Technische Universität Dresden

Opportunistic Routing with Network Coding in Powerline Communications

Ievgenii Tsokalo

von der Fakultät Elektrotechnik und Informationstechnik der Technischen Universität Dresden

zur Erlangung des akademischen Grades eines

Doktoringenieurs

(Dr.-Ing.)

genehmigte Dissertation

Vorsitzender: Univ.-Prof. Dr. techn. Klaus Janschek

Gutachter: Prof. Dr.-Ing. Ralf Lehnert — Tag der Einreichung: 08.05.2017

Prof. Dr.-Ing. Lutz Lampe — Tag der Verteidigung: 27.10.2017

With love to my wife Anna, mother Oleksandra, father Anatolii and sisters Lena and Tanya

The author thanks Prof. Dr.-Ing. Ralf Lehnert, Technische Universität Dresden, for his technical and mental support throughout all research and preparation of the written thesis. Also, he thanks Prof. Dr.-Ing. Dr.h.c. Frank H.P. Fitzek, Technische Universität Dresden, for support and consultations in questions of Network Coding. Also, he thanks Prof. Dr.-Ing. Lutz Lampe, British University of Columbia, for his assistance in questions of physical layer and PLC channel modeling, and providing the possibility for collaboration with his research team.

Abstract

Opportunistic Routing (OR) can be used as an alternative to the legacy routing (LR) protocols in networks with a broadcast lossy channel and possibility of overhearing the signal. The power line medium creates such an environment. OR can better exploit the channel than LR because it allows the cooperation of all nodes that receive any data. With LR, only a chain of nodes is selected for communication. Other nodes drop the received information. We investigate OR for the one-source one-destination scenario with one traffic flow. First, we evaluate the upper bound on the achievable data rate and advocate the decentralized algorithm for its calculation. This knowledge is used in the design of Basic Routing Rules (BRR). They use the link quality metric that equals the upper bound on the achievable data rate between the given node and the destination. We call it the node priority. It considers the possibility of multi-path communication and the packet loss correlation. BRR allows achieving the optimal data rate pertaining certain theoretical assumptions. The Extended BRR (BRR-E) are free of them. The major difference between BRR and BRR-E lies in the usage of Network Coding (NC) for prognosis of the feedback. In this way, the protocol overhead can be severely reduced. We also study Automatic Repeat-reQuest (ARQ) mechanism that is applicable with OR. It differs to ARQ with LR in that each sender has several sinks and none of the sinks except destination require the full recovery of the original message. Using BRR-E, ARQ and other services like network initialization and link state control, we design the Advanced Network Coding based Opportunistic Routing protocol (ANChOR). With the analytic and simulation results we demonstrate the near optimum performance of ANChOR. For the triangular topology, the achievable data rate is just 2% away from the theoretical maximum and it is up to 90% higher than it is possible to achieve with LR. Using the G.hn standard, we also show the full protocol stack simulation results (including IP/UDP and realistic channel model). In this simulation we revealed that the gain of OR to LR can be even more increased by reducing the head-of-the-line problem in ARQ. Even considering the ANChOR overhead through additional headers and feedbacks, it outperforms the original G.hn setup in data rate up to 40% and in latency up to 60%.

Abbreviations

ACK Acknowledgement

ADC Analog to Digital Converter

ANChOR Advanced Network Coding based Opportunistic Routing protocol

AODV Ad-Hoc On Demand Distance Vector Routing Protocol

ARQ Automatic Repeat reQuest

AWGN Additive White Gaussian Noise

BAA Bit-Allocation Algorithm

BAT Bit Allocation Table

B.A.T.M.A.N. The Better Approach To Mobile Adhoc Networking

BB-PLC Broadband PLC

BER Bit Error Ratio

BLER BLock Error Ratio

BRR Basic Routing Rules

BRR-E Basic Routing Rules Extended

CA Collision Avoidance

CBTS Contention-Based Transmission Slots

CCACK Cumulative Coded Acknowledgments

CD Collision Detection

CDF Cumulative Distribution Function

CFTS Contention-Free Transmission Slots

CRC Cyclic Redundancy Check code

CS Coded Symbol

CSMA Carrier Sense Multiple Access

CSPR Common Single-path Routing Protocol

CTF Channel Transfer Function

CV Coding Vector

CW Contention Window

DCMC Discrete input Continuous output Memoryless Channel

DLL Data Link Layer

DOF Degree Of Freedom

DSR Dynamic Source Routing Protocol

EATT Expected Anypath Transmission Time

EAX Expected Anypath Transmission Count

ExOR Extremely Opportunistic Routing

FEC Forward Error Correction

FSK Frequency Shift Keying

GID Generation IDentifier

HARQ Hybrid Automatic Repeat reQuest

LDPC Low-Density Parity-Check code

LDR Linear Dependence Ratio

LDS Linearly Dependent Symbols

LLC Logical Link Control layer

LPDU Logical link control layer Protocol Data Unit

LRT Local Routing Table

LT Luby Transform code

MAC Multiple Access Control layer

MAP Multiple Access Plan

MDS Maximum Distance Separation code

MORE MAC-independent Opportunistic Routing & Encoding

MPDU Multiple access control layer Protocol Data Unit

NB-PLC Narrowband PLC

NC Network Coding

NDM Network Discovery Message

NSB Null Space Base

OFDM Orthogonal Frequency Division Multiplexing

OGID The Oldest Generation IDentifier

OLSR Optimized Link State Routing

ORP Opportunistic Routing Protocol

OSI Open Systems Interconnection model

PDU Protocol Data Unit

PEC Packet Erasure Channel

PER Packet Error Ratio

PHY Physical layer

PLC PowerLine Communications

PSD Power Spetral Density

Pt2MPt Point-To-MultiPoint

Pt2Pt Point-To-Point

QC-LDPC Quasi-Cyclic Low-Density Parity-Check code

QoS Quality of Service

RAPTOR RAPid TORnado code

RLNC Random Linear Network Coding

RP Routing Protocol

RPI Routing Protocol Information

RPIM Routing Protocol Information Message

RR Retransmission Request

RRM Retransmission Request Message

RS Recoded Symbol

RSSI Receive Signal Strength Indicator

RTO Retransmission TimeOut

RTT Round Trip Time

RV Random Variable

SNR Signal to Noise Ratio

SNS Simple Network Simulator

SOAR Simple Opportunistic Adaptive Routing protocol for wireless mesh networks

SRP Spanning tree Routing Protocol

SSN Serial Sequence Number

TCP Transmission Control Protocol

TDD Time Division Duplexing

TDM Time Division Multiplexing

TDMA Time Division Multiple Access

TDMAP Time Division Multiplexing Access Plan

TTL Time To Live

UDP User Datagram Protocol

UPA Universal Powerline Alliance

List of Symbols

α	Throughput gain of ORP to LRP
$\mathcal{G}$	Network graph
$\rho(v)$	Unique DOF of vertex v
ε	Packet erasure ratio
ε_e	Packet erasure ratio on edge e
$\vartheta(Z)$	Total DOF of vertex v
b_i	BER after decoding for the i-th subcarrier if all bits of the given FEC codeword are modulated on this subcarrier
C	Channel capacity
c	Coding rate
c	Coding vector
$c(v)$	Coding rate of vertex v
D	Data rate
$d(v)$	Sending data rate of vertex v
D^C	Data rate with CSRP
D^O	Data rate with ORP
ds	Distance
E	Set of edges

$e = (v, u)$ Edge with source vertex v and sink vertex u

$E[X]$ Expectation value of RV X

f_{uv} Traffic flow from vertex u to vertex v

G_s Generation size

$GF(q)$ Galois Field of size q

H^i Hash matrix with index i

$I(S)$ Mutual information over the cut S where $S = \{v_s\}$

$I(v)$ Set of input edges of vertex v

$I(X, Y)$ Mutual information between messages X and Y

J Average jitter

L Average latency

$L(v)$ Cooperation group associated with the sender vertex v

M_F FEC block size

N_a Average ARQ window size

N_c Size of the cooperation group

N_m Maximum ARQ window size

N_r The average number of LLC packets in a MPDU that pass CRC check

$O(v)$ Set of ouput edges of vertex v

O_c Offset overhead of the PHY frame

O_p Coefficient for the proportional overhead of the PHY frame

$p(v)$ Priority of vertex v

$p_f(v, u)$ Filtering probability of vertex v for information received from vertex u

p_l Linear dependence ratio

q Galois field size

R Upper bound on achievable data rate

$Rank(A)$ Rank of matrix A

V Set of vertices

$V^+(E)$ Set of sink vertices of edges in E

$V^-(E)$ Set of source vertices of edges in E

v_d Destination vertex

v_r Relay vertex

v_s Source vertex

Glossary

Cooperation Inter-action of the group of vertices that target to deliver the same message.

