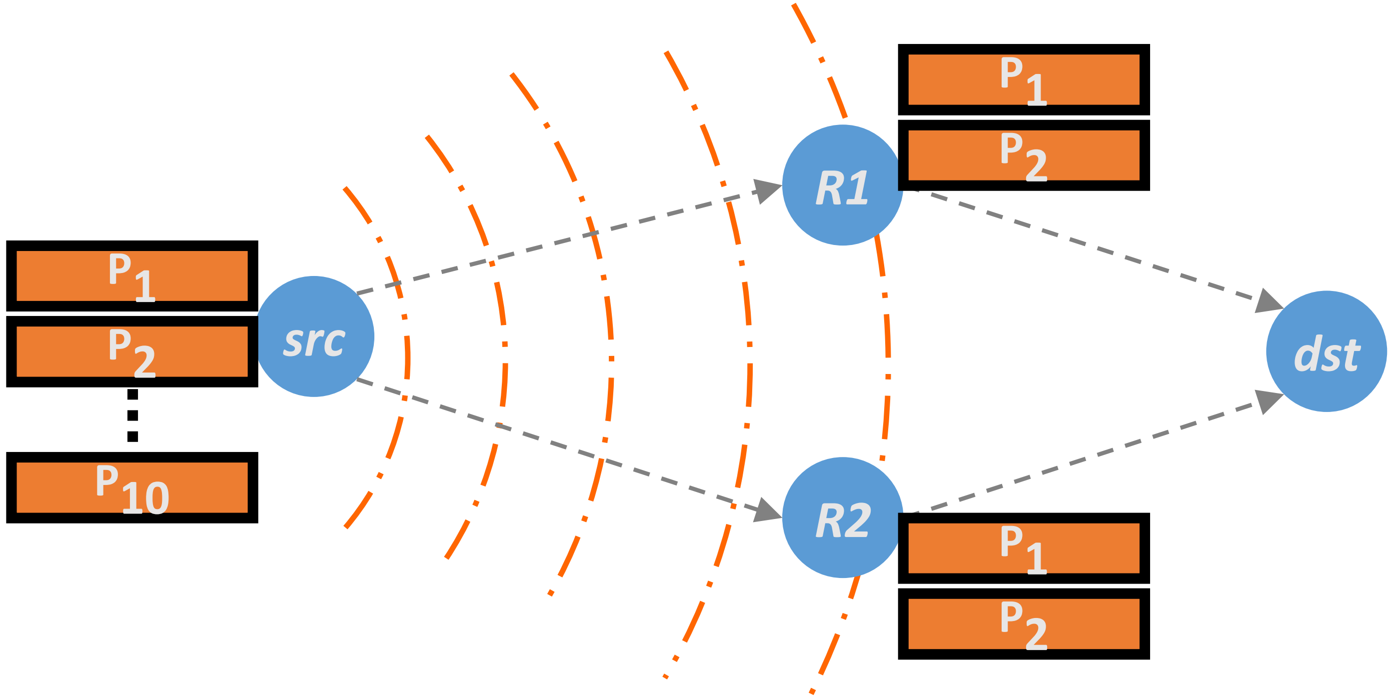
- Spatial diversity to the rescue
  - Any router forward packet → Prob. of loss 0.54 = 6%

Challenge with Using Spatial Diversity

- Overlap in received packets → Routers forward duplicates

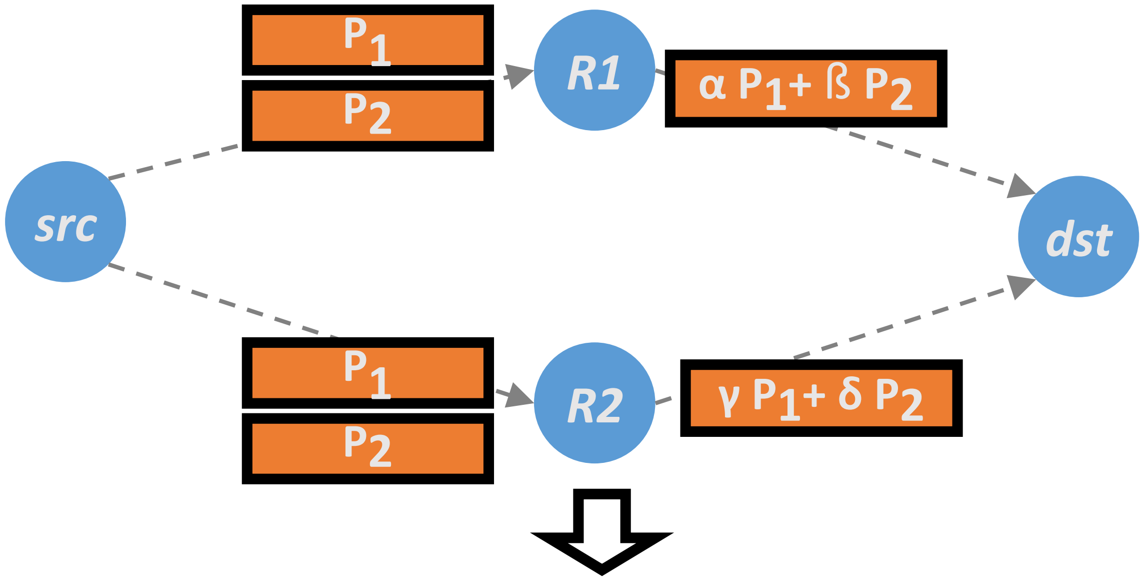
Challenge with Using Spatial Diversity

- Overlap in received packets → Routers forward duplicates



Random Linear Network Coding

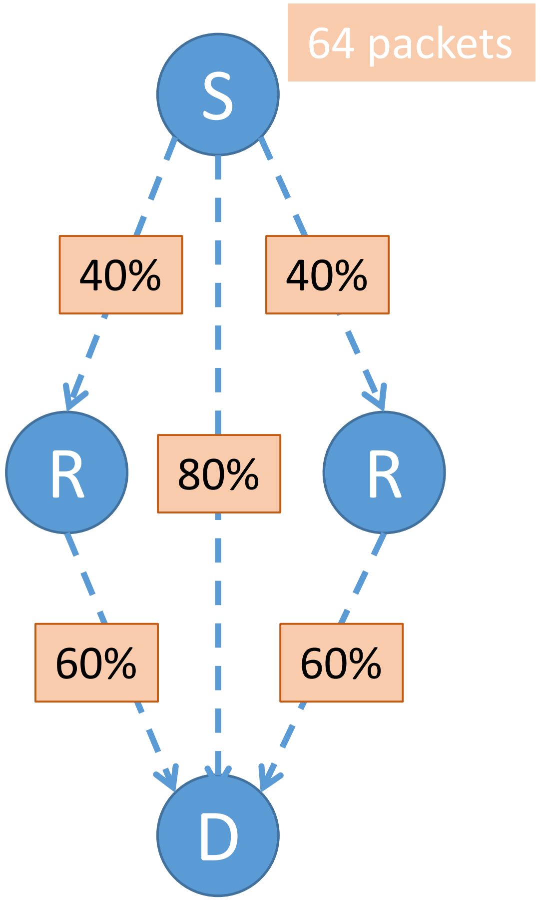
- Each router forwards random combinations of packets

