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)

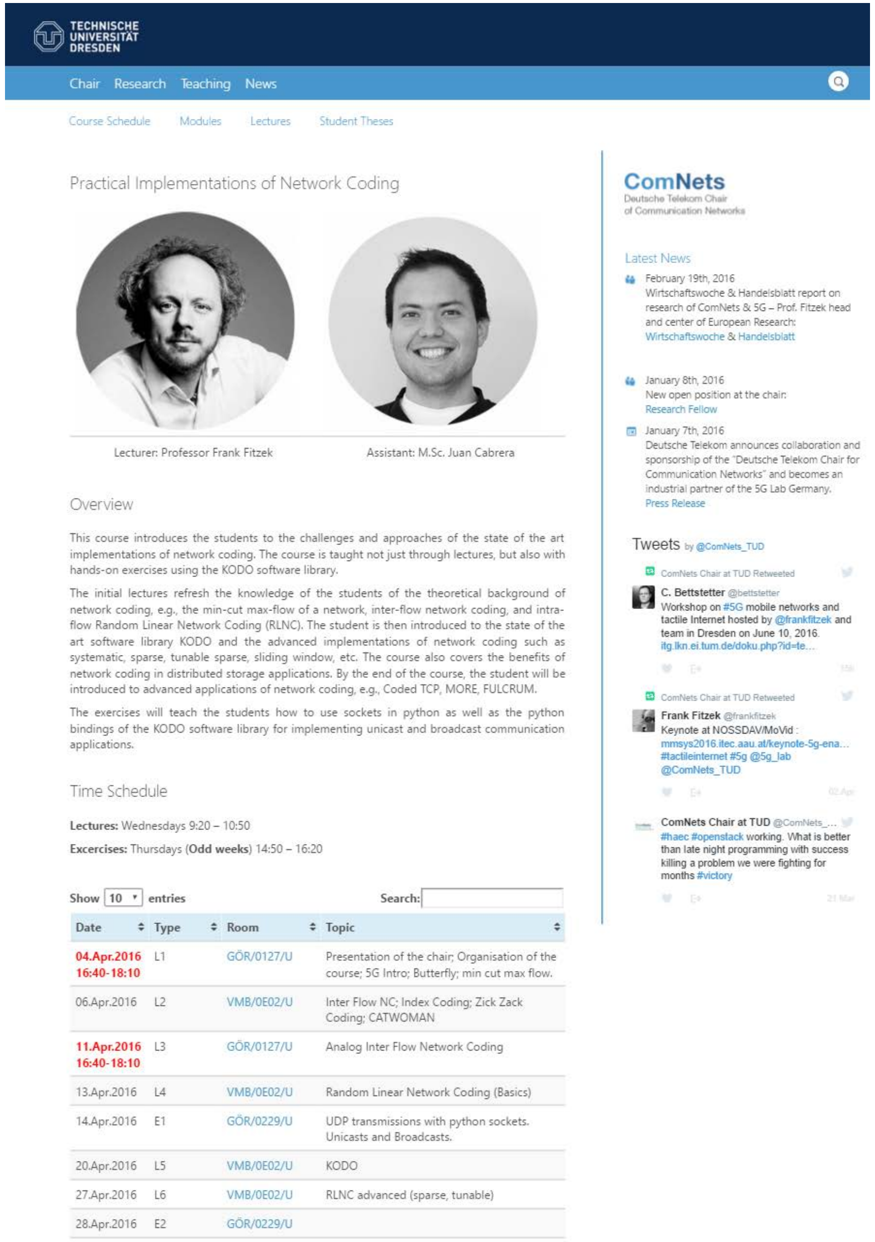
Netzwerkkodierungstheorie

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

Exercises: 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!

TECHNISCHE UNIVERSITÄT DRESDEN

A Practical Guide to RLNC Libraries

TECHNISCHE UNIVERSITÄT DRESDEN

Systematic RLNC

D Uncoded packets

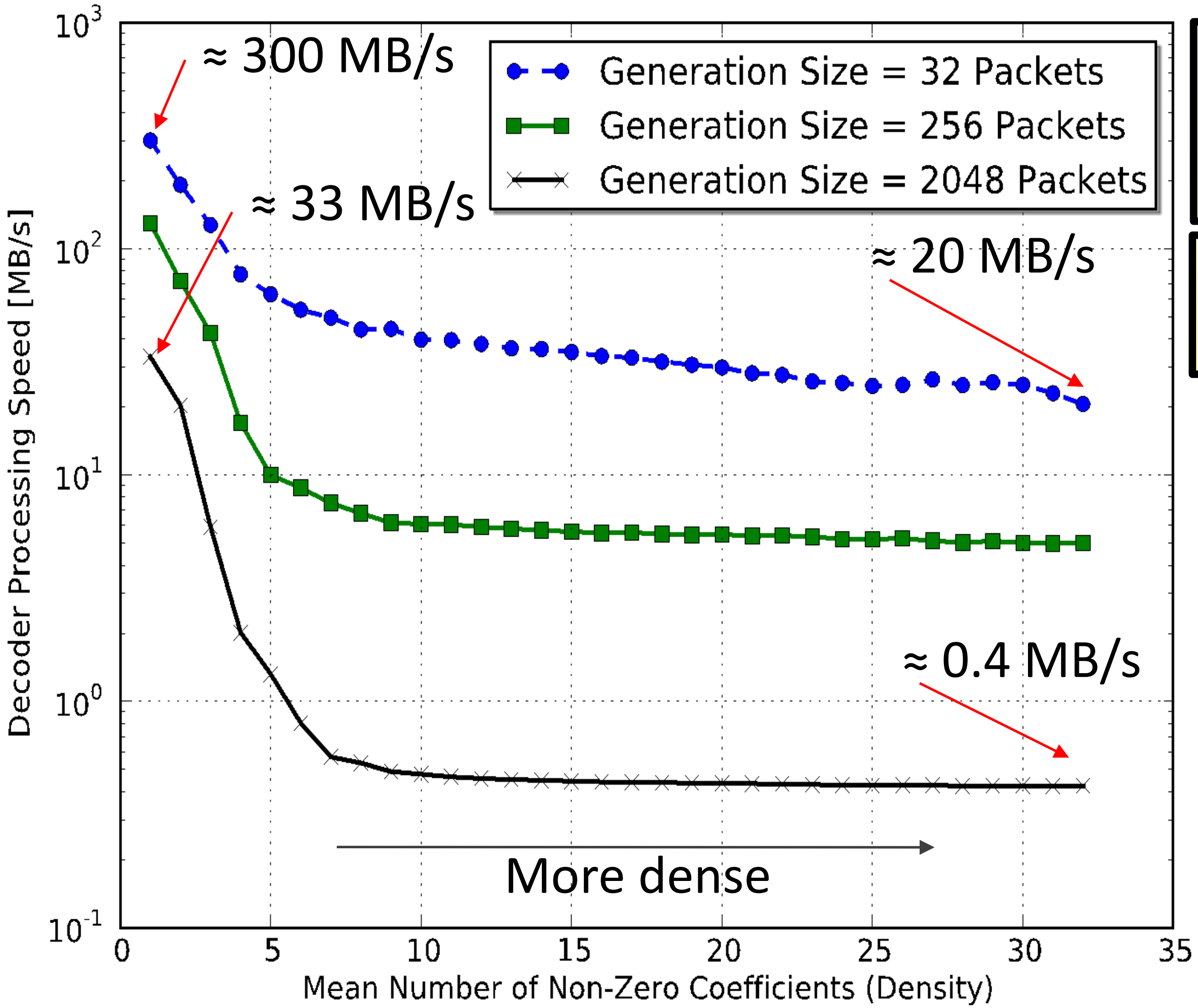
$$\begin{pmatrix} CP_1 \\ CP_2 \\ CP_3 \\ CP_4 \\ CP_5 \\ CP_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} \end{pmatrix} \begin{pmatrix} P_1' \\ P_2' \\ P_3' \\ P_4' \\ P_5' \\ P_6' \end{pmatrix}$$

M x *M*

- Operations first elimination (Product): *D2 (M-D)*
  - Gaussian elimination n x n matrix, n = M - D requires An3 + Bn2 + Cn operations
  - Distribution of D determines average # of operations
    - Linked to channel model
  - Erasures IID Be(Pe):

$$A(MPe)^3 + B'(MPe)^2 + C'(MPe)$$

Sparse Network Codes

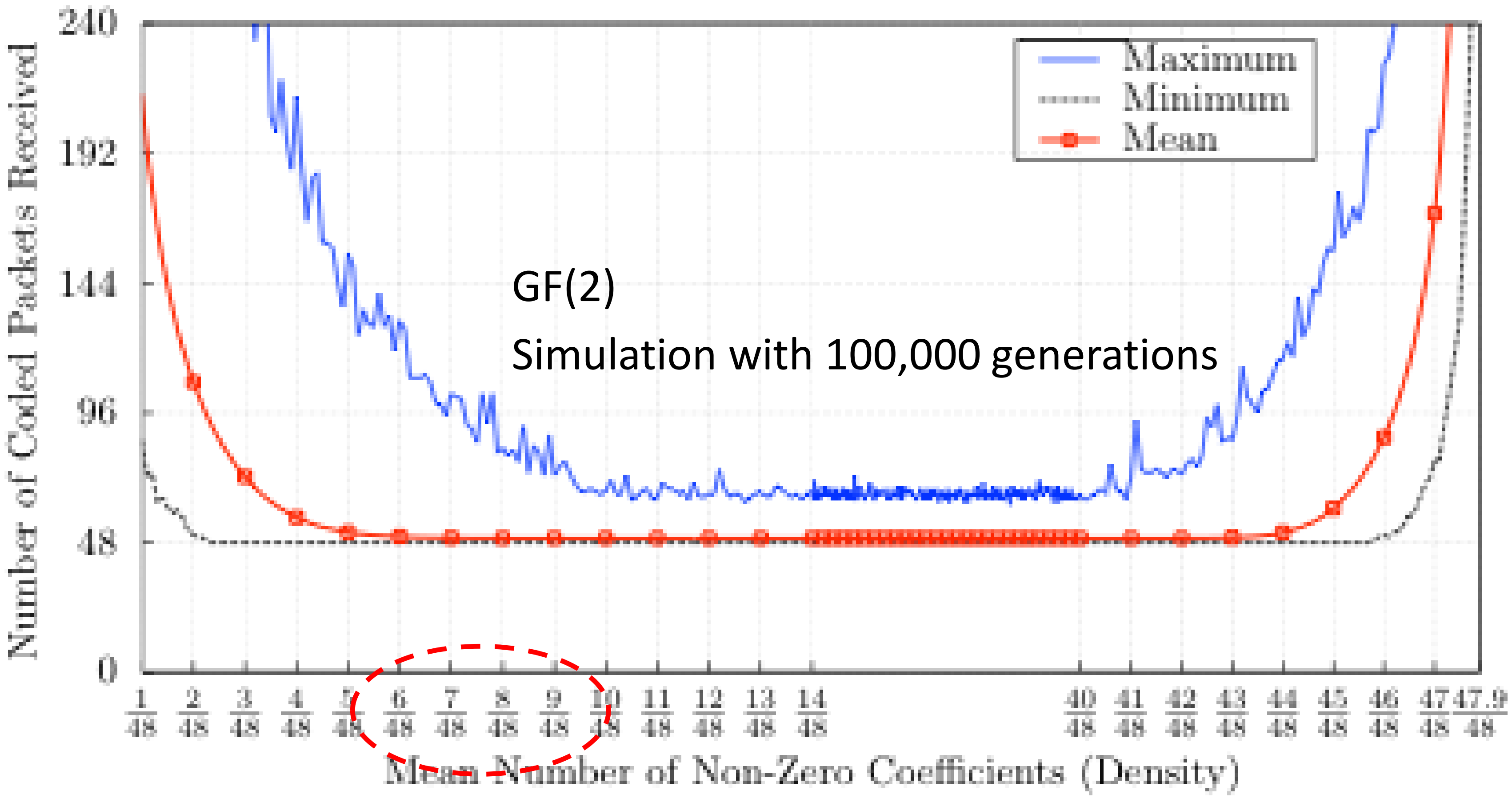


One or two orders of magnitude in the coding speed by sparsity.