Cooperation group The set of vertices that the given sender has defined as the cooperting nodes.

Cut $S \subset V$, $v_s \in S$, $v_d \notin S$.

Degree of freedom In RLNC, it is used to denote the dimension of the space spanned by a certain set of vectors.

False-positive probability The probability of the wrong interpretation of the feedback information in CCACK, $P[(B = false) \cap (C = true)]$. B is the RV describing the CRC status of the certain CS at vertex u that is sent by the vertex v, and C is the RV describing the guess of v about the CRC status of this CS at u basing on the CCACK feedback information.

Filtering coefficient Obtains values in range $[0; 1]$; "1" means that all received data will be forwarded and "0" means the opposite.

G3 Narrow-band PLC standard.

IEEE1901 Broad-band PLC standard.

IEEE1901.2 Narrow-band PLC standard.

Intra-flow NC Coding of data belonging to the same flow but arriving from different nodes.

Linear dependency ratio The ratio of all linear dependent codewords to all codewords received.

Linear dependent coding vectors The vectors that can be derived from each other by multiplication on some scalar.

Loss ratio Analog to Packet Erasure Ratio (PER).

Maximum ARQ window size The maximum number of PDUs that can be sent without receiving the acknowledgement. It equals the maximum number of PDUs that can be stored in the receiver buffer and the maximum number of PDUs that can be simultenously acknowledged..

Packet erasure Discarding of the packet due to the error detection by the CRC check.

Packet Erasure Ratio (PER) The ratio of the number of packets that do not pass the CRC check to the total number of packets.

Packet loss Analog to packet erasure.

Packet train The set of the LLC layer protocol data units that form the payload of the MAC protocol data unit.

Vertex priority The outer bound on the achievable data rate between the given vertex and the destination.

Contents

1 Introduction **2**

1.1 Intra-flow Network Coding 6

1.2 Random Linear Network Coding (RLNC) 7

2 Performance Limits of Routing Protocols in PowerLine Communications (PLC) **13**

2.1 System model 14

2.2 Channel model 14

2.3 Upper bound on the achievable data rate 16

2.4 Achieving the upper bound data rate 17

2.5 Potential gain of Opportunistic Routing Protocol (ORP) over Common Single-path Routing Protocol (CSPR) 19

2.6 Evaluation of ORP potential 19

3 Opportunistic Routing: Realizations and Challenges **24**

3.1 Vertex priority and cooperation group 26

3.2 Transmission policy in idealized network 34

3.2.1 Basic Routing Rules (BRR) 36

3.3 Transmission policy in real network 40

3.3.1 Purpose of Network Coding (NC) in ORP 41

3.3.2 Extended Basic Routing Rules (BRR) (BRR-E) 43

3.4 Automatic ReQuest reply (ARQ) 50

3.4.1 Retransmission request message contents 51

3.4.2 Retransmission Request (RR) origination and forwarding 66

3.4.3 Retransmission response 67

3.5 Congestion control 68

3.5.1 Congestion control in our work . . . 70
3.6 Network initialization . . . 74
3.7 Formation of the cooperation groups (coalitions) . . . 76
3.8 Advanced Network Coding based Opportunistic Routing protocol (ANChOR) header . . . 77
3.9 Communication of protocol information . . . 77
3.10 ANChOR simulation . . . 79
3.10.1 ANChOR information in real time . . . 80
3.10.2 Selection of the coding rate . . . 87
3.10.3 Routing Protocol Information (RPI) broadcasting frequency . . . 89
3.10.4 RR contents . . . 91
3.10.5 Selection of RR forwarder . . . 92
3.10.6 ANChOR stability . . . 92
3.11 Summary . . . 95

4 ANChOR in the Gigabit Home Network (G.hn) Protocol 97
4.1 Compatibility with the PLC protocol stack . . . 99
4.2 Channel and noise model . . . 101
4.2.1 In-home scenario . . . 102
4.2.2 Access network scenario . . . 102
4.3 Physical layer (PHY) layer implementation . . . 102
4.3.1 Bit Allocation Algorithm (BAA) . . . 103
4.4 Multiple Access Control layer (MAC) layer . . . 109
4.5 Logical Link Control layer (LLC) layer . . . 111
4.5.1 Reference Automatic Repeat reQuest (ARQ) . . . 111
4.5.2 Hybrid Automatic Repeat reQuest (HARQ) in ANChOR 114
4.5.3 Modeling Protocol Data Unit (PDU) erasures on LLC . . 116
4.6 Summary . . . 117

5 Study of G.hn with ANChOR 119
5.1 ARQ analysis . . . 119
5.2 Medium and PHY requirements for “good” cooperation . . . 125
5.3 Access network scenario . . . 128

5.4 In-home scenario 135
5.4.1 Modeling packet erasures 136
5.4.2 Linear Dependence Ratio (LDR) 139
5.4.3 Worst case scenario 143
5.4.4 Analysis of in-home topologies 145

6 Conclusions **154**

A Proof of the neccessity of the exclusion rule **160**

B Gain of ORPs to CSRPs **163**

C Broadcasting rule **165**

D Proof of optimality of BRR for triangular topology **167**

E Reducing the retransmission probability **168**

F Calculation of Expected Average number of transmissions (EAX) for topologies with bi-directional links **170**

G Feedback overhead of full coding matrices **174**

H Block diagram of G.hn physical layer in ns-3 model **175**

I PER to BER mapping **177**

List of Figures

1.1 Scenarios for intra-flow NC (typical in PLC) 6
1.2 Forming generations 7
1.3 Encoding example 8
1.4 Recoding example 8

2.1 Fully meshed network. Multi-relay scenario 13
2.2 Potential gain α (alpha) of ORP over CSPR (eq. 2.5) for topology in fig. 1.1a 20
2.3 Potential gain α (alpha) of ORP over CSPR (eq. 2.5) for topology in fig. 2.4 21
2.4 Diamond topology. Special case 22
2.5 Potential gain α (alpha) of ORP over CSPR (eq. 2.5) for topology in fig. 1.1c. N - number of vertices ($N = |V|$) 23

3.1 Fragment of meshed network. Calculating Expected Anypath Transmission Count (EAX) 27
3.2 Example of topology. Mesh network (all links are bi-directional) 34
3.3 BRR in operation (fig. 1.1a). $p(v_r) > p(v_s)$ 39
3.4 Simulation scenario and results 41
3.5 Two-hop multi-relay scenario. $\varepsilon_0 = 0.4,\ \varepsilon_1 = 0$ 42
3.6 Monte-Carlo simulation. m - number of relays; $a_+ = s_+/G_s$ - ratio of useless symbols; $a_- = s_-/G_s$ - ratio of not sent symbols 43
3.7 Fragment of the mesh network. For all shown edges: $\varepsilon_e < 1$. Red dashed edges are not used for calculation of $\rho(v)$ (eq. 3.13) 47
3.8 Test network scenario for feedback importance 53
3.9 Verification of eq. 3.17 using eq. 3.18. $\alpha = \varepsilon^{|L(v_s)|}$ 54
3.10 Test network scenario for feedback estimation 65

3.11 Verification of eq. 3.26 65
3.12 ARQ window. Generation IDentifier (GID) 71
3.13 Transmission window dependence on neighbour ARQ windows. $p(u) > p(v)$. Transmission window of vertex v is shaded 72
3.14 Diamond network with erasure ratios. $d(v) = 1$ Mbps $\forall v \in \{v_s, v_1, v_2\}$ 80
3.15 Loss ratios (topology in fig. 3.14) 81
3.16 Priorities (topology in fig. 3.14) 82
3.17 Coalitions (topology in fig. 3.14) 82
3.18 Filtering probabilities (topology in fig. 3.14) 83
3.19 Long line topology with erasure ratios. $d(v) = 1$ Mbps $\forall v \in V$. 84
3.20 Data rates 84
3.21 Types of sent ANChOR messages 85
3.22 Distribution of Retransmission Request Message (RRM)s 86
3.23 Comparison of the optimal and actual Time Division Multiplexing Access Plan (TDMAP)s 87
3.24 Percentage of RRMs and excessive redundancy (see topology in fig. 3.14) 88
3.25 Data rate uncoded (see topology in fig. 3.14). Changing coding rate 89
3.26 Coded data rate (see topology in fig. 3.14) versus the probability p of attaching RPI to the ANChOR header 90
3.27 Selecting the kind of RR contents (see topology in fig. 3.14) . . 91
3.28 Selection of RR forwarder (see topology in fig. 3.14) 91
3.29 Stability of the output data rate (using eq. 3.34) 93
3.30 Stability of the measured erasure probabilities (using the topology in fig. 3.14) 93
3.31 Stability of the source priority (using eq. 3.35) 94