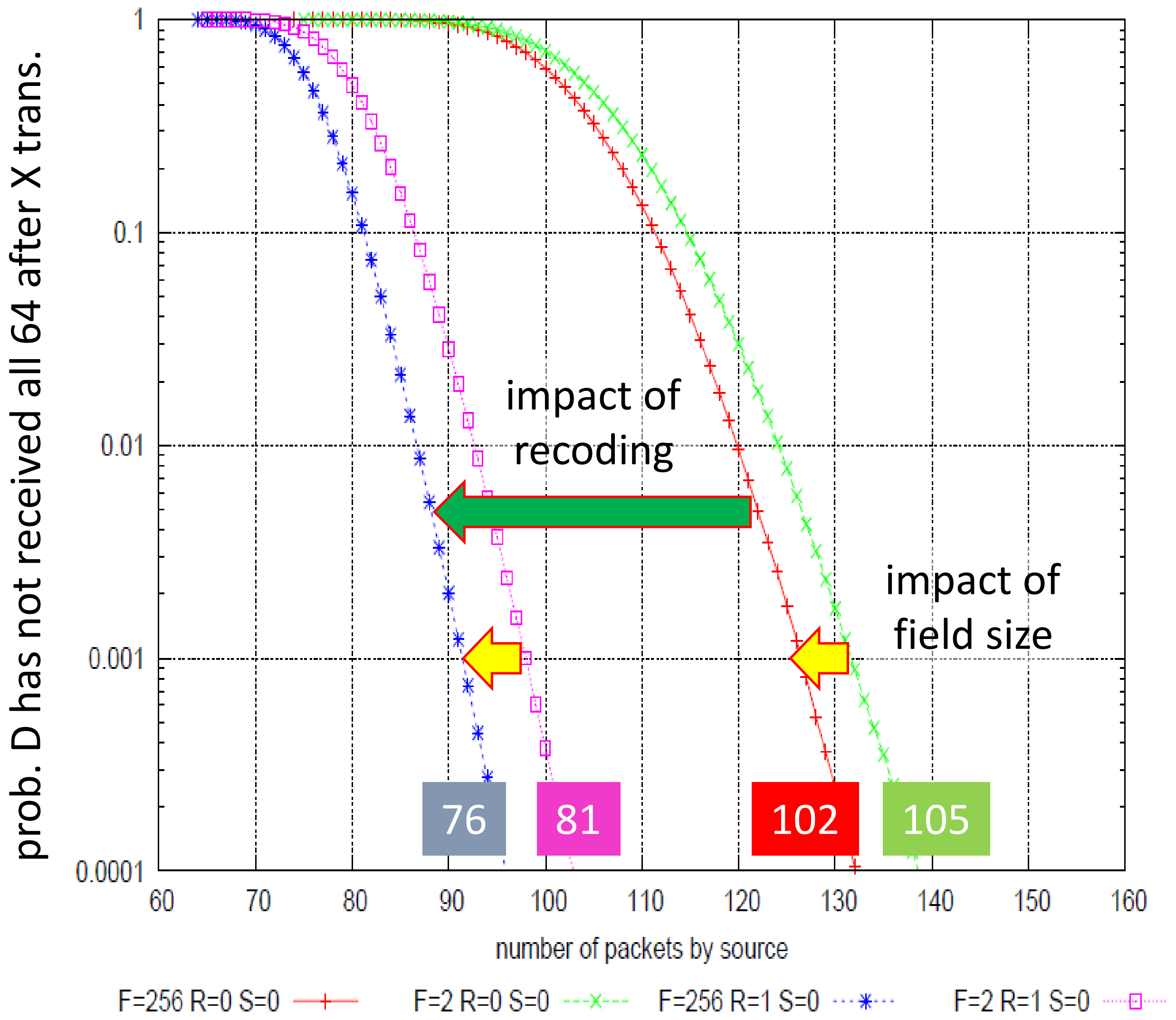


Randomness prevents duplicates

Network coding exploits spatial diversity to improve dead spots