Dualism Theory and Implementation!

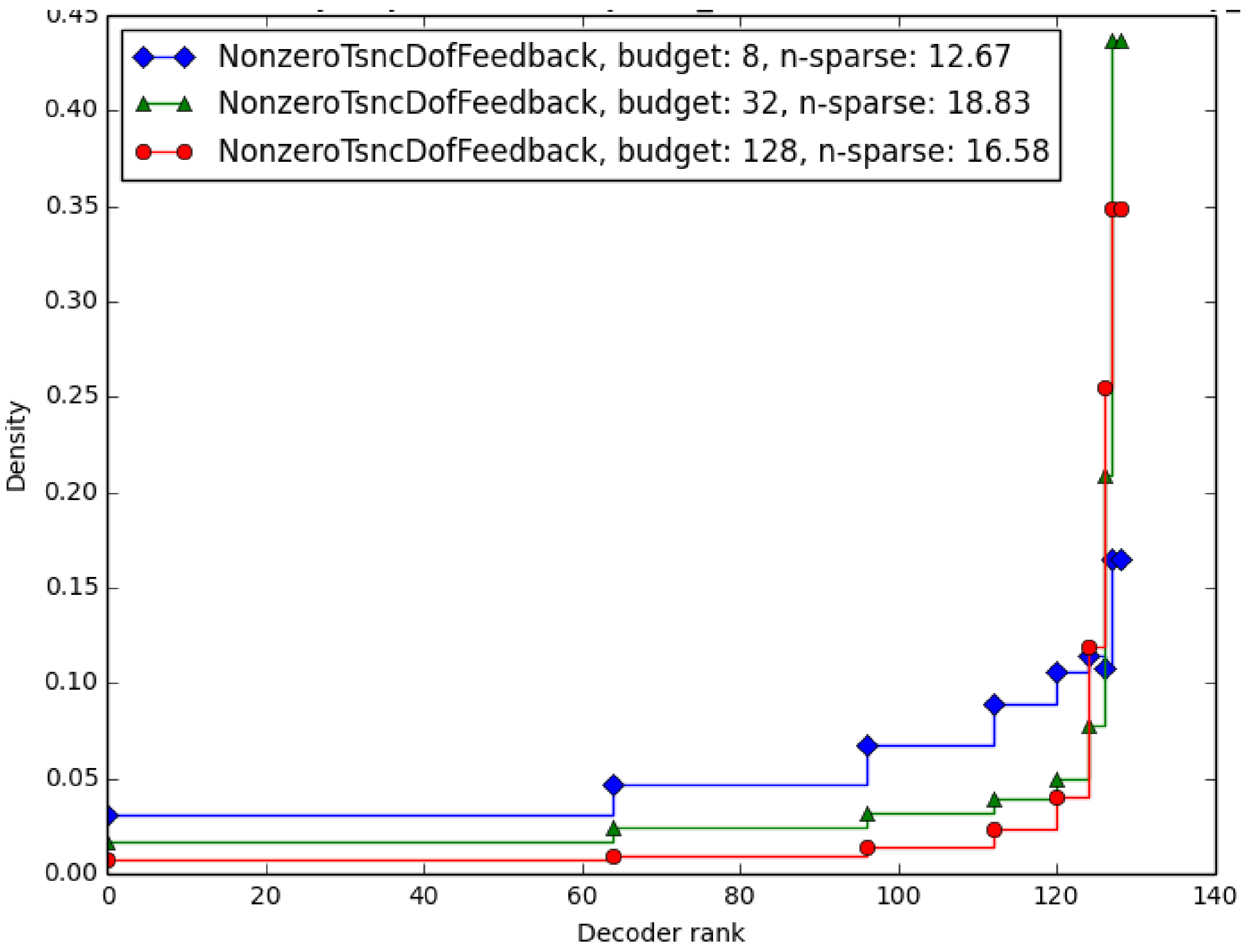
Sparse Network Codes

- Can we operate in the “speed up” region while maintaining good performance?
- Yes, but the key is not to use a fixed density
- (Tunable sparse network coding, 2012)