4.1 Protocol stack. Shaded blocks do not follow G.hn completely [1] 97
4.2 Analysis of Bit-Allocation Algorithm (BAA) [2] on Point-To-Point (Pt2Pt) scenario. Target Packet Error Ratio (PER)=0.4. Corresponding Bit Error Ratio (BER)=0.04695 for $c = 20/21$ (see appendix I) 107

4.3 Adaptive bit loading algorithm from [2]. BER=0.04381 (corresponds to PER=0.1 with 20/21 ideal coder). Example of bit loading for different distances between Tx and Rx vertices (connected with the cable of different length) 108
4.4 Contention Window (CW) size that maximize the MAC efficiency [1] 110
4.5 Multiple access control layer Protocol Data Unit (MPDU) construction in G.hn. $N = 540$ or $N = 120$ bytes 112
4.6 Example: G.hn ARQ; sender side; window (buffer) size - 14 segments 113
4.7 MPDU construction in ANChOR on v_s. $N = 540$ or $N = 120$ bytes 114
4.8 Queueing model of the G.hn Logical Link layer [3] 115

5.1 Measurement results. $P_s = 1400$ bytes; N_a - average traffic burst size; $T(N_a)$ - time to transfer the traffic burst 122
5.2 Simulation. Average MPDU size 124
5.3 Simulation. Data rate on LLC after Cyclic Redundancy Check code (CRC) check 124
5.4 BER and BLock Error Ratio (BLER) characteristics of Quasi-Cyclic Low-Density Parity-Check code (QC-LDPC) Forward Error Correction (FEC) in G.hn using BPSK modulation and FEC block size $M_F = 540$ bytes. Mother codes of check matrices in [4] 127
5.5 ns-3 simulation of Signal to Noise Ratio (SNR). Long line scenario (50 meters between nodes, 39 receiving nodes). G.hn TX Power Spetral Density (PSD) [4]. Worst case scenario of coloured noise from [5] 128
5.6 Access network. Evaluation for the first three vertices in the line with chain topology. 40 modems in total. $ds = 50$ meters. Idealized FEC. $c = 20/12$ 130
5.7 Access network. Evaluation for three vertices in the head-of-the-line. 40 modems in total. $ds = 300$ meters. Idealized FEC. $c = 20/12$ 131

5.8 BLER as a function of SNR. $c = 1/1$. $M_F = 540$ bytes 132
5.9 Performance metrics. No FEC. Packet size 540 byte. Adaptive BAA [2] 132
5.10 Performance metrics. No FEC. Packet size 32 bytes. Adaptive BAA [2] 133
5.11 Performance metrics. No FEC. Packet size 32 bytes. Single modulation index BAA 133
5.12 Performance metrics. No FEC. Packet size 32 bytes. Single modulation index BAA. 36 subcarriers used 134
5.13 ANChOR in G.hn. In-home scenario. Triangular topology . . . 139
5.14 ANChOR in G.hn. In-home scenario. Pt2Pt connection: $\varepsilon_{(v_s,v_d)} = 0.6$. $a = 0$ 141
5.15 ANChOR in G.hn. In-home scenario. Triangular topology: $\varepsilon_{(v_s,v_r)} = 0.1$, $\varepsilon_{(v_s,v_d)} = 0.6$, $\varepsilon_{(v_r,v_d)} = 0.3$. $a = 0$ 142
5.16 ANChOR in G.hn. In-home scenario. Triangular topology: $\varepsilon_{(v_s,v_r)} = 0.1$, $\varepsilon_{(v_s,v_d)} = 0.6$, $\varepsilon_{(v_r,v_d)} = 0.3$. $b = 0.6$ 142
5.17 Triangular topology: $\varepsilon_{(v_s,v_r)} = \varepsilon_{(v_s,v_d)} = \varepsilon_{(v_r,v_d)} = 0.3$. $v_s = v_0$, $v_r = v_1$. $\Delta p = 0$ (eq. 5.6), $b = 0.01$ (eq. 3.31) 143
5.18 Triangular topology: $\varepsilon_{(v_s,v_r)} = \varepsilon_{(v_s,v_d)} = \varepsilon_{(v_r,v_d)} = 0.3$. $v_s = v_0$, $v_r = v_1$. $\Delta p = 1000$ (eq. 5.6), $b = 0.05$ (eq. 3.31) 144
5.19 In-home scenario. Varying G_s, $m = 1$ 146
5.20 In-home scenario. Varying G_s, $m = 1$. Cumulative Distribution Function (CDF)s 147
5.21 In-home scenario. $G_s = 32$, $n = 12$ 148
5.22 In-home scenario. Varying ARQ buffer size, $m = 1$ 149
5.23 In-home scenario. Varying ARQ buffer size, $m = 1$. CDFs . . . 149
5.24 Usage of relays 151
5.25 In-home scenario. $G_s = 32$, $n = 12$ 151
5.26 Influence of m on N_a 152
5.27 In-home scenario. $G_s = 32$, $n = 12$ 152
5.28 In-home scenario. $G_s = 32$, $n = 12$ 153

A.1 1-hop relay network with a direct path 160
A.2 Point-to-Point transmission model 160

B.1 1-hop relay network with a direct path and i.i.d Bernoulli loss process on edges 163

H.1 PHY structure (compare with [4]) Shaded blocks are transparent in the simulator 176

I.1 PER to BER mapping 178

List of Tables

3.1 Notations 59
3.2 ANChOR header structure 78
3.3 Size of ANChOR header parts 79
3.4 Simulation parameters 81

5.1 Overhead parameters. Application layer. Approximated $T(N_a)$ based on the measurement results 122
5.2 G.hn PHY variables influencing O_c and O_p 123
5.3 Overhead parameters. LLC layer in G.hn. Analytical derivation based on [6, 4] 123

Chapter 1

Introduction

PowerLine Communications (PLC) is the promising technology that has already recommended itself as the high-speed in-home communication solution [7] and as the communication solution for electricity metering in access power networks [8, 9]. The main PLC advantage lies in the usage of existing power cables. It substantially reduces the deployment costs. For power utilities, that are responsible for equipment of their networks with communication devices (in the European Union till 2020 [10]), the usage of PLC also reduces the network exploitation costs since they are the owners of this communication medium. Currently, several power utilities are running pilot projects for testing PLC in their typical field scenarios [11, 12, 9]. Millions of PLC modems are already deployed [13, 14].

Despite the success of PLC, this technology still pertains several ideas from the protocols operating over coax or Ethernet cables, which appear to be not optimal in PLC. In this work, we design a routing protocol that allows to achieve the near-optimal performance.

The vital importance in a Routing Protocol (RP) operation plays the link quality metric. Universal Powerline Alliance (UPA) solution (also known as DS2) uses the Spanning tree Routing Protocol (SRP) (described in OPERA specification [15]) that follows IEEE 802.1D. The network master node calculates the routes that have the minimum number of hops for each pair of nodes. Thus, the link quality metric is estimated in the number of hops. This is an easy and optimal solution in case of ethernet networks since the highest communication speed is normally equal for each hop. The amendment IEEE

802.1w defines the Rapid Spanning Tree Protocol that decreases the adaptation time in case of topology change but uses the same link quality metric. In PLC, the quality of single-hop links differs a lot. As a result, sometimes it is better to use a repeater instead of the slow direct link, i.e. the number of hops in PLC does not give sufficient information for the optimal performance.

The Gigabit home network standard (G.hn) [6] defines centralized and distributed modes of the routing protocol operation, which gives a substantial improvement in PLC mesh networks. Due to changes in the PLC channel, the routes should be constantly updated. In the distributed mode, the node computes the Local Routing Table (LRT) itself, which does not require the transfer of the link state information to the master node. In both modes, similar to SRP, G.hn evaluates the link metric as the number of hops.