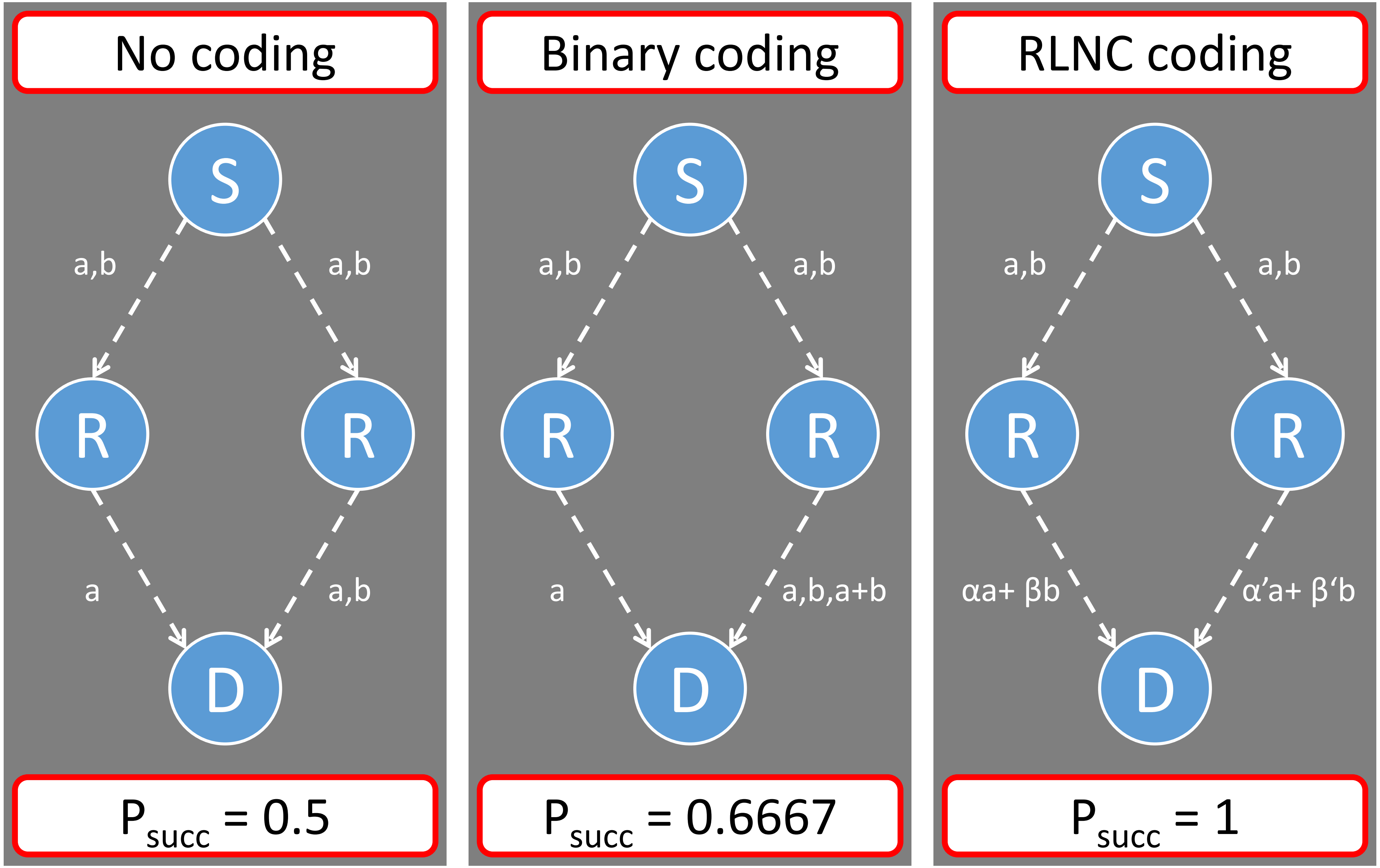
Impact of Recoding



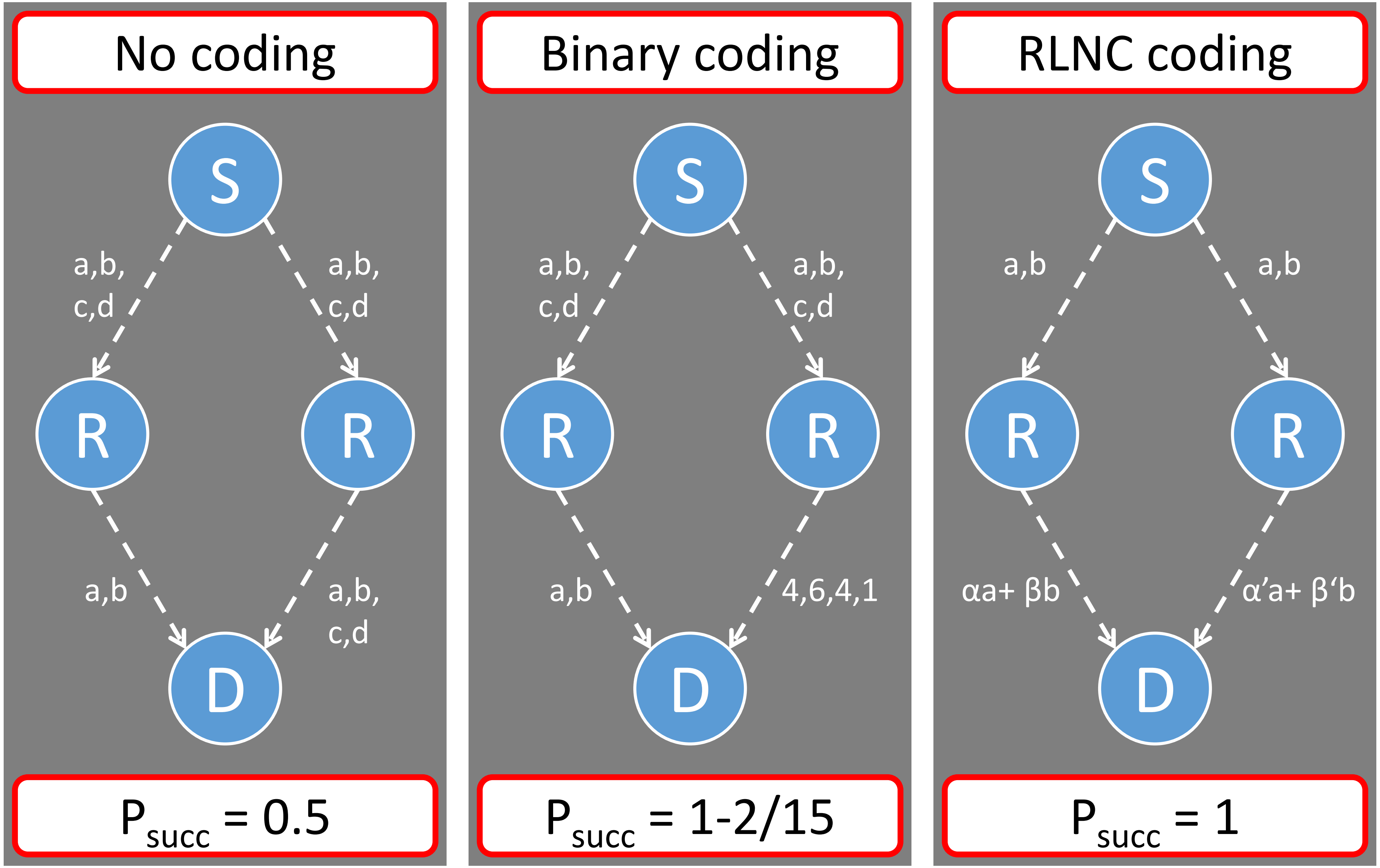


No need for signalling!