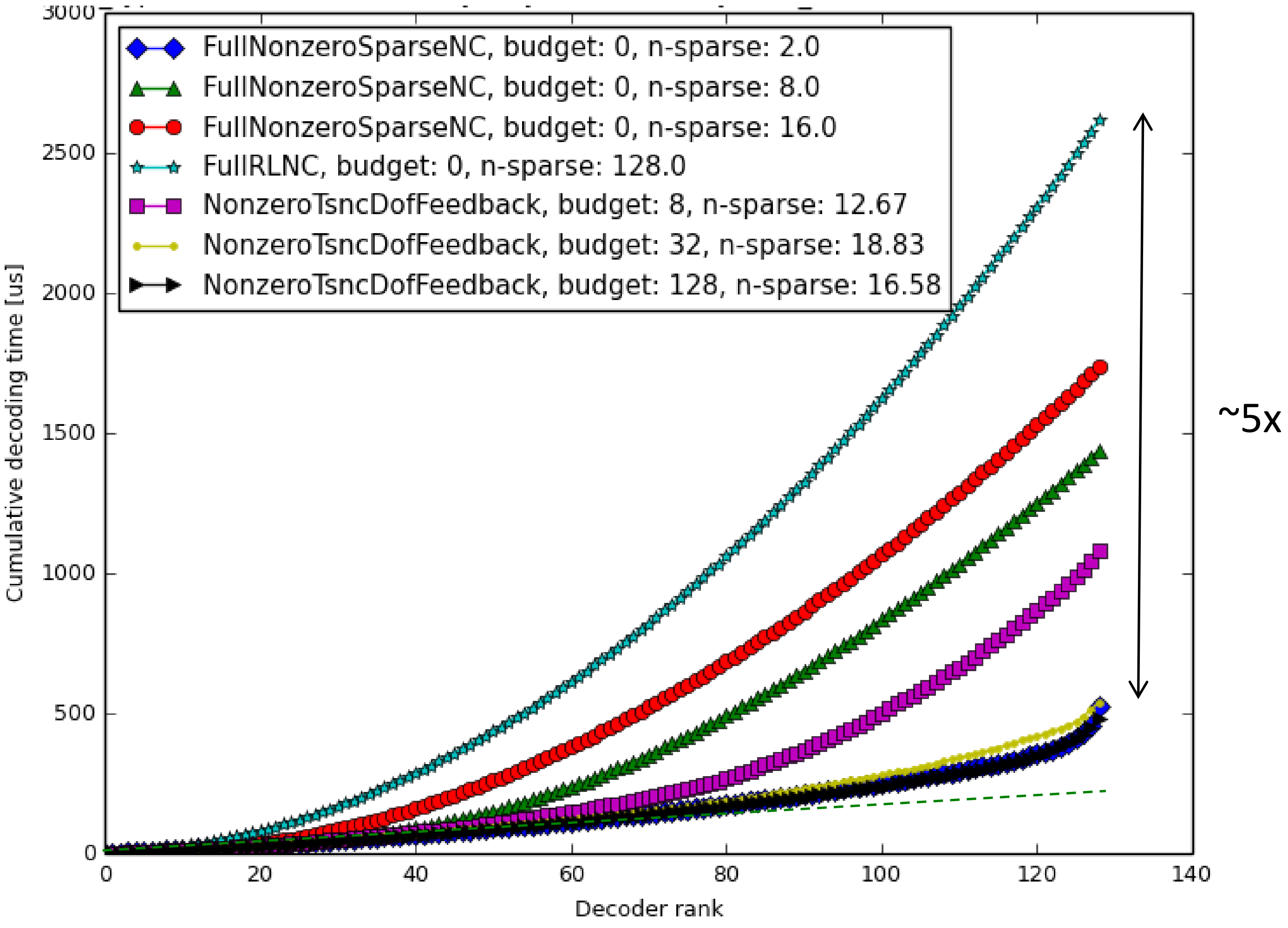


Tunable Sparse Network Codes

- Scheme with feedback: targets specific performance degradation



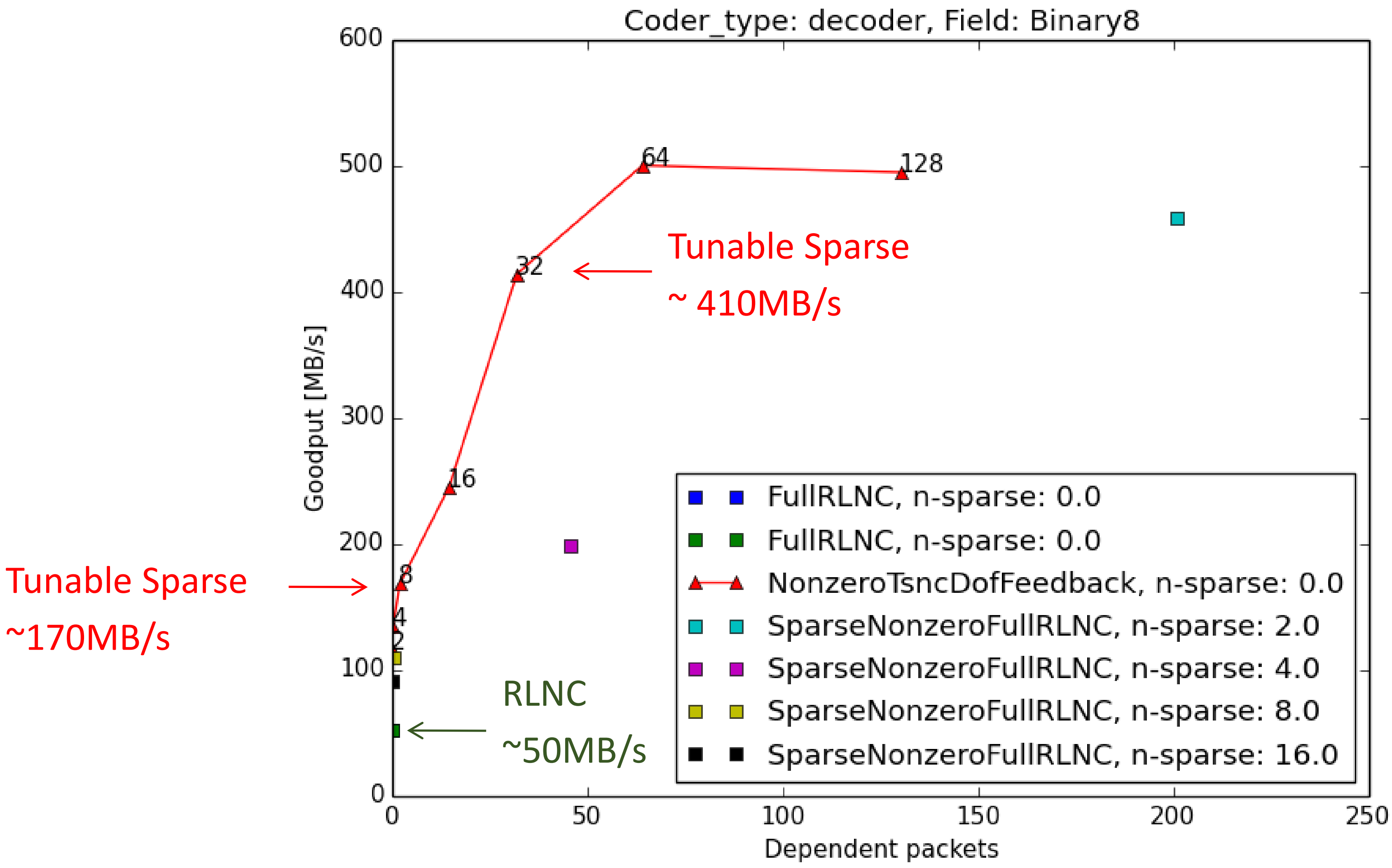
Tunable Sparse Network Codes



Scheme with feedback: targets specific performance degradation

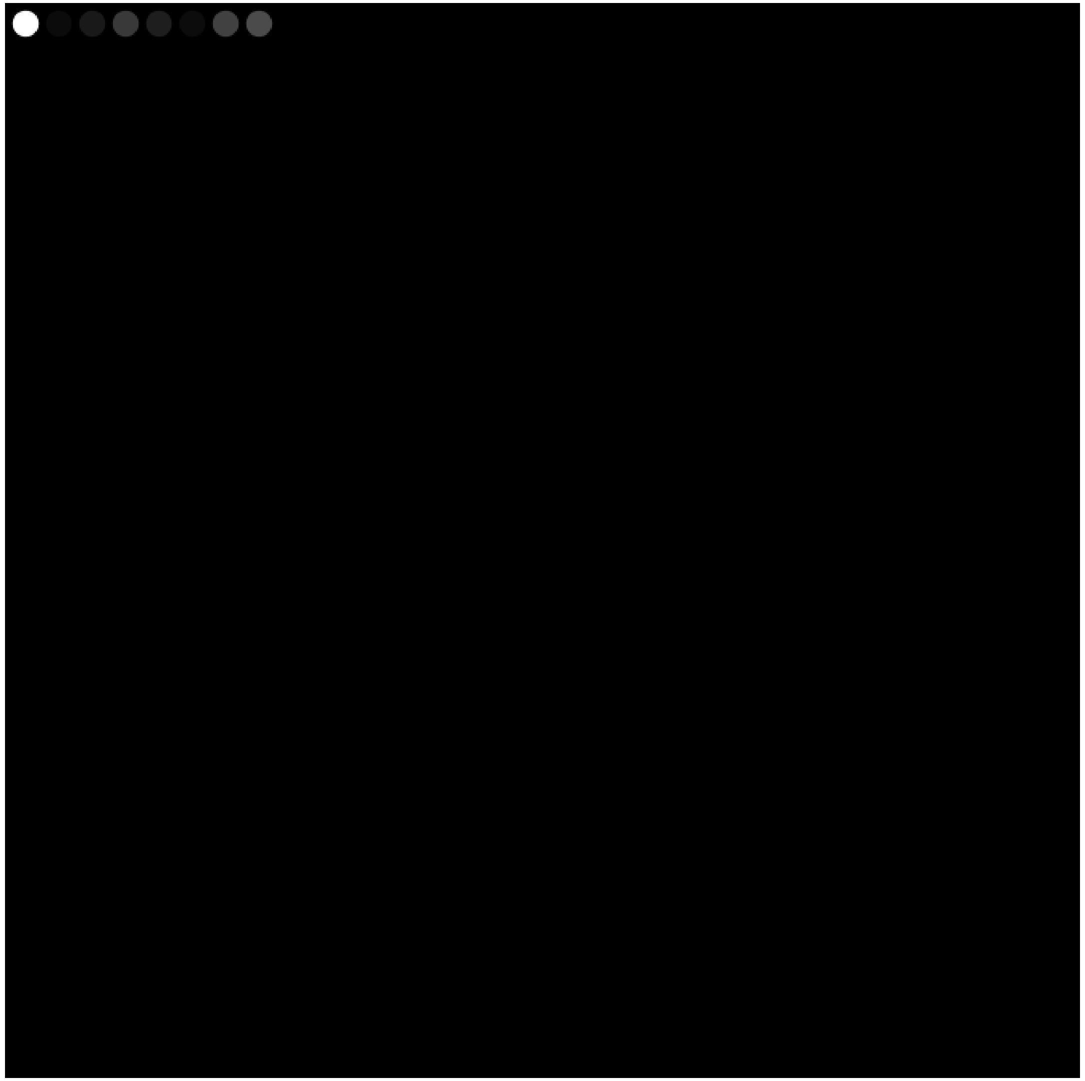
Example: 4 additional transmissions, i.e., 4% overhead

Tunable Sparse Network Codes



Sliding Window

- Terminology
  - On the fly encoding/decoding: Newly incoming packets are added to the block.
  - Sliding window: Newly incoming packets are added to the block and acknowledged packets are removed.
  - Sliding window can work without blocks

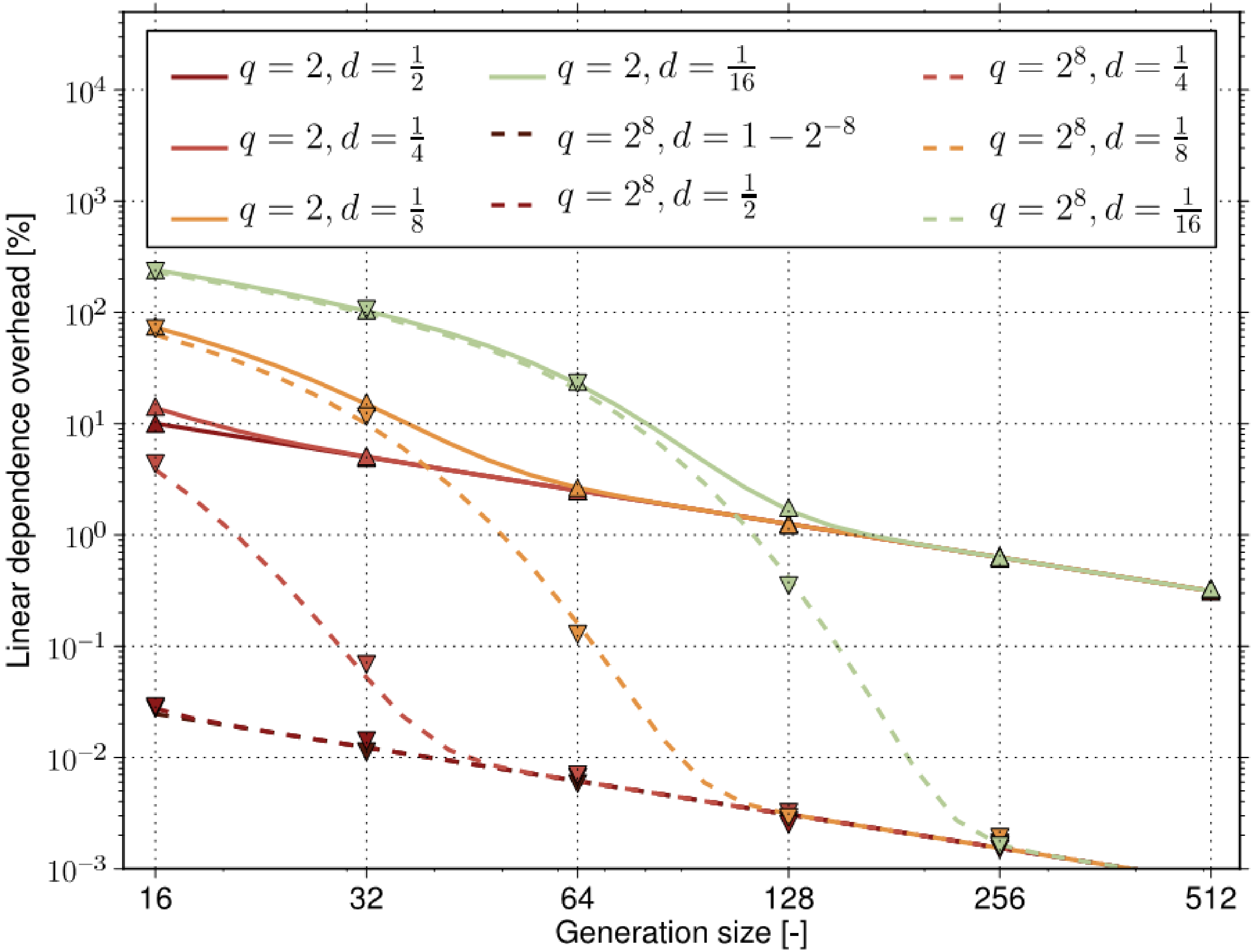


KODO: Overhead

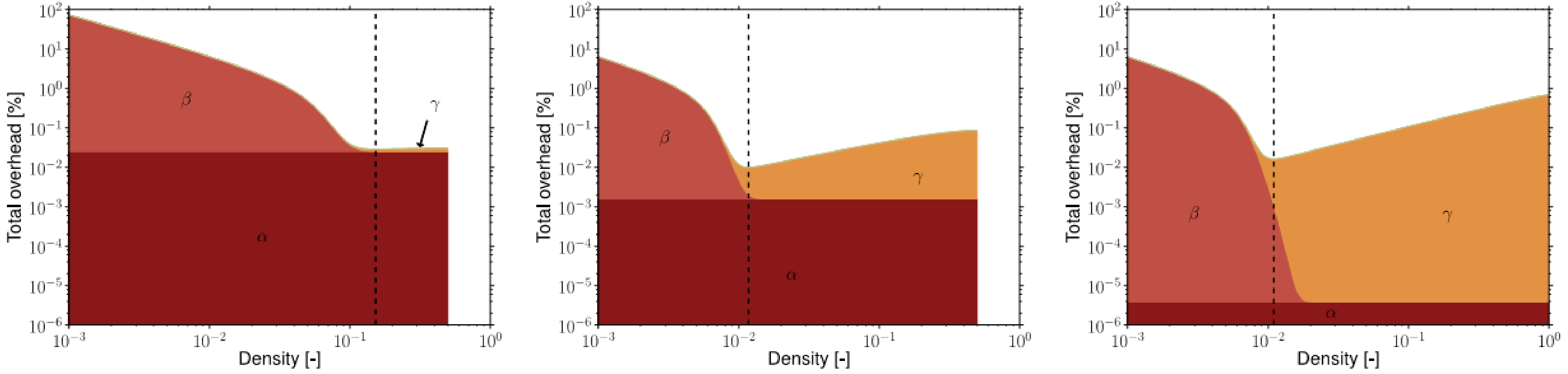
What is overhead?