IEEE1901 [16] calculates the Effective Bit Loading Estimate (EBLE) as the link quality. Each node maintains LRT that contains EBLE values for all neighbors. A given node selects the relay among the neighbors corresponding to the highest EBLE. HomePlug GreenPHY [17] uses the same approach. In literature, EBLE is also referred to as Expected number of Transmissions (ETX) [18]. Since ETX takes into account different quality of Pt2Pt links, that belong to the route, it allows to select best paths more precise.

ETX is also commonly used in wireless networks (e.g. in Ad-Hoc On Demand Distance Vector Routing Protocol (AODV) [19] and The Better Approach To Mobile Adhoc Networking (B.A.T.M.A.N.) [20]). In our research, we show that ETX cannot allow the optimal RP performance in PLC for an arbitrary network topology. Instead, we propose our own metric that better estimates the achievable data rate than ETX.

Most RPs nowadays are using a single chain of repeaters on the way between the given source and destination (Optimized Link State Routing (OLSR), AODV, Dynamic Source Routing Protocol (DSR), B.A.T.M.A.N.). We refer to them as Common Single-path Routing Protocols (CSRPs). The common research topics, that study CSRPs, include improving the effectiveness and speed of adaptation to the changing channel conditions and reducing the overhead for route maintenance. These issues are not the main target of our research. We improve the underlying idea of the route construction. Due to the overhearing in the broadcast PLC channel, several receivers can par-

tially decode the sent message. Opprotunistic Routing Protocols (ORPs) take use of it. They allow to participate in the communication for each node being able to decode a part of the received information. Therefore, it is said that the ORPs use multiple paths even though the single broadcast transmission medium is used. In CSPRs, all nodes not belonging to the route drop the received information. ORPs can smartly use any received information that increases the usage of the channel capacity [21].

In fact, we show in [21] that ORPs approach the limit of the channel capacity that makes them optimal in terms of data rate. Nevertheless, in practice, ORPs have several short-comings. There is a number of implementations nowadays [22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32] that suffer from such disadvantages as a great number of feedback messages, maintenance/inaccuracy of the link quality information, a big amount of duplicates of user data. These disadvantages cast doubts on the feasibility of ORPs to compete with CSPRs.

In [31, 33, 34, 35, 28, 23], it is proposed to use Network Coding (NC) that solves several ORP problems like the excessive feedback. The application of NC in PLC creates new opportunities in the design of routing protocols and Hybrid Automatic ReQuest reply (HARQ) mechanisms [36, 37, 38]. The biggest advantage from the NC application in routing can be expected in fully meshed multi-relay networks with multi-source-multi-destination traffic [39]. In such networks, the vertices have the most opportunities for a *cooperation*.

The focues of this work is a single-source-single-destination scenario with multiple repeaters cooperating with each other. We give a detailed study of this scenario, which can become the basic idea for the multi-source-destination scenario in future works. In our scenario, the improvement is possible only if the physical layer (PHY) is not able to correct all bit errors, which results in packet losses on the layers above.

In fact, the certification tests of PLC devices allow non-zero PHY BLock Error Ratio (BLER). In G3 Narrowband PLC (NB-PLC) standard, the PER limit is at 5% [40]. In another NB-PLC standard, IEEE1901.2, for the minimal input signal of 10 mVrms BLER can comprise up to 10% in robust mode and with PHY payload size of 100 octets. [41, p.152]. In Broadband PLC (BB-PLC), the limit is lower. For example in IEEE1901, the tolerable BLER comprises 0.1% in presence of Additive White Gaussian Noise (AWGN) only,

and 1% in the presence of the "constant sinusoidal interfering signal occupying any single frequency" [16, p.1158]. In the presence of impulsive noise, which is the strongest noise component in PLC [42], BLER can be much higher. Surely, the PHY layer can increase the coding rate and use more robust modulation indices to decrease BLER. But then the PHY layer data rate after decoding can decrease due to the lower sending data rate and the greater coding redundancy. Therefore, instead of trying to decrease the BLER we propose to use it as a favor. Indeed, the presence of block errors can create the channel diversity on the Data Link Layer (DLL). This diversity enables the cooperation in ORPs.

There are also studies of cooperation in PLC without the usage of NC. For example, in [43] the receivers can cooperate on the PHY layer sending only the side information that is not present on other receivers. The same principle is used when NC is implemented on the DLL layer [31, 32, 44, 39]. But there is a significant difference in the type of the channel diversity. In [43], the authors consider the presence of the Additive White Gaussian Noise (AWGN) noise only. In this case, the channel diversity can be achieved through the difference of the Channel Transfer Functions (CTFs) only. Thus, it is derived from the network topology and types of cable. It does not change with time. In [31, 32, 44, 39], the authors do not study the physical channel since NC is implemented on DLL. Instead, it is a logical channel that is often assumed to be a broadcast Packet Erasure Channel (PEC). Its capacity is influenced not only by the AWGN and the CTFs but also the impulsive noise and the PHY layer blocks. Therefore, even when CTFs do not provide a sufficient channel diversity [45] it can be increased when considering the impulsive noise and the PHY layer structure.

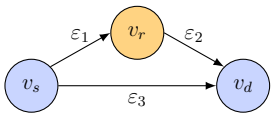
The diversity can be also reduced by the correlation of the block loss processes. In IEEE1901 and G.hn, the PHY layer applies a scrambler on all the PHY layer frame bits before encoding that uniformly distributes the error bits among all PHY blocks and decreases the level of correlation. This motivates us to consider that the block errors at different receivers can be partially uncorrelated. This assumption is also confirmed by the measurement of bit errors simultaneously at several receivers with Frequency Shift Keying (FSK) NB-PLC modems in [38].

We consider the NC implementation on the Logical Link Control layer

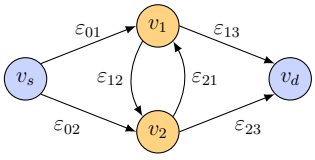
(LLC), which is the part of the DLL layer. In this section, we demonstrate a special type of NC that is applicable for our scenario.

1.1 Intra-flow Network Coding

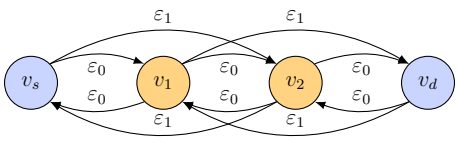
Let the traffic flow on the LLC layer be uniquely denoted with the tuple of the source identifier (ID), destination ID, and the flow ID [6]. We consider the scenario with one source, one destination and only one uni-directional traffic flow between them. Coding the data of the same network flow is called intra-flow NC [39]. We use this type of NC throughout all our work. Nevertheless, NC is commonly explained not with the intra-flow coding. The popular examples are the so called butterfly scenarios [39]. Both wired and wireless "butterflies" consider the presence of two sources and two destinations. Since the data from two pairs of source-destination is coded together, this is also called the inter-flow NC. The wireless butterfly is an example of inter-flow unicast NC and the wired butterfly - the inter-flow multicast NC. Note that the intra-flow NC is a subproblem in the inter-flow unicast and multicast problems. Typical example topologies for intra-flow NC are the triangular and diamond networks (figs. 1.1a, 1.1b). In PLC, they can model the in-home network. We also study the typical PLC scenario in the access power network (fig. 1.1c). With this topology, one can model an electricity metering network or the network of inter-connected smart homes through power cables. In fig. 1.1c, each PLC modem can "talk" to its physically nearest and the second neighbor on the line.



(a) Triangular network



(b) Diamond



(c) Long line

Figure 1.1: Scenarios for intra-flow NC (typical in PLC)

The number of relay vertices can be increased to fit the realistic scenarios.

1.2 Random Linear Network Coding (RLNC)

In the scenarios in figs. 1.1b and 1.1c the network has more than one relay. In fact, the traditional routing protocols may establish the communication route using less number of relays than the physically installed. Using NC, each relay can get cooperation opportunities. From one side, the NC-based routing protocol can be more efficient in terms of throughput favoring to route the data on the fastest path. From another side, it allows more vertices sending the received data. For the MAC, it means a greater effort on channel resource allocation. For the LLC layer it means more effort on the estimation of link quality metrics and the establishment of the routes. Therefore, we target to design a decentralized routing protocol. In this case, the increased control demand can be distributed among all vertices in the network and unload the bottleneck link to the network master. The decentralization becomes possible if Random Linear Network Coding (RLNC) is used.