Coding


No coding
S
a,b
a,b
R
R
a
a,b
D
P succ = 0.5
Binary coding
S
a,b
a,b
R
R
a
a,b,a+b
D
P succ = 0.6667
RLNC coding
S
a,b
a,b
R
R
αa+ βb
α'a+ β'b
D
P succ = 1

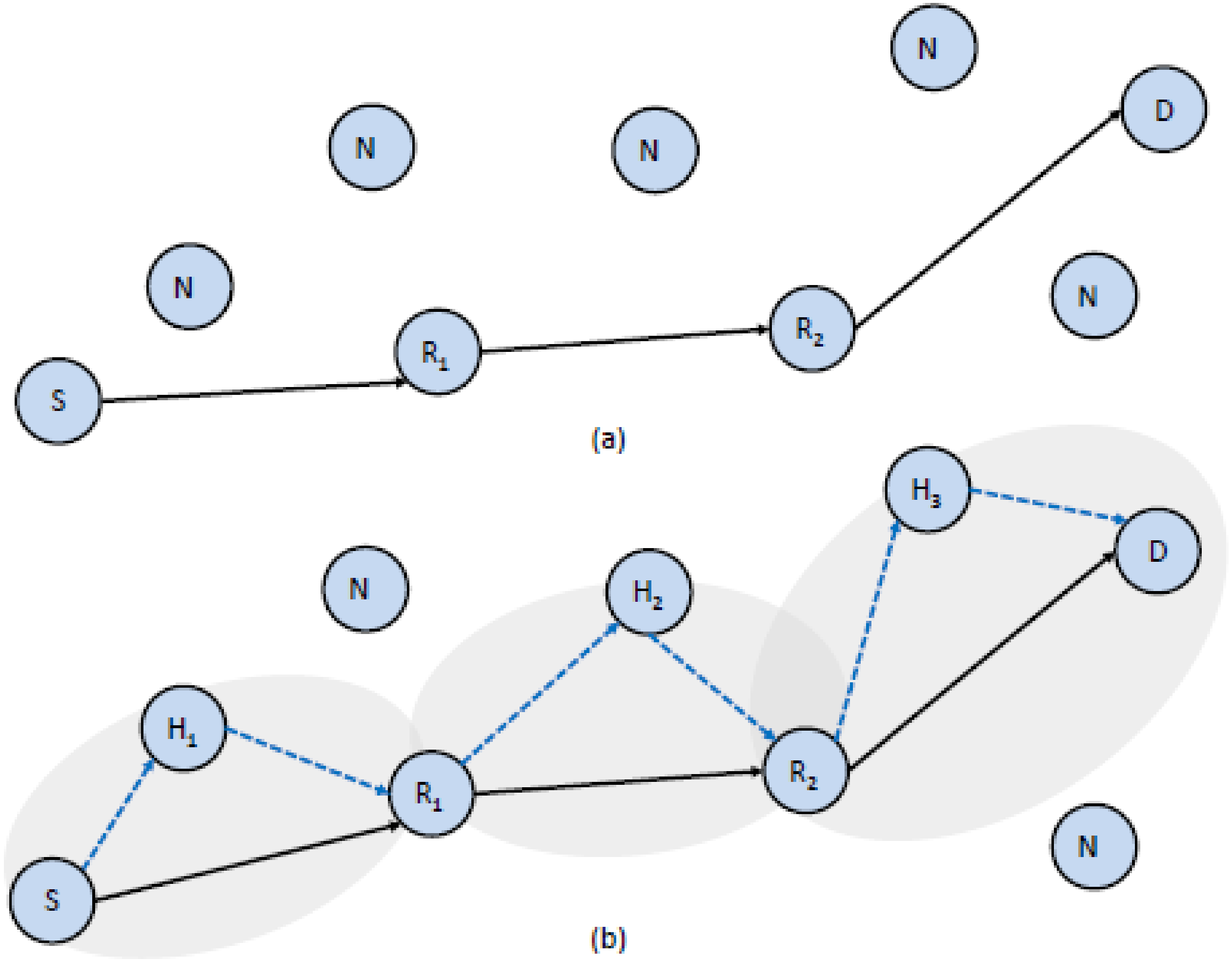
Coding


No coding
S
a,b,
c,d
a,b,
c,d
R
R
a,b
a,b,
c,d
D
P succ = 0.5
Binary coding
S
a,b,
c,d
a,b,
c,d
R
R
a,b
4,6,4,1
D
P succ = 1-2/15
RLNC coding
S
a,b
a,b
R
R
αa+ βb
α’a+ β‘b
D
P succ = 1