- Overhead by encoding vector
  - Generation size G
  - Field size F
  - Payload P
  - Number of bits for each packet $A = G * \log2(F)$
- Linear dependent retransmissions
  - In case a packet is linear dependent it has to be retransmitted (together with the header)
  - $A = P + G * \log2(F)$

F/G and Linear Dependency



Designing rules …

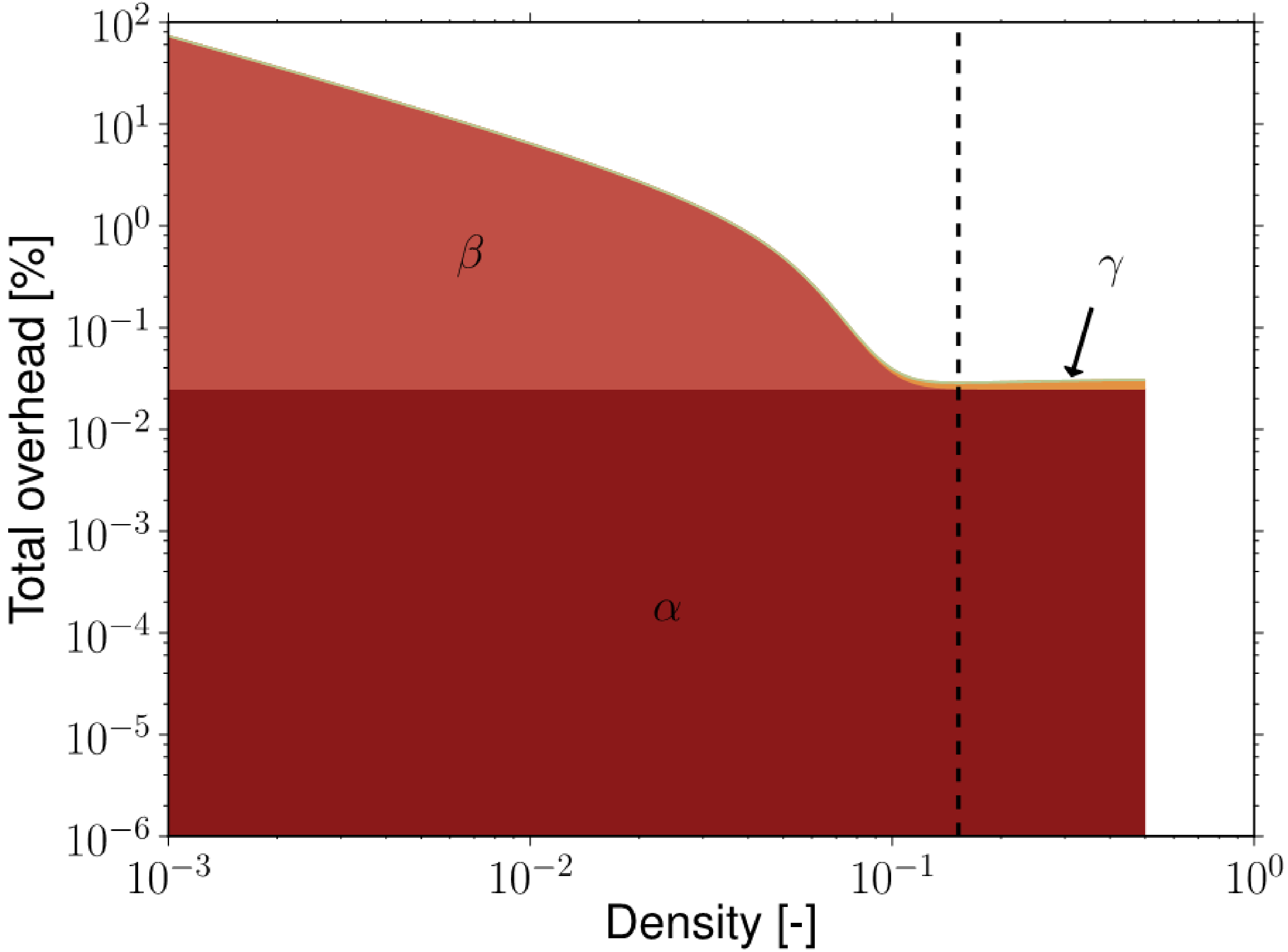


(a) $q = 2,\ g = 64$ (b) $q = 2,\ g = 1024$ (c) $q = 2^8,\ g = 1024$

a = additonal transmissions due to field size
b = additional packets due to linear dependency
c = bits in the encoding vector

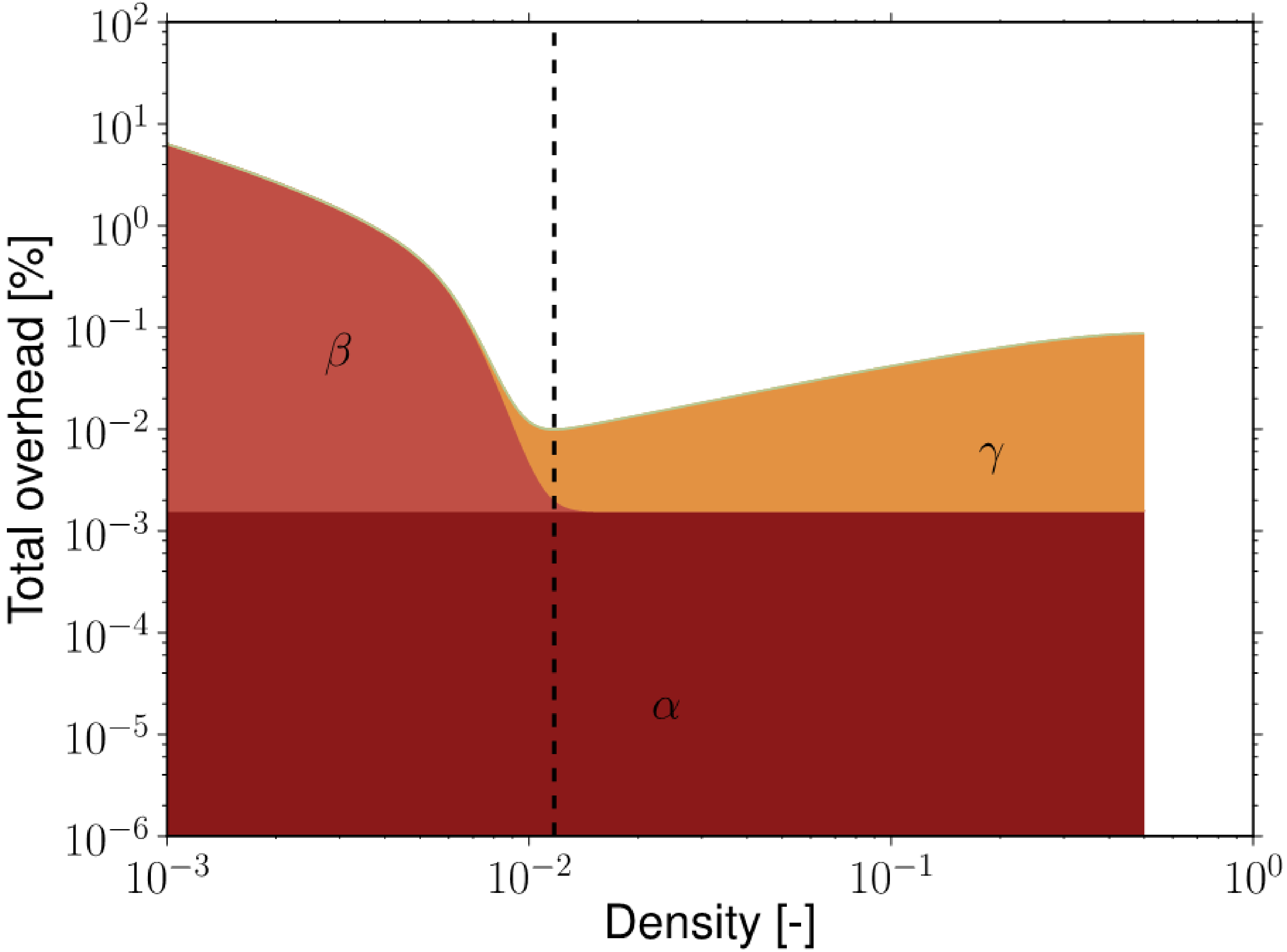
Janus Heide and Morten V. Pedersen and Frank H.P. Fitzek and Muriel Medard. **On Code Parameters and Coding Vector Representation for Practical RLNC.** 2011. in *IEEE International Conference on Communications (ICC) - Communication Theory Symposium*. Kyoto, Japan.

FS=2;GS=64



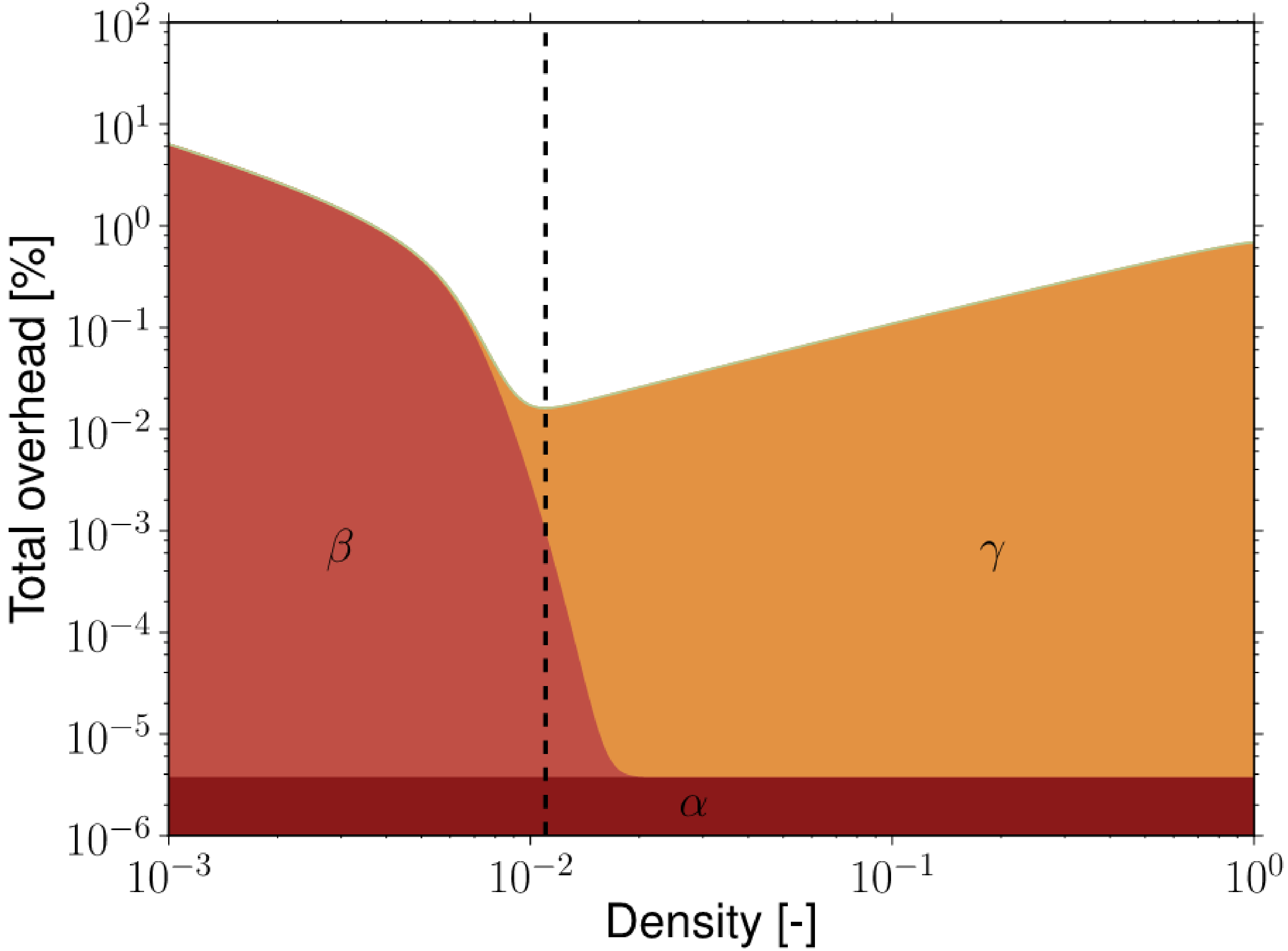
Total overhead [%]
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
β
γ
α
10^{-3}
10^{-2}
10^{-1}
10^{0}
Density [-]

FS=2;GS=1024



Total overhead [%]
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
β
γ
α
10^{-3}
10^{-2}
10^{-1}
10^{0}
Density [-]

FS=28;GS=1024



Total overhead [%]
10^2
10^1
10^0
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
β
γ
α
10^{-3}
10^{-2}
10^{-1}
10^0
Density [-]

Effects of Heterogeneous Packet lengths on Network Coding

TECHNISCHE UNIVERSITÄT DRESDEN

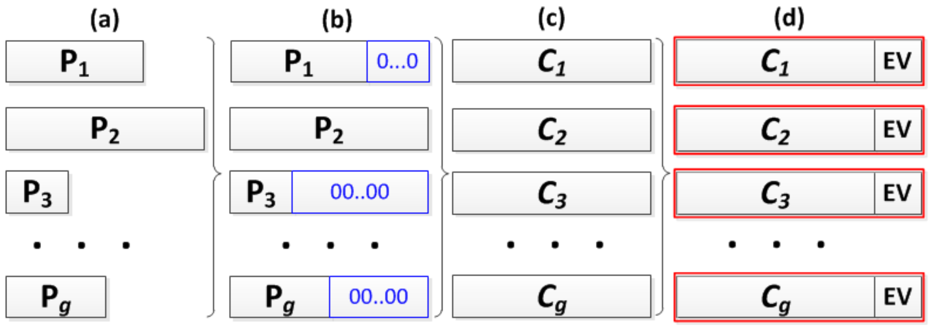
Effects of Heterogeneous Packet lengths on Network Coding

Overhead due to the RLNC encoding process

(a) Coding overhead: defined by the Encoding Vector size, containing the coding coefficients.