Consider the example with single-source-destination in a meshed network. We describe the coding with RLNC on the source and on the relaying vertices separately.

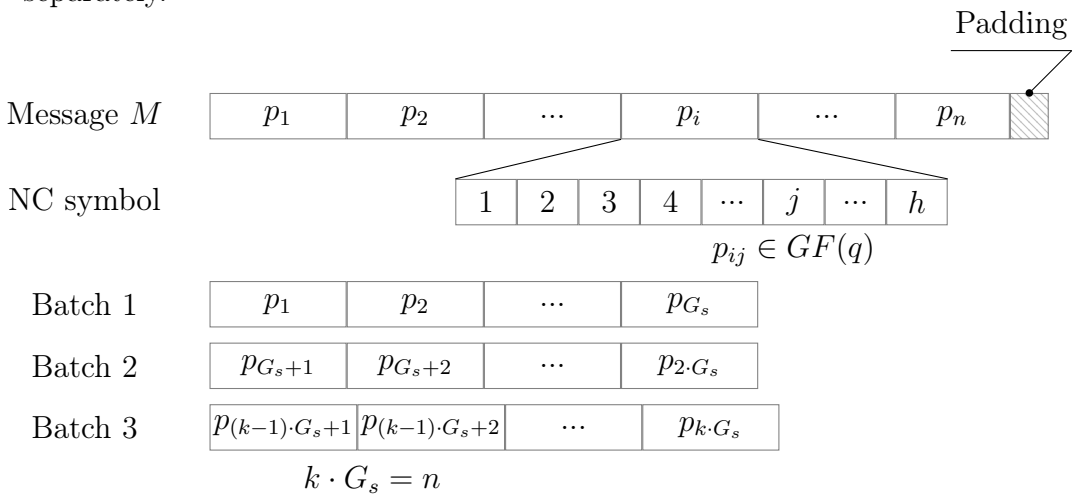


Figure 1.2: Forming generations

The in-coming message M on v_s is segmented into chunks $M = \{p_1, p_2, ..., p_n\}$ of equal length (s bits) that are also called *the original symbols* (see fig. 1.2). The last chunk may be padded to have the same length as all others. The original symbols are grouped into batches $B_i = \{p_{(i-1) \cdot G_s+1}, p_{(i-1) \cdot G_s+2}, ..., p_{i \cdot G_s}\}$, $i \in [1; n/G_s]$ (fig. 1.3), of equal length G_s without overlapping. Such batches

are also called *the generations*. The encoding and decoding operations are performed on each generation separately.

The original symbols are considered as vectors in the Galois field $GF(q)$ with size q. Therefore, $h = s/log_2(q)$ should be integer. Each original symbols can be further divided in smaller chunks of $log_2(q)$ bits: $p_j = \{p_{j1}, p_{j2}, ..., p_{jh}\}$. Each chunk of the original symbol is encoded separately.

In order to produce a Coded Symbol (CS) of the generation i, the source generates the vector $c_{1\times G_s}$ randomly from scalars in $GF(q)$ and multiplies it with B_i. The source can produce an arbitrary number m of CSs p' generating more coding vectors [39]:

$$\begin{pmatrix} p'_{11} & \cdots & p'_{m1} \\ & \ddots & \\ p'_{1h} & \cdots & p'_{mh} \end{pmatrix} = \begin{pmatrix} p_{11} & \cdots & p_{G_s1} \\ & \ddots & \\ p_{1h} & \cdots & p_{G_sh} \end{pmatrix} \cdot \begin{pmatrix} c_{11} & \cdots & c_{1m} \\ & \ddots & \\ c_{G_s1} & \cdots & c_{G_sm} \end{pmatrix} \tag{1.1}$$

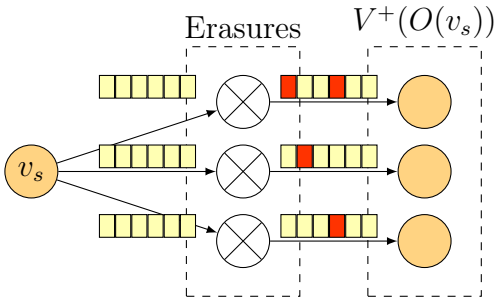


Figure 1.3: Encoding example

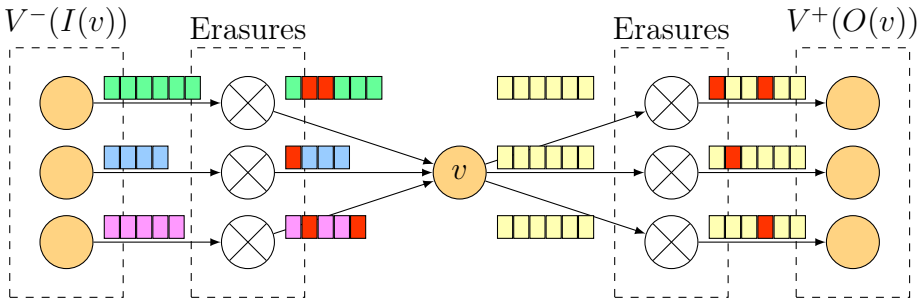


Figure 1.4: Recoding example

Then, the CSs are appended with the coding vectors and broadcasted to all sinks $V^+(O(v_s))$ of the output edges $O(v_s)$ of v_s (see fig. 1.3). We consider the underlying channel to be a Packet Erasure Channel (PEC). Some symbols

are lost due to a random erasure process on each edge (marked with red). Therefore, the vertices in the group of repeaters $V^+(O(v_s))$ receive partially different information.

The input edges of relays connect them either to v_s or to other relays (fig. 1.4). Here, the relay v has a set of input and output edges $I(v)$ and $O(v)$ correspondingly. All received CSs from the same generation are stored in the same coding matrix (from any source vertex of input edges).

Then, the relay recodes the received information producing the Recoded Symbol (RS). They are created in the same fashion as CSs on v_s. Let p'_i be the i-th received CS. It can be also RS received from another relay. The relay generates the recoding vectors $r_{1\times G_s}, ..., r_{m\times G_s}$ randomly from scalars in $GF(q)$ (and completely autonomously). Then, m RSs are produced as follows:

$$\begin{pmatrix} p''_{11} & \cdots & p''_{m1} \\ & \ddots & \\ p''_{1h} & \cdots & p''_{mh} \end{pmatrix} = \begin{pmatrix} p'_{11} & \cdots & p'_{G_s1} \\ & \ddots & \\ p'_{1h} & \cdots & p'_{G_sh} \end{pmatrix} \cdot \begin{pmatrix} r_{11} & \cdots & r_{1m} \\ & \ddots & \\ r_{G_s1} & \cdots & r_{G_sm} \end{pmatrix} \tag{1.2}$$

In difference to the coding on the source, p'_i contains not only the payload but also the coding vector. And, the relay does not attach the recoding vector to RSs. Attaching the recoding vector is not necessary since p''_i already contains the original coding vector multiplied by the recoding vector. Thus, the size of p'_i equals the size of p''_j, i.e, the recoding is possible at zero cost of overhead.

The recoding is a special feature of NC. It is not common to rate-based codes like Reed-Solomon or Low-Density Parity-Check code (LDPC) and to rateless codes like Luby Transform code (LT) or RAPid TORnado code (RAPTOR). Basically, it allows to produce the coded packets without decoding the original message. It is sufficient to receive one CS to start recoding. Also, it allows to combine the received CSs from different vertices because all CSs and RSs are obtained with linear operations on the original symbols and thus describe the same vector space.

As far as all CSs have the corresponding coding vectors in their headers, the destination can construct the coding matrix and solve the system of linear equations for the original symbols. Surely, the destination should previously collect a sufficient number of symbols so that the rank of the coding matrix equals G_s.

Degree Of Freedom (DOF)

The Degree of freedom in RLNC is used to denote the dimension of the space spanned by a certain set of vectors. The number of DOF equals the rank of the coding matrix. But DOF gives more information than the rank only since it is possible to find out the space planes defined with the matrix. For example, two matrices may have the same rank but differ in the defined space planes. Hence, the rank of the concatated matrix can be greater than of any two initial matrices.