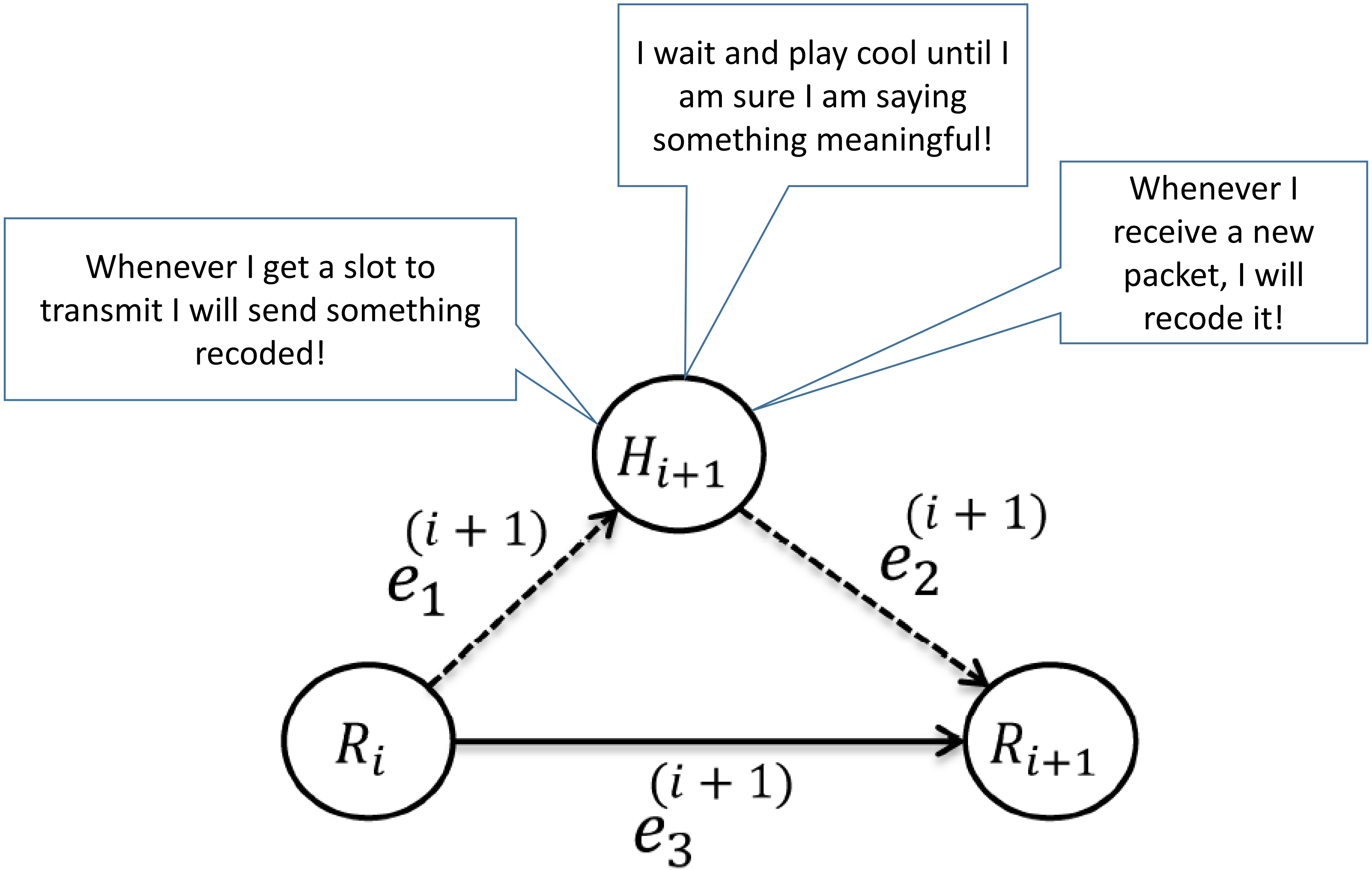
TECHNISCHE UNIVERSITÄT DRESDEN

Impact of the protocol design

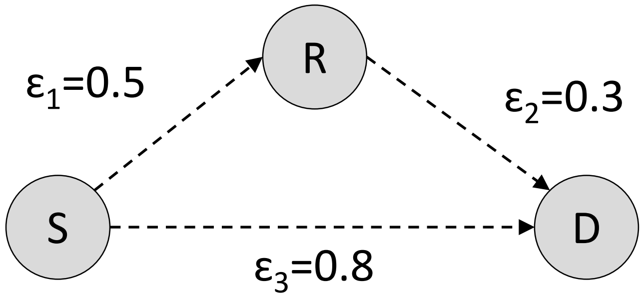
RLNC in real meshed



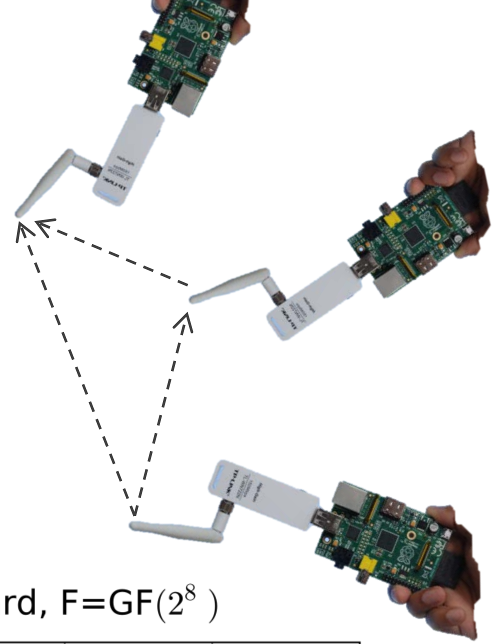
Strategies under Investigation

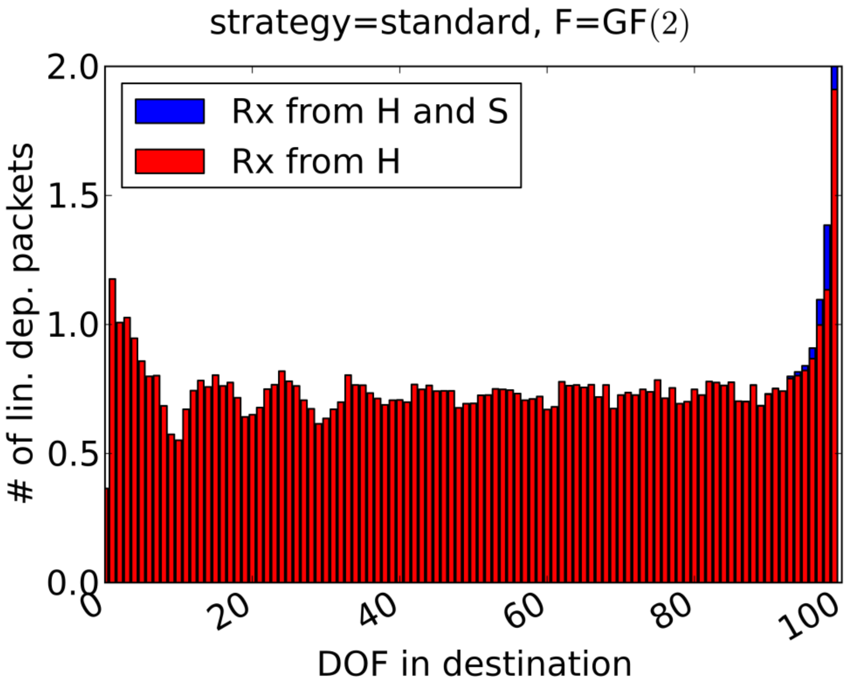


Recode

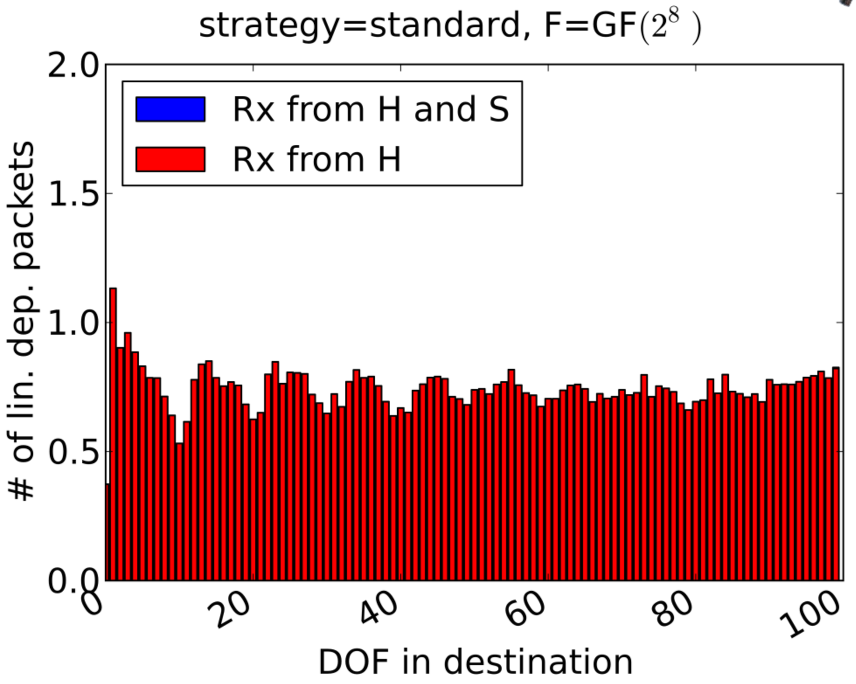


R
S
D
ε1=0.5
ε2=0.3
ε3=0.8





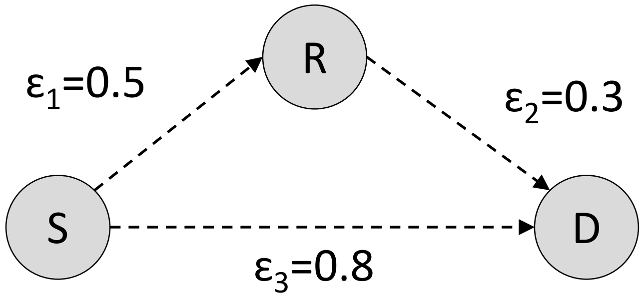
strategy=standard, F=GF(2)
Rx from H and S
Rx from H
# of lin. dep. packets
DOF in destination
0.0
0.5
1.0
1.5
2.0
0
20
40
60
80
100