(b) Linear dependency overhead: need of extra transmissions when linear dependent combinations are received

(c) Padding overhead: created by coding with the biggest data packet of a generation with different sizes

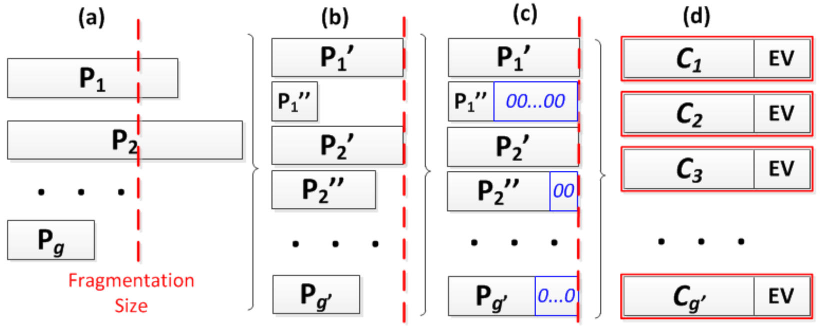


Naive assumption: all packets in a generation have the same size

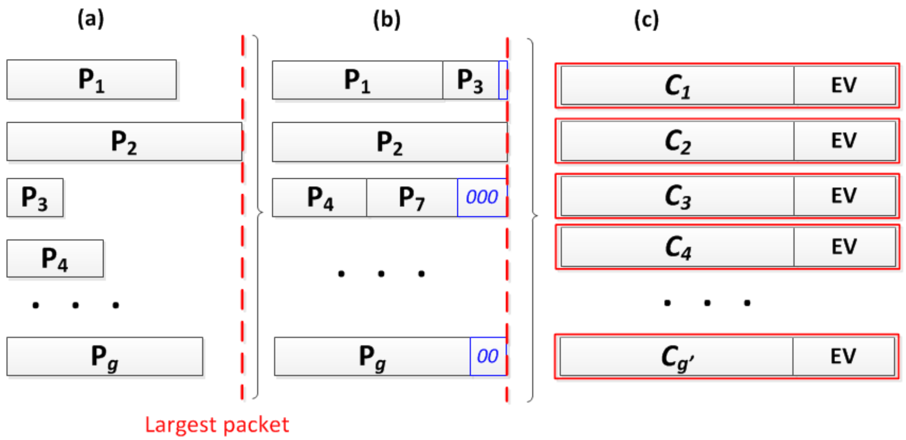
Real life data: heterogenousity of packet lengths

Packetization solutions

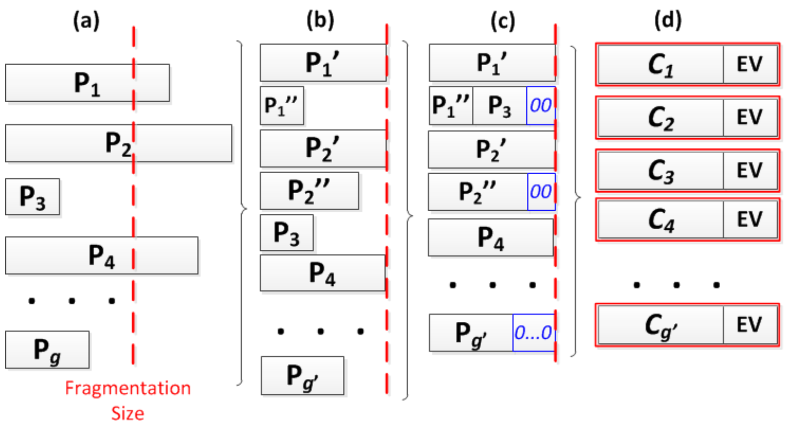
Chop & Code



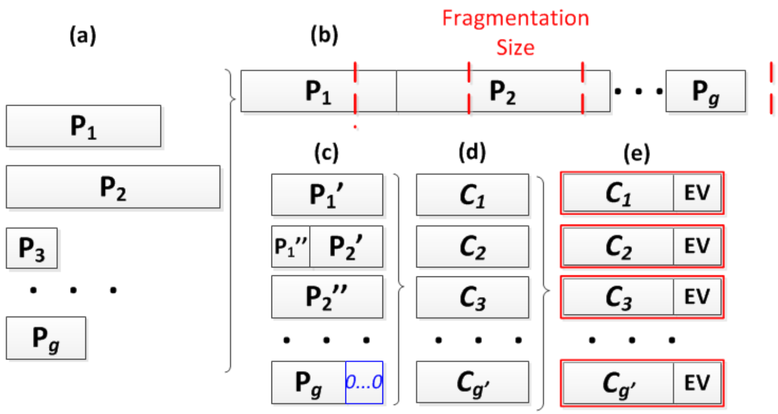
Bundle & Code



Chop & Bundle & Code

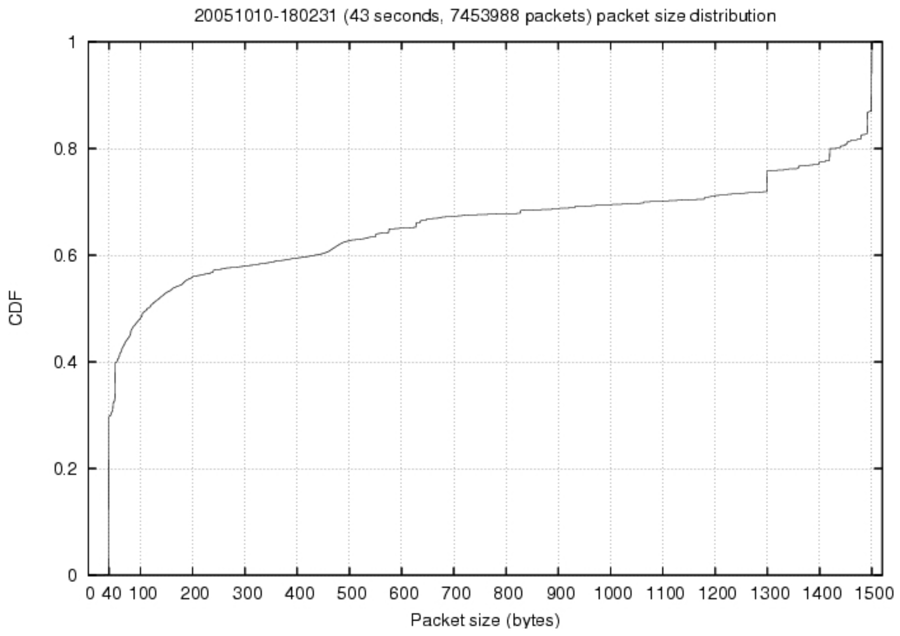


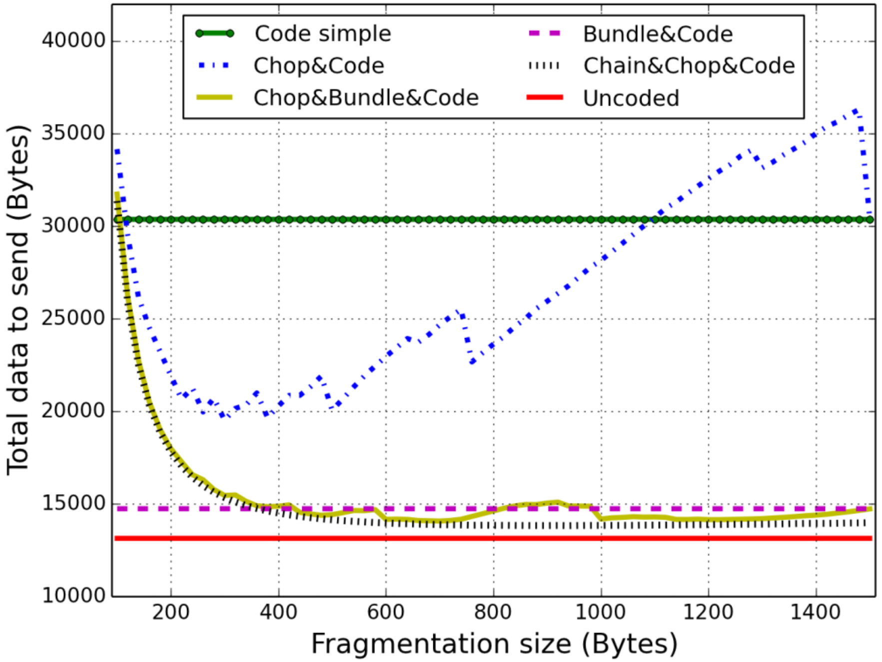
Chain & Chop & Code



Internet data from CAIDA

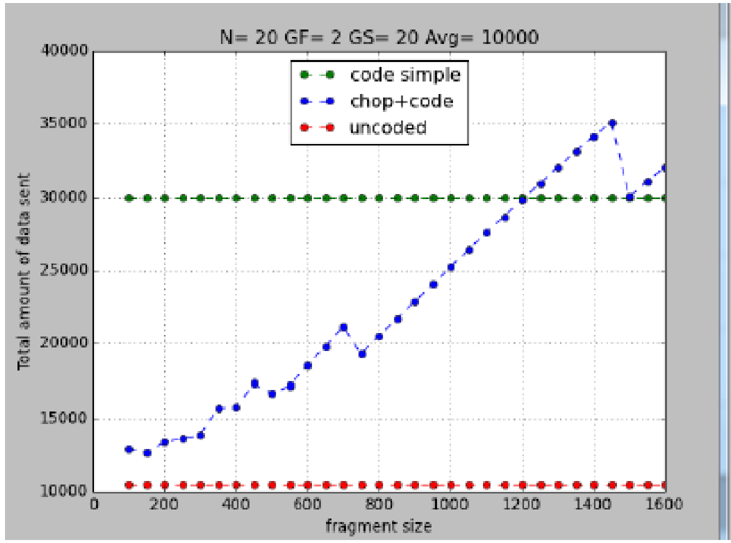
- Generations of 20 packets with field size q=2^8
- Packets are chosen randomly following the CDF, and run the simulations 10000 times and averaged the results
- Internet packet size distribution is mostly of 40 Bytes and 1500 Bytes (with approximate probabilities of 40% and 20% respectively). [1]