Linear dependent coding vectors

An advantage of RLNC lies in absence of a central coordinator distributing the network codes. But it has also a short-coming. There exists a non-zero probability of linear dependence between the generated coding vectors. It means that for certain two vectors c_1 and c_2 there may exist a certain scalar a so that $c_1 = a \cdot c_2$. Thus, c_2 belongs to the vector space spanned by c_1. It means that any vertex receiving both c_1 and c_2 can increase the rank of its decoding matrix at most by one, i.e. one of two coding vectors is useless. In some cases, the transmission of linear dependent CSs can be avoided. Let vertex v generate $c_1, c_2, ..., c_{n+1}$. Before sending the CS with c_i, it can add it to the matrix containing $c_1, c_2, ..., c_{i-1}$. If the rank of the matrix increases then c_i is not linear dependent with $c_1, c_2, ..., c_{i-1}$. It is also called *innovative*. But in many other cases, the problem of linear dependence cannot be avoided with such small effort. Assume v broadcasts the set of innovative vectors $C_v^{Tx} = c_1, c_2, ..., c_{n+1}$ and the vertices u and w receive subsets $C_u^{Rx}, C_w^{Rx} \subseteq C_v^{Tx}$ correspondingly. Then, u and w recode the received vectors and broadcast the vectors from the sets C_u^{Tx} and C_w^{Tx}. Both u and w can guarantee that C_u^{Tx} and C_w^{Tx} contain only innovative symbols in the same fashion as v. But without an extensive coordination between them, they cannot guarantee that all vectors in $C_u^{Tx} \cup C_w^{Tx}$ are linear independent. This problem is especially intensified with ORPs because of the increased number of potential senders. We try to overcome it in the design of our ORP.

Any vertex can also intendedly add linear dependent symbols for coding redundancy. Thus, any vertex can overestimate the harshness of the commu-

nication channel adding the excessive redundancy.

On-the-fly RLNC

The classical RLNC is a block code. It issues a buffering and decoding delay, which can increase the latency and jitter of symbols above the layer with NC implementation in comparison to the case without coding. The destination sends the symbols from NC layer to the layer above in batches equal to the generation size. This increases the jitter. In addition, the source waits to have sufficient original symbols before encoding (equal to generation size), which increases the latency. Both problems can be reduced if the on-the-fly RLNC is used. It is also known as online NC [46]. In this case, the source can encode the original symbols with any rank of the encoder matrix, i.e. as often as it would be possible to send the symbols without coding. The destination is capable of doing the partial Gaussian elimination. It allows to send some decoded symbols before the rank of the decoder matrix is complete.

In [46], the on-the-fly RLNC is compared to the classical RLNC and Reed-Solomon code on the fair example with no recoding. The gain of the on-the-fly RLNC in in-order delay (latency) [37, 36] reaches an order of magnitude for certain loss ratios.

Systematic RLNC

The systematic RLNC performs similar to any systematic code. The payload of the systematic CSs contains the uncoded original symbol. It is advantages to be used for such reasons as:

- reducing the computational demand (the systematically coded symbols require no Gaussian elimination);
- decreasing the number of linearly dependent CSs (the systematically coded symbols require no random selection of the coding vector);
- decreasing the in-order delay ([37, 36]).

In [37, 36], it is proposed to produce non-systematically coded symbols only for redundancy. The coding vectors of redundant symbols are obtained randomly from the certain field and contain the zero elements only as a result of the

random process. Such coding is also known as *full-vector* coding. And, it is a traditional coding approach. If G_s is the generation size and k is the number of redundant symbols then the resultant coding rate equals $c = G_s/(k + G_s)$. Let $\mathbf{p}$ be a vector of CSs with size $(k+G_s)$. The authors in [37, 36] studied the placement of the full-vector coded symbols inside of $\mathbf{p}$. They compared the in-order delay for two placement strategies: put all k symbols at the end of $\mathbf{p}$ or distribute k symbols evenly in $\mathbf{p}$ with a certain step $l = (k + G_s/k)$. Since l must be an integer value the latter strategy reduces the number of possible coding rates to the following: $c = (l-1)/l,\ l \in Z$. But the in-order delay can be much improved when on-the-fly RLNC is used.

Computational requirements

For a long time, NC has remained infeasible due to high computational cost. Recent studies, e.g. [47], show several of practical techniques targeted to reduce the number of computations. They allowed to implement a library for NC in C++. A detailed description can be found online [48]. It allows to use NC not only on personal computers but also on low computation power processors like ARM Cortex M0, ATmega328, etc.

Fortunately, it is also possible to scale the required computational power changing the field and generation size.

Chapter 2

Performance Limits of Routing Protocols in PLC

The performance of a Routing Protocol (RP) is usually optimized w.r.t. the targeted performance metric. We concentrate on the throughput. The highest achievable data rate is upper-bounded by the capacity of the underlying channel. In this section, we analyze the channel properties and evaluate the upper bound on the data rate. Eventually, we propose the metric for evaluation of RP performance.

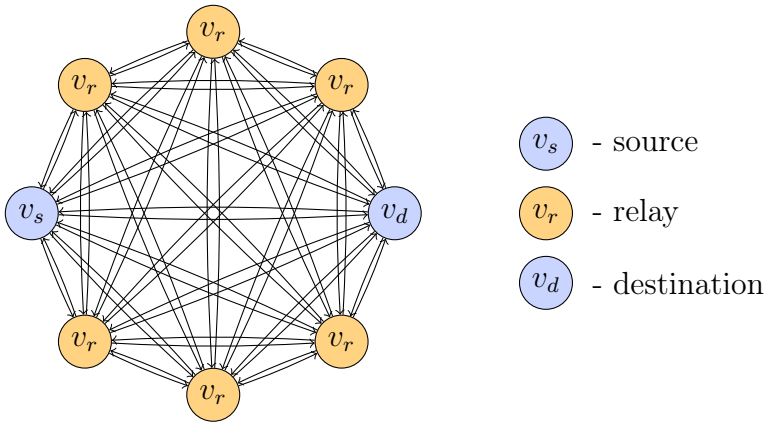


Figure 2.1: Fully meshed network. Multi-relay scenario

In most PLC protocols [6, 7, 15, 16, 41, 14], the channel below the routing protocol is the packet erasure channel with Time Division Multiplexing (TDM). This section gives a general discussion on such channel. Hence, the obtained results can be also applied in wireless technologies with the similar channel,

e.g. wifi, LTE, etc.

2.1 System model

We consider a single-source-destination routing problem in the meshed network (fig. 2.1). The source vertex v_s and the destination vertex v_d are connected with the directed network graph $\mathcal{G} = \{V, E\}$, where V ($v_s, v_d \in V$) is the set of vertices and E is the set of edges. Each edge $e \in E$ can be uniquely identified by its source $v^-(e)$ and sink $v^+(e)$ vertices. If $v^-(e) = v_i$ and $v^+(e) = v_j$, $v_i, v_j \in V$, then $e = (v_i, v_j)$. The sets of source and sink vertices of the edge set Z are denoted as $V^-(Z)$ and $V^+(Z)$ correspondingly. The sets of edges $I(v)$ and $O(v)$ denote the in- and out-coming edges of the vertex v. We consider no Frequency Division Multiple Access (FDMA) and no Multiple-Input Multiple-Output (MIMO) communication. Therefore, each pair of vertices is connected with only one edge per direction ($I(v_i) \cap O(v_j) = \{(v_i, v_j)\}$). We also define $|S|$ as the cardinality of set S (set of edges or set of vertices).

2.2 Channel model

We consider the implementation of the RP either between Network and Transport layers or between Data Link (DLL) and Physical (PHY) layers. In any case, the RP does not have a direct logical connection to the physical medium. Thus, we consider the channel consisting of the logical links between vertices. Still, it pertains some characteristics of the PLC channel.

Here, we consider that RP communicates packet data units (PDUs) appended with a Cyclic Redundancy Code (CRC) (e.g., Logical Link Layer in G.9961 ITU standard [6]). These PDUs may be further encoded on PHY with Reed-Solomon, Low-Density-Parity-Check (LDPC) or other codes for Forward Error Correction (FEC). We do not study the coding on PHY. Instead, we assume that the considered PDUs may have some bit errors after FEC decoding, which can be reliably detected (though not corrected) by CRC. Therefore, we study the Packet Erasure Channel (PEC). Let vertex $v_i \in V$ communicate a message M to vertex $v_j \in V$ through the channel consisting of the single edge $e = (v_i, v_j)$. It encodes M to X and v_j observes Y on its channel output. The codeword X corresponds to a packet of l bits with alphabet $GF(2^l)$ for $l \gg 1$ and $Y \in GF(2^l) \cup \{\epsilon\}$, where ϵ denotes an erasure. Then, the channel

transfer function (CTF) between v_i and v_j can be defined with the probability distribution $p(Y|X)$ as follows:

$$\begin{cases} p(Y = x|X = x) = 1 - \varepsilon_e \\ p(Y = \epsilon|X = x) = \varepsilon_e, \end{cases} \tag{2.1}$$

where $\varepsilon_e \in [0, 1]$ is the PDU loss ratio on the edge e. Note that the above equation implies $p(Y \neq x, Y \neq \epsilon|X = x) = 0$, i.e. the error can be reliably detected.