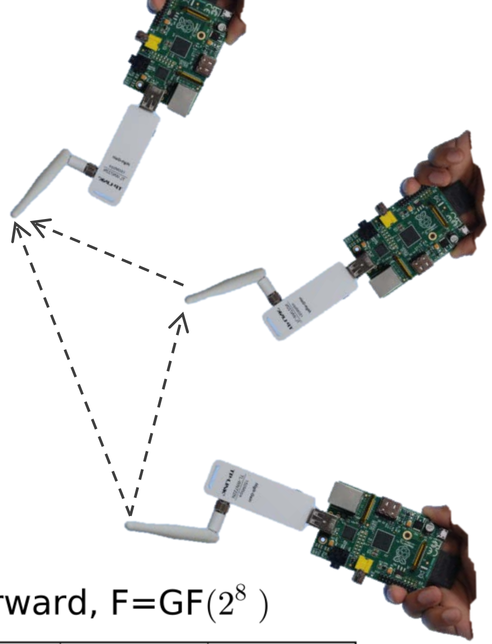


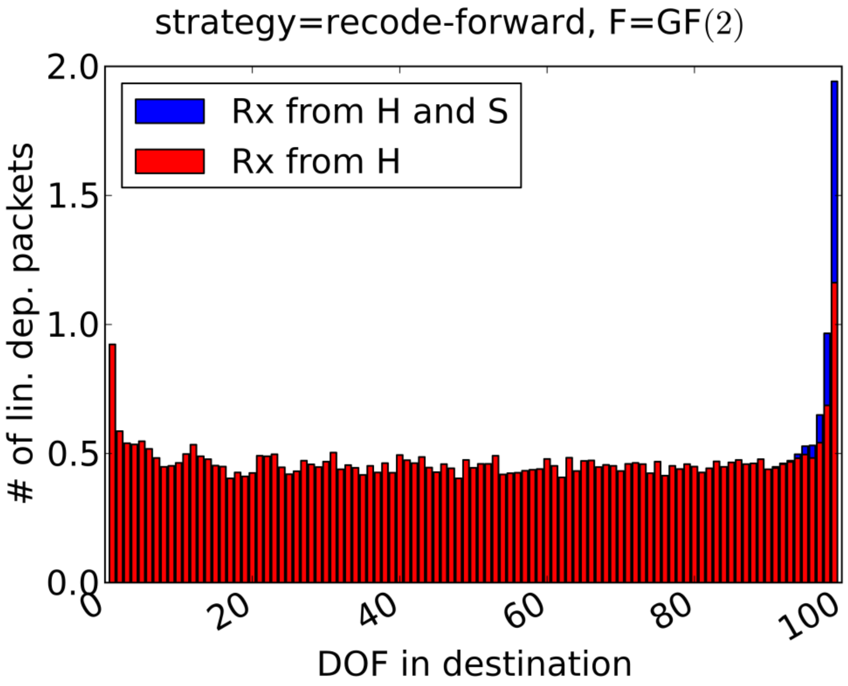
strategy=standard, F=GF(2^8)
Rx from H and S
Rx from H
# of lin. dep. packets
DOF in destination
0.0
0.5
1.0
1.5
2.0
0
20
40
60
80
100