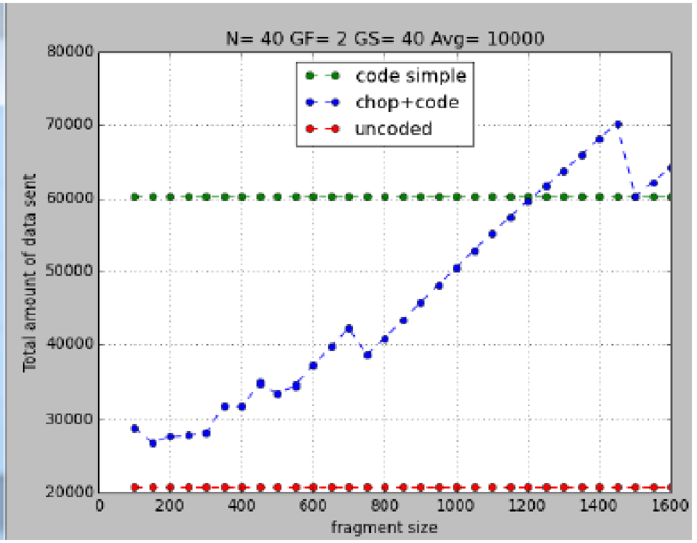




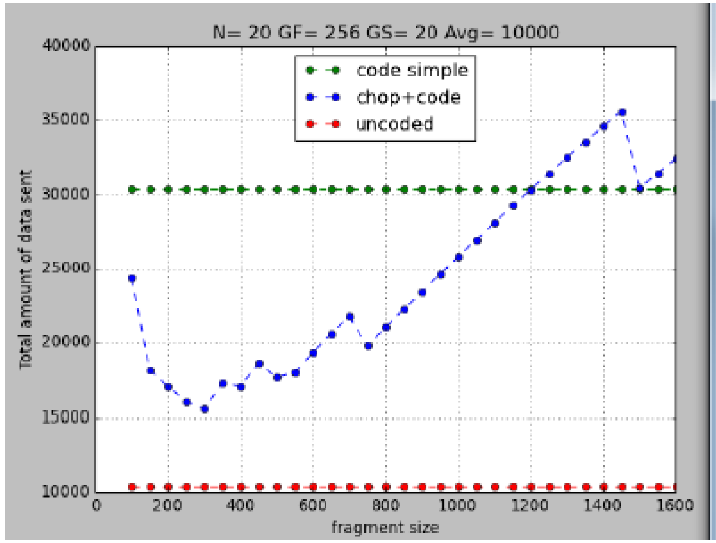
[1] http://www.isi.edu/~johnh/PAPERS/Sinha07a.pdf

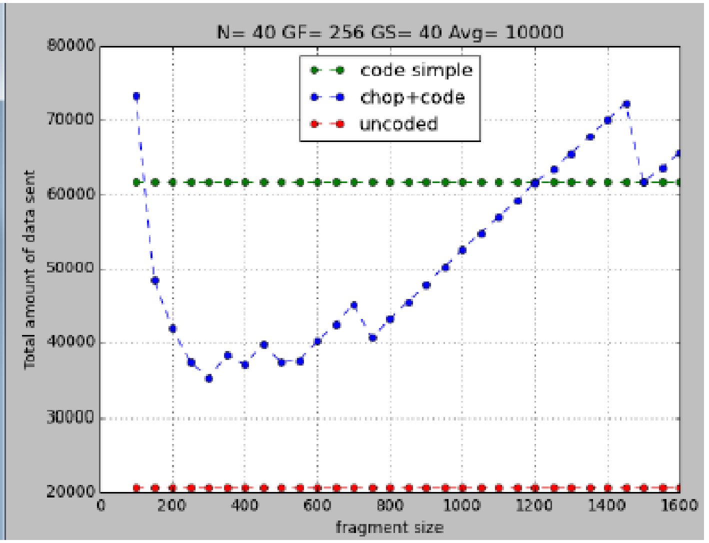
Packet Size Comparison 1





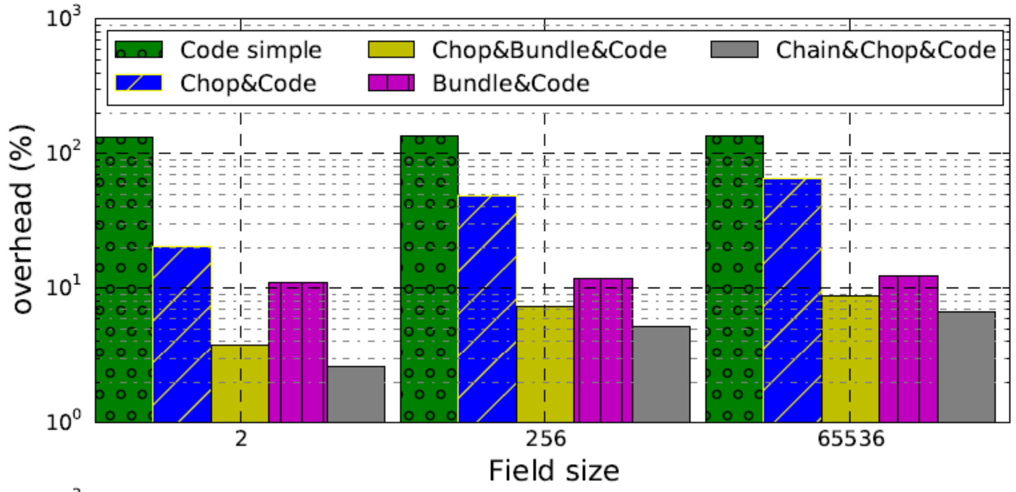
GF=2^8

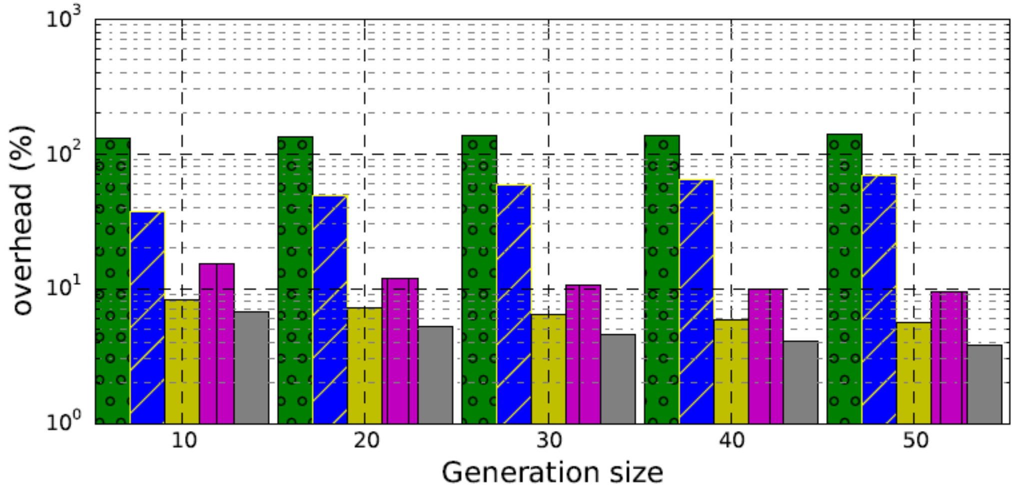




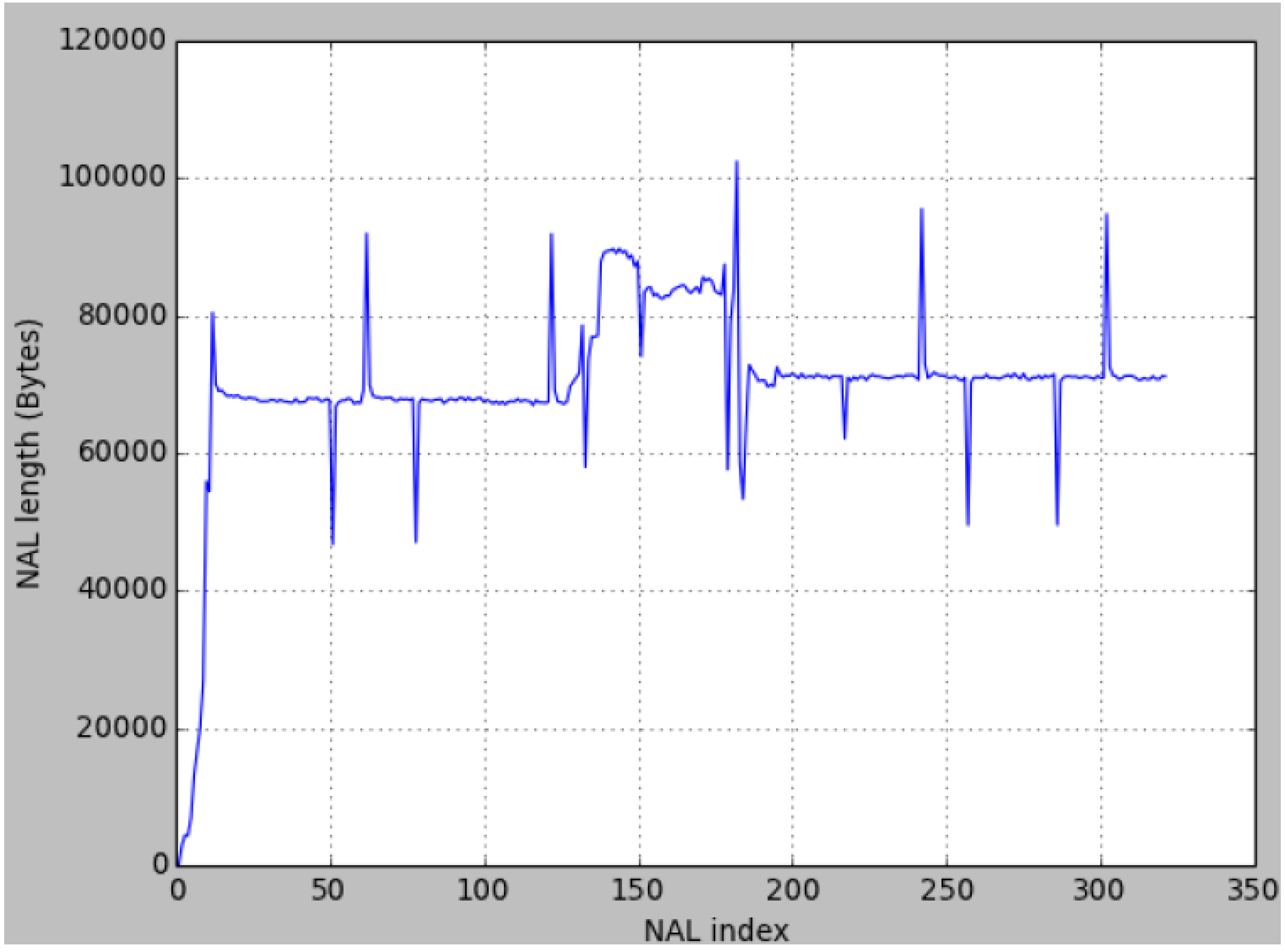
Internet data from CAIDA (2)

- Total coding overhead (%) added for each solution respect the uncoded data for:
  - different field sizes (and generation size constant of 20 packets)
  - different generation sizes (and field size constant q=28)
- Coding directly adds more than 100% of overhead.
- Chain& Chop& Code achieves to reduce to less than 5% for a large generation size.