Let $X^n = \{X_0, X_1, ..., X_{n-1}\}$ be the sequence of codewords emitted by v_i. If the colored noise component dominates among other noise types then $p(Y|X)$ can be considered *time in-variant* implying $p(Y_i|X_i) = p(Y_j|X_j)\ \forall\, i, j \in [0, n-1]$. Although we admit that the PLC noise can have a strong cyclo-stationary component [49], for simplicity, we do not consider the time correlation of $p(Y|X)$. We also assume the channel to be memoryless in sense that Y_i depends on the past $\{X_0, X_1, ..., X_{i-1}\}$ only through the present transmitted symbol X_i.

If E consists of more than one edge, the upper bound on the communication data rate between v_s and v_d can be increased through the cooperation of relay vertices $v \in V \setminus \{v_s, v_d\}$ [50]. In this work, we use TDM with Time Division Duplexing (TDD) for channel resource sharing because it is the most widely used technique in PLC [6, 16, 17, 15]. But it implies certain limits on cooperation policies between vertices.

With the use of TDM/TDD, due to the *broadcast* nature of the PLC channel, the neighbors of the given sending node should remain in the reception state until the current transmission ends. In the fully meshed network, all nodes are the neighbors to each other in the sense that each pair of nodes is connected with an edge. Without loss of generality, any PLC network can be described as the fully meshed network with arbitrary small $\varepsilon_e \in [0, 1]\ \forall e \in E$. We assume that for any sending node $v \in V$, the reception PSD at the channel output of any other node $u \in V \setminus v$ is sufficiently high to recognize that the node v is in the sending state. It implies the presence of only one transmitting node at any time.

Summarizing the channel characteristics, it can be defined as follows.

Definition 2.2.1. The communication channel between v_i and v_j, $v_i, v_j \in V$,

is a discrete memoryless packet erasure broadcast channel with TDM and without time correlation of Channel Transfer Function (CTF)s.

2.3 Upper bound on the achievable data rate

The tight upper bound on the achievable data rate in the communication channel (definition 2.2.1) should consider all channel properties.

The first description of the relay channel model was given by Meulen in his Ph.D. thesis in 1968. In [51] he presents the theoretical elaborations from the two-way channel firstly described by Shannon to the relay channel model. This work was much extended by Cover and Gamal in [52]. In both works, the cut-set bound theorem [53] is used to evaluated the upper bound on the achievable data rate. Since we consider TDM/TDD, the cut-set bound is not tight enough. Instead, we use [54, Theorem 1]. A similar approach is demonstrated in [55].

The information rates in this section are given in packets per channel use. Therefore, the logarithm appearing in the expression of mutual information is to the base of 2^l, where l is the size of the codeword in bits. For notational convenience, we label the vertices in V by $0, \ldots, |V|-1$, whereby $v_s = v_0$ and $v_d = v_{|V|-1}$. We also specify a set of transmitters $V_s = V \setminus \{v_d\}$.

We previously assumed that all codewords have equal size. Now, let the communication period be divided in time slots of such duration that the transmission of one codeword by certain node $v \in V_s$ takes exactly one time slot. Due to possibility of different sending data rates of nodes in V_s, the time slot duration for each node may differ. With TDM/TDD, only one node can use each time slot. Let the vertex v uses K_v time slots with indices $0, 1, ..., k, ..., K_v - 1$. Then, the k-th slot used by v can be denoted as t_v^k. Let $t_v = \sum_{k \in [0, K_v)} t_v^k$. For convinience, we normalize all values t_v and define the TDM access plan as $T = \{t_0, \ldots, t_{|V_s|-1} \in [0,1]^{|V_s|} : \sum_{v \in V_s} t_v = 1\}$. Let the channel input of vertex v be X_v. And, let the channel output of vertex $u \in V \setminus v$ in any time slot, when v is in the sending state, be denoted by $Y_u^{(v)} = E\epsilon + (1-E)X_v$, where E is a binary random variable with $\Pr\{E = 1\} = \varepsilon_{(v,u)}$, $\Pr\{E = 0\} = 1 - \varepsilon_{(v,u)}$ and ϵ means the erasure. Then, every achievable rate R must satisfy

$$R \leq \sum_{v \in S} t_v \cdot I(X_v; Y_{S^C}^{(v)}) \tag{2.2}$$

for some joint probability mass function $p(x_1, \ldots, x_{|V_s|})$ and for every cut $S \in \mathcal{C} = \{S \subseteq V_s : v_s \in S\}$, and $S^C = V \setminus S$. This statement is similar to [54, Theorem 1]. In the special case when only one node $v \in V$ can be in a sending state at a time, and there is only one source node, the equation from [54, Theorem 1] can be simplified to eq. 2.2.

Thus, the upper bound on R is the solution to the following linear optimization problem [21]:

$$\begin{aligned} \max_{R,t_1,\ldots,t_{|V_s|}} \quad & R \\ \text{s. t.} \quad & R \leq \sum_{v \in S} t_v \cdot I(X_v; Y_{S^C}^{(v)}), \quad \forall S \in \mathcal{C}, \\ & (t_1, \ldots, t_{|V_s|}) \in T. \end{aligned} \tag{2.3}$$

If there is no spacial correlation, the codewords $Y_u^{(v)}$, $u \in S^C$, are conditionally independent. Given X_v, it is straightforward to show that $I(X_v; \mathbf{Y}_{S^C}^{(v)}) \leq 1 - \prod_{u \in S^C} \varepsilon_{(v,u)}$ (see [56, corrolary 2] for $K = 1$). In presence of spacial correlation, the capacity of each cut S decreases, which can be considered in the evaluation of $I(X_v; \mathbf{Y}_{S^C}^{(v)})$ if the correlation function is known.

Notice that the same TDM access plan can be realized with both Time Division Multiple Access (TDMA) and Carrier Sense Multiple Access (CSMA). In the former case, v may access the channel K_v time slots in a row, while in the latter case, the nodes get the channel access randomly after each time slot. The upper bound given by eq. 2.3 is the same in both cases.

Note that the solution for the optimization problem in eq. 2.3 consists of R and T, i.e. there is a certain TDM access plan maximizing R.

In the corrolaries of appendix A, we evaluate R for the triangular topology analytically. There are two possible solution depending on the quality of the edges (v_r, v_d) and (v_s, v_d). Obviously, the relay does not improve R if the connection quality between v_r and v_d is worse than between v_s and v_d (see eq. A.3).

2.4 Achieving the upper bound data rate

In this section, we analyze the ability of certain RPs to reach R as defined by eq. 2.3.

We basically distinguish between two big groups of RPs. The first group,

Common Single-path Routing Protocols (CSRPs), uses the "static routing" in the sense that each relaying node knows the next relaying node before sending the packet and the relays form a chain. Among them, there are OLSR [57], AODV [19], DSR [58], B.A.T.M.A.N. [20]. Another group uses Opportunistic Routing Protocols (ORPs). Here, the relaying node selects a group of relaying nodes Z instead of one next relay that results in a mesh network. Due to the packet erasures, some selected relays $u \in Y,\ Y \subseteq Z$, may not get the packet. Then, the nodes $v \in \{Z \setminus Y\}$ decide between each other who relays the received packet. Among ORPs there are such protocols as Extremely Opportunistic Routing (ExOR) [59], MAC-independent Opportunistic Routing & Encoding (MORE) [31], Simple Opportunistic Adaptive Routing protocol for wireless mesh networks (SOAR) [32], efficient network coding based opportunistic routing through Cumulative Coded Acknowledgments (CCACK) [30], etc.