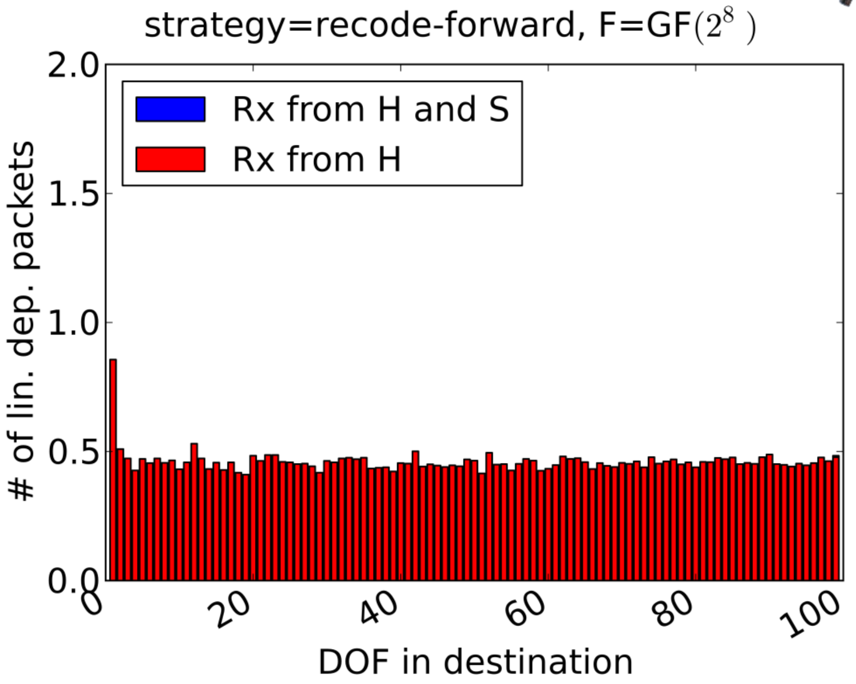
TECHNISCHE UNIVERSITÄT DRESDEN

Recode on new arrival

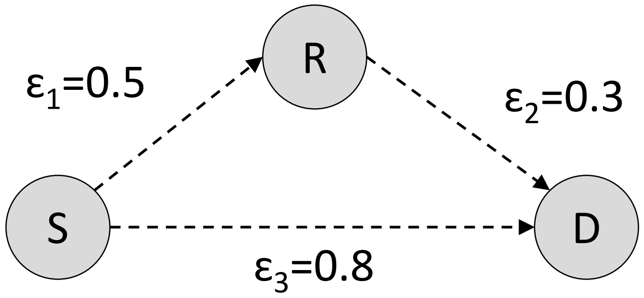




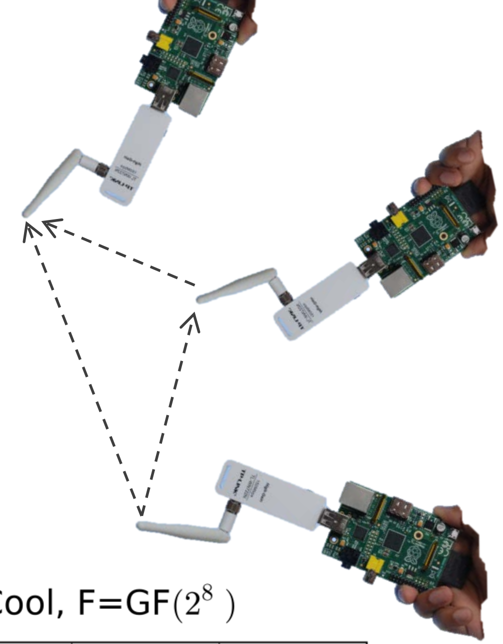


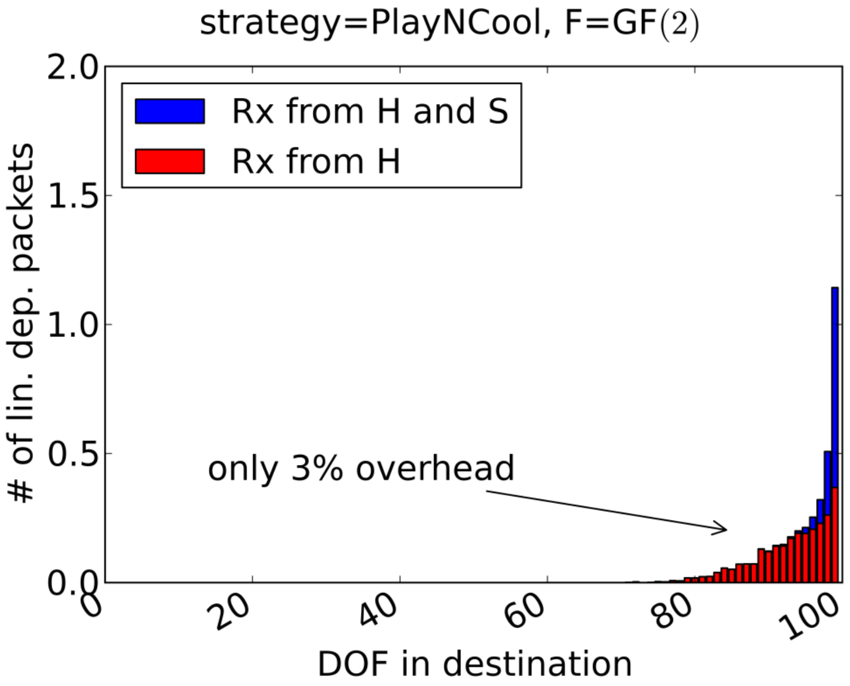


PlayNCool

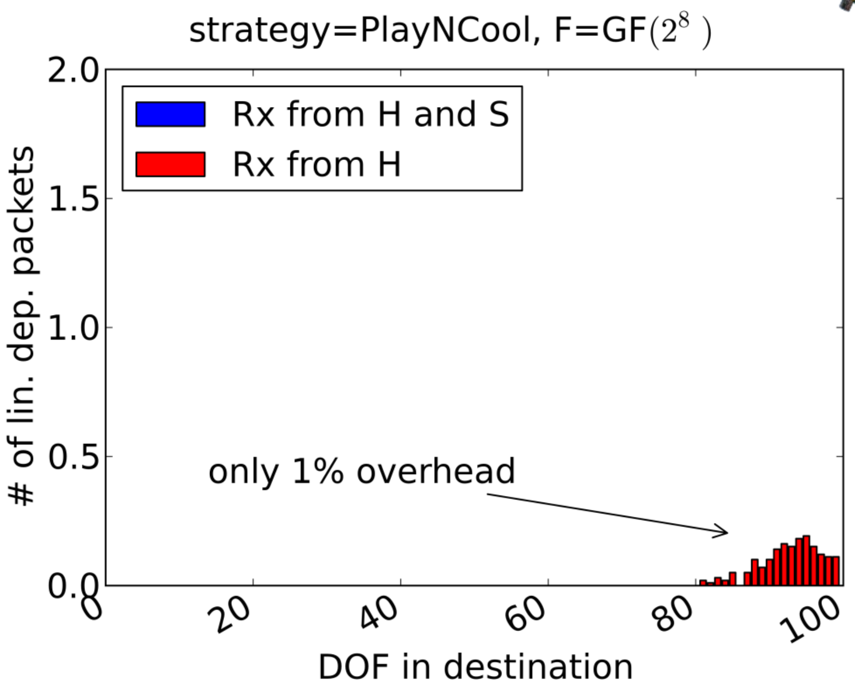


R
S
D
ε1=0.5
ε2=0.3
ε3=0.8



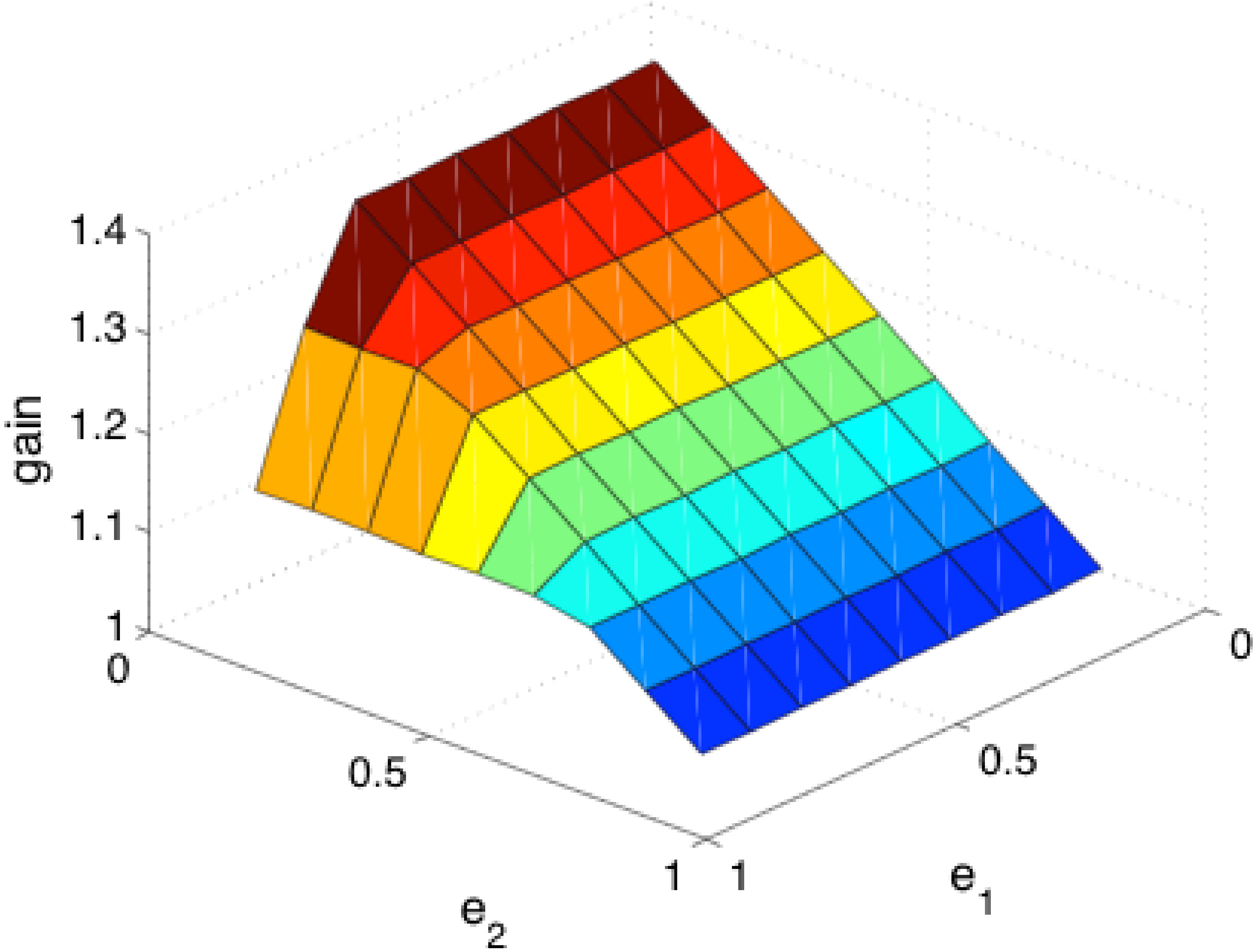


strategy=PlayNCool, F=GF(2)
Rx from H and S
Rx from H