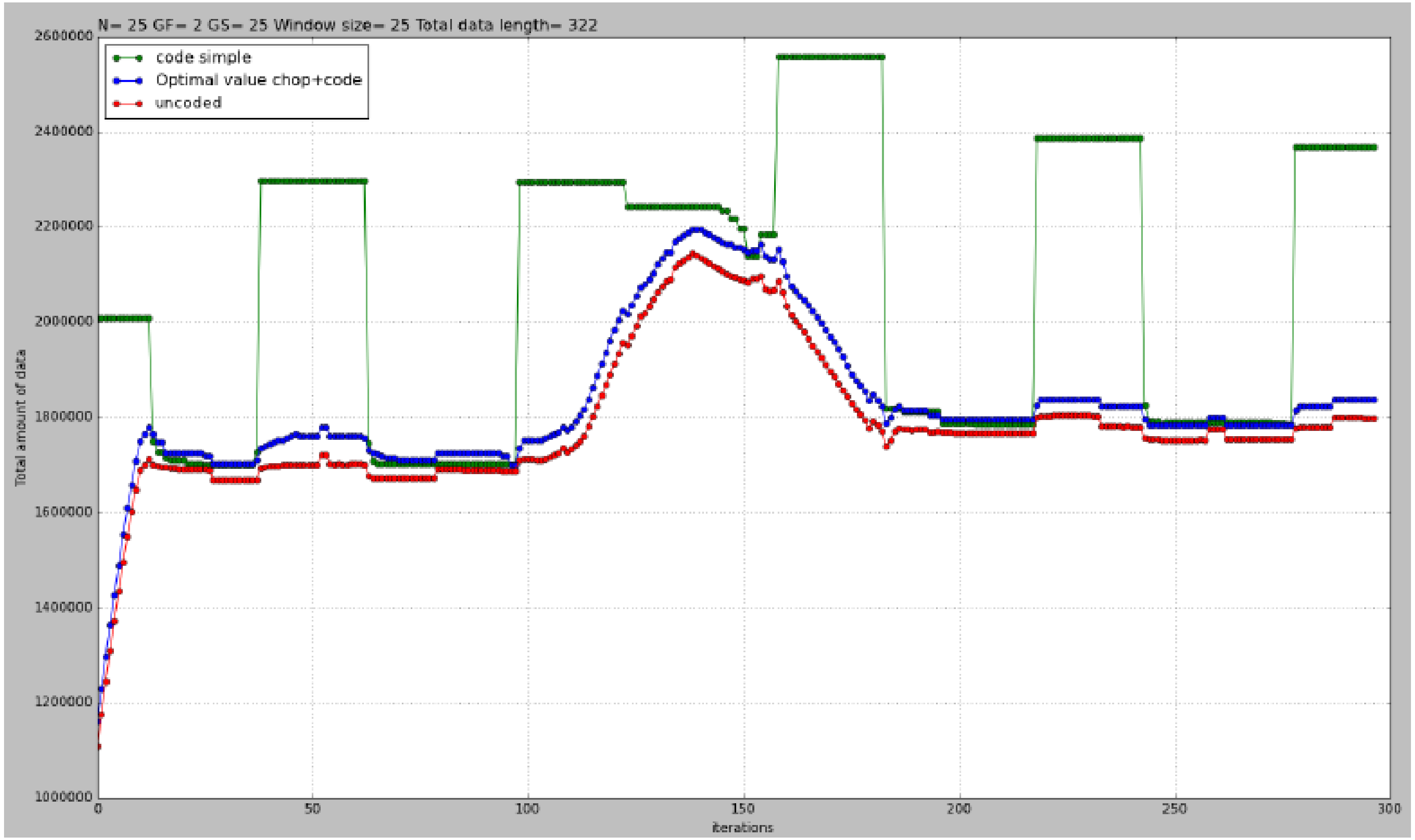




Packet Size Example 2



Packet Size Comparison 2



TECHNISCHE UNIVERSITÄT DRESDEN

KODO: Energy consumption and complexity

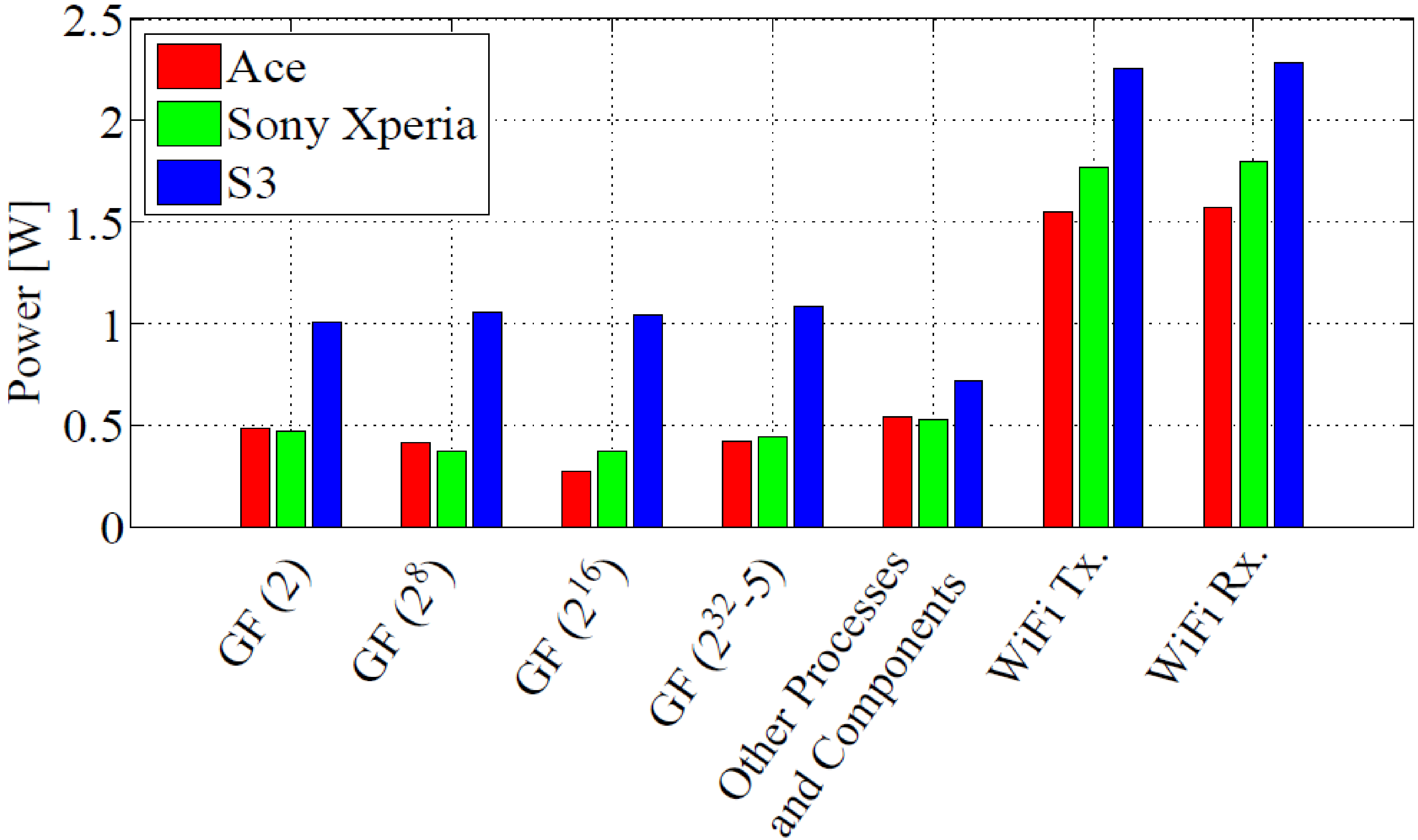
Energy consumption: KODO

Energy consumption: KODO

TABLE I
OVERVIEW OF THE FINITE FIELD IMPLEMENTATIONS USED.

$GF(2)$	The binary field $GF(2)$ was implemented using only the XOR operation, which typically yields very efficient implementations on most modern processors.
$GF(2^8)$	The binary extension field $GF(2^8)$ uses full multiplication and division lookup tables. Using this approach will replace the required polynomial multiplication or division of two field elements with a single table lookup. More details on these lookup tables are described in [17]. Addition and subtraction require only a bit-by-bit XOR.
$GF(2^{16})$	Computing a full multiplication and division table for $GF(2^{16})$ will typically require too much memory for most platforms (in the order of GBs). Instead, our implementation uses an extended logarithmic lookup, which reduces the memory requirements to the order of KBs. Addition and subtraction are performed as bit-by-bit XOR operation. Details are provided in [17].
$GF(2^{32}-5)$	This field is also known as an OPF (Optimal Prime Field). It has the advantage that it does not require any lookup tables and uses the standard arithmetic logic unit of the processor for all arithmetic operations. A detailed description of this field can be found in [18].