# of lin. dep. packets
DOF in destination
only 3% overhead
0.0
0.5
1.0
1.5
2.0
0
20
40
60
80
100



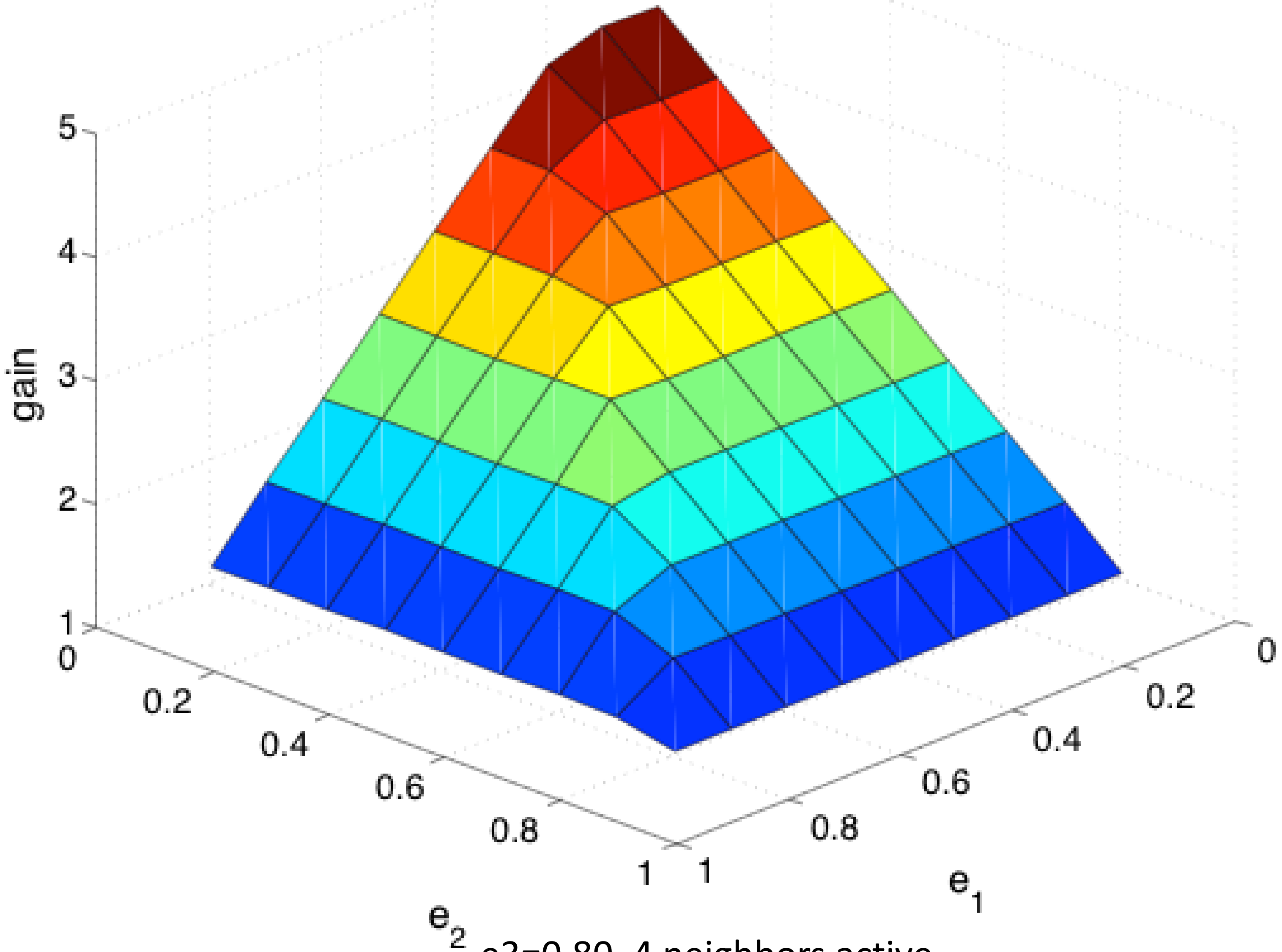
strategy=PlayNCool, F=GF(2^8)
Rx from H and S
Rx from H
# of lin. dep. packets
DOF in destination
only 1% overhead
0.0
0.5
1.0
1.5
2.0
0
20
40
60
80
100

Moderate Losses in the Direct Link



e3=0.30, 4 neighbors active

High Losses in the Direct Link



e3=0.80, 4 neighbors active