Energy consumption: KODO



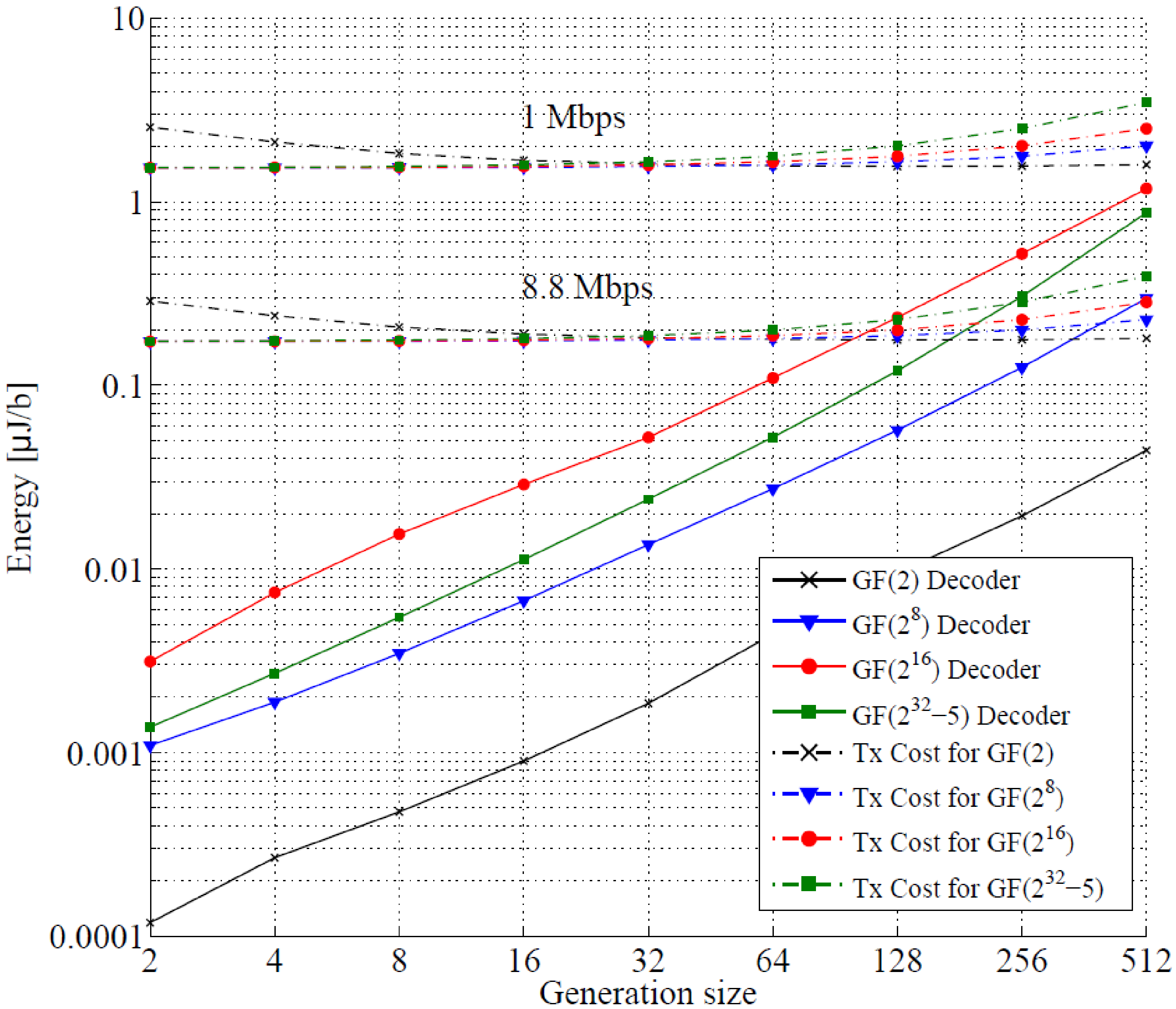
TECHNISCHE UNIVERSITÄT DRESDEN

Energy consumption: KODO

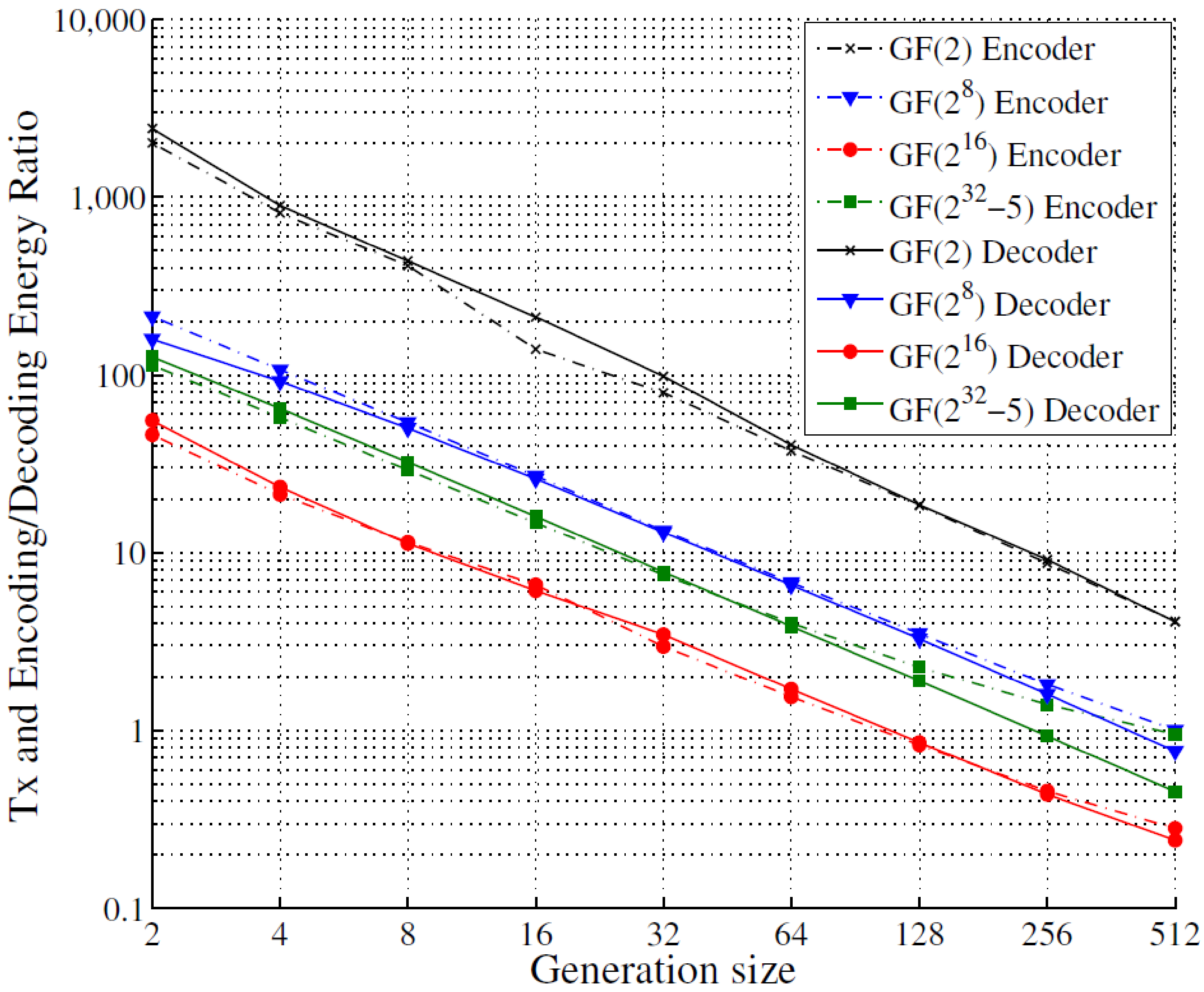
TABLE I
ENCODE AND DECODE PROCESSING SPEED (MBPS) AND AVERAGE PROCESSING POWER (W) FOR THREE MOBILE DEVICES.

Encoder	$GF(2)$			$GF(2^8)$			$GF(2^{16})$			$GF(2^{32}-5)$		
Pkts/Gen.	Ace	Xperia	S3	Ace	Xperia	S3	Ace	Xperia	S3	Ace	Xperia	S3
2	1675.8	3328.1	3916.22	238.4	463.2	669.71	45.5	100.5	216.67	175.6	293.1	396.52
4	921.1	1615.3	2413.61	120.1	233.0	342.34	18.9	46.2	99.14	89.1	148.0	209.18
8	527.9	938.2	1394.60	59.4	117.4	172.91	9.1	24.8	36.04	44.9	74.0	80.76
16	253.1	347.1	640.2	30.5	58.3	85.6	4.7	14.1	27.1	22.7	36.7	52.5
32	132.6	206.9	338.6	15.3	28.5	41.6	2.4	6.2	13.9	11.3	18.0	25.8
64	66.5	99.8	152.3	7.5	14.3	20.7	1.2	3.1	7.0	5.6	9.0	12.8
128	30.3	49.5	77.6	3.6	7.1	10.3	0.6	1.6	3.5	2.8	4.5	6.4
256	11.7	23.3	38.7	1.6	3.4	5.1	0.3	0.8	1.6	1.3	2.2	3.2
512	4.8	10.8	18.5	0.8	1.7	2.5	0.1	0.4	0.8	0.6	1.1	1.6
Decoder	Ace	Xperia	S3	Ace	Xperia	S3	Ace	Xperia	S3	Ace	Xperia	S3
2	2079.0	4005.0	5545.00	148.6	345.3	477.57	53.5	120.7	203.53	187.6	325.5	440.20
4	944.6	1774.0	2499.21	92.2	200.7	294.45	17.9	50.8	102.71	96.8	166.1	231.63
8	561.2	998.8	1384.16	51.9	108.9	158.78	8.7	24.4	45.01	51.1	82.2	103.40
16	287.0	527.3	783.7	27.7	56.2	82.7	5.1	13.1	25.7	25.0	39.9	56.6
32	150.6	256.2	403.2	14.4	27.8	40.3	2.5	7.3	13.3	12.0	18.7	26.7
64	67.6	107.8	169.6	7.3	13.8	20.0	1.2	3.5	6.9	5.6	8.6	12.2
128	31.9	49.8	77.3	3.4	6.6	9.6	0.6	1.6	3.4	2.4	3.7	5.4
256	12.7	24.4	38.4	1.4	3.0	4.5	0.3	0.7	1.4	0.9	1.5	2.1
512	4.8	10.7	18.1	0.6	1.3	1.9	0.1	0.3	0.6	0.3	0.5	0.8
Power (W)	0.486	0.474	1.012	0.421	0.377	1.057	0.279	0.378	1.041	0.425	0.448	1.090

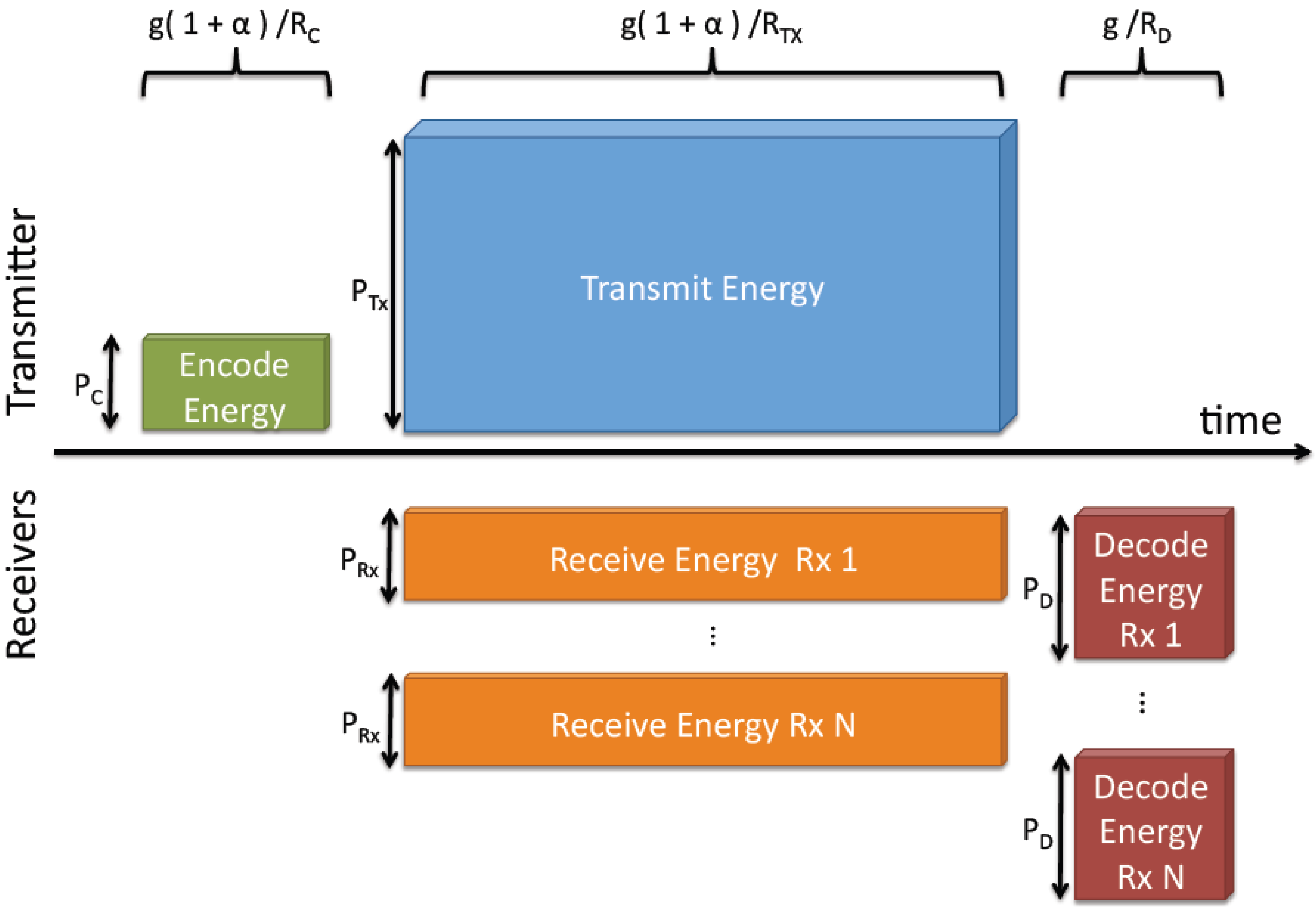
Energy consumption: KODO



Energy consumption: KODO



Energy consumption: KODO



Energy consumption: KODO