As mentioned above, CSPRs do not use all vertices in the network to route the data. Instead, they define a chain of vertices $Q = \{v_0, v_1, ...\}$, $Q \subseteq V,\ v_s, v_d \in Q$ forming the route between certain source and destination vertex. As a result, the number of possible cuts $|\mathcal{C}|$ is reduced from $|V| - 1$ to $|Q| - 1$. In addition, with CSPR, the codeword from v_i can be relayed only by v_{i+1} even though any $v_j \in Q,\ j \neq i + 1$ may receive it, i.e., not all edges connecting the vertices in Q are allowed to be used. Considering this limitations, there exist a tighter upper bound on the achievable data rate with CSPR than given by eq. 2.3. For its evaluation, we can use eq. 2.3 with minor changes. We redefine the set of possible cuts $\mathcal{C} = \{S \subseteq V_s : v_s \in S\},\ V_s = Q \setminus v_d$ and consider the edge set $E = \{(v_i, v_{i+1} :\ v_i \in V_s)\}$ for calculation of the mutual information:

$$I(X_{v_j}; \mathbf{Y}_{S^C}^{(v_j)}) = \begin{cases} I(X_{v_j}; Y_{v_l}^{(v_j)}) & \text{if } l = j + 1 \\ 0 & \text{otherwise} \end{cases},\ \forall j, l \in [0, |Q| - 1],\ S \in \mathcal{C} \quad (2.4)$$

Note that ORPs use all nodes and edges in $\mathcal{G}$. Thus, eq. 2.3 gives the tight upper bound for this group of RPs.

It is clear that the upper bound on the achievable data rate with ORPs is higher than with CSPRs.

2.5 Potential gain of ORP over CSPR

Here, we evaluate the potential gain of ORP over CSPR. As far as we optimize RP for throughput, we use the upper bound on the achievable data rate for evaluation of its performance:

$$\alpha = (R - R')/R, \tag{2.5}$$

where R is the ORP upper bound calculated with eq. 2.3 and R' is the CSPR upper bound calculated either with eq. 2.4. Another common approach for R' calculation is Dijkstra's algorithm.

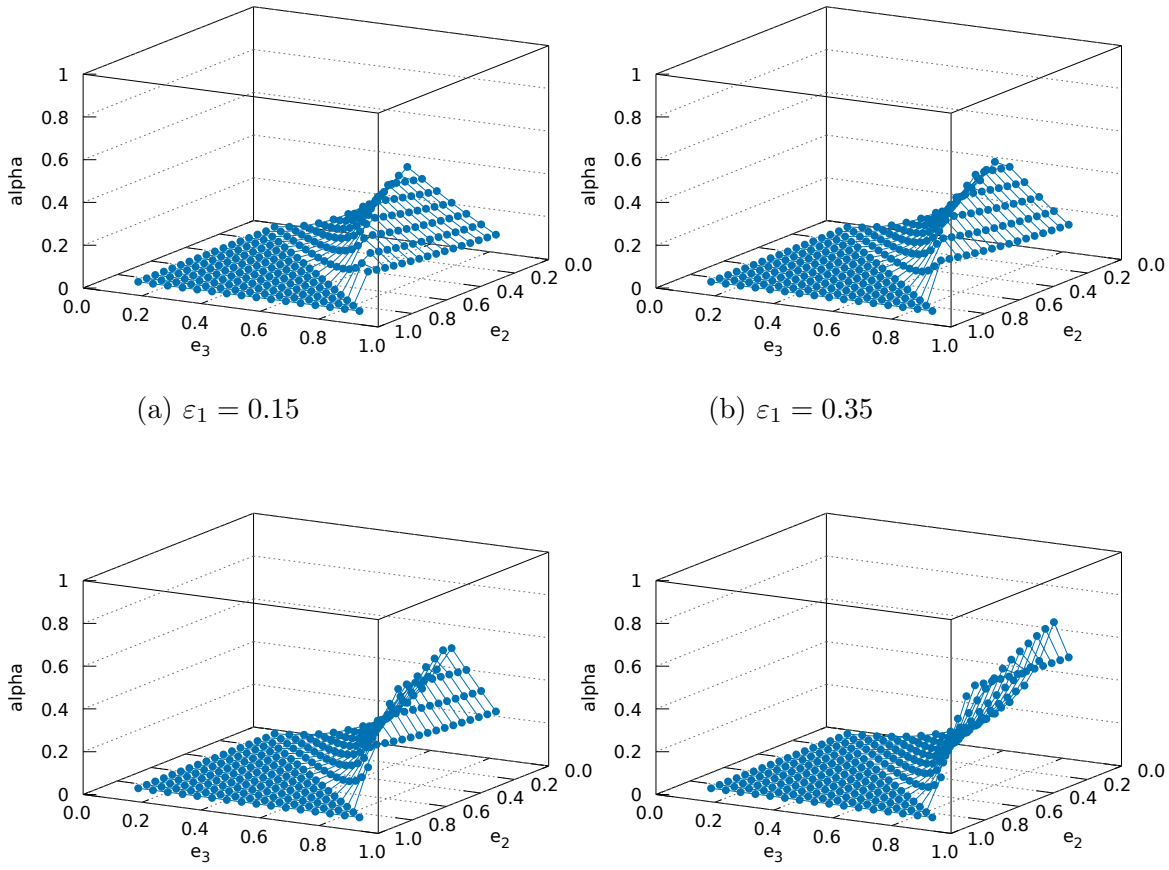
In appendix B, we evaluate α for the triangular topology analytically. It appears that the maximum α can be achieved if the direct path and the path with the relay have the same quality in terms of the effective data rate.

To the author's best knowledge, there are just a few works on ORP in PLC [38], [3], [60]. The first two do not deal with the evaluation of the ORP potential gain. In [60], the ORP potential gain for certain topology type and without optimality proof is presented. Here, we propose the approach suitable for an arbitrary topology. Moreover, eq. 2.5 gives the potential gain using the optimal (highest achievable) data rate.

2.6 Evaluation of ORP potential

Using the C++ based solver of linear programming optimization tasks [61], we calculate the upper bound on the achievable data rate R (eq. 2.3) with ORP. Then, we calculate the upper bound on the achievable data rate R' with a CSPR using Dijkstra's algorithm. The potential gain of ORP over CSPR, α (eq. 2.5), for triangular topology (fig. 1.1a) is shown in figs. 2.2a - 2.2d.

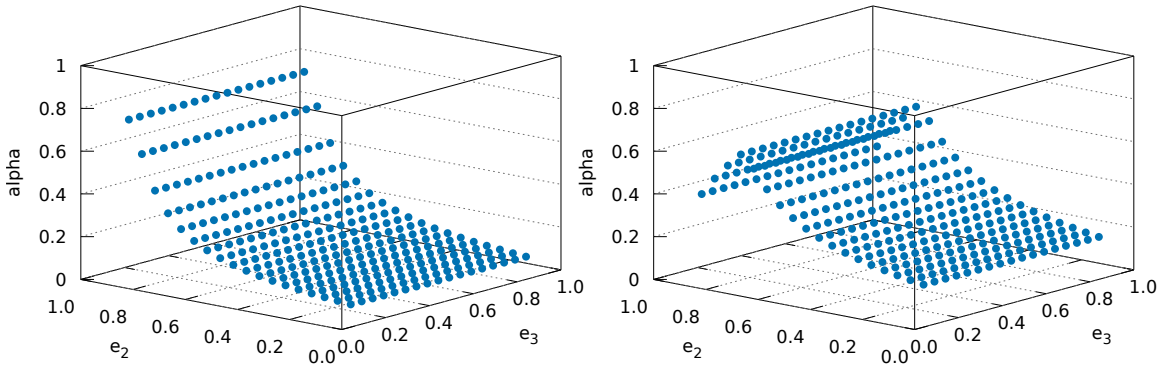
These results were obtained under the assumption of no spatial loss correlation and the same sending data rate d of both source and relay. It is possible to distinguish three areas with a different tendency, which also coincides with the analytic expression for α given in eq. B.3. The area with zero gain corresponds to the case when on the edge (v_r, v_d) there are more losses than on the edge (v_s, v_d). It means that the relay cannot improve the throughput if it has a worse connection to the destination than the source. We use this experience later in the design of the routing rules. In figs. 2.2a - 2.2d, there is also a clear maximum. It corresponds to those combinations of ε_1, ε_2, ε_3 that satisfy the



(a) $\varepsilon_1 = 0.15$ (b) $\varepsilon_1 = 0.35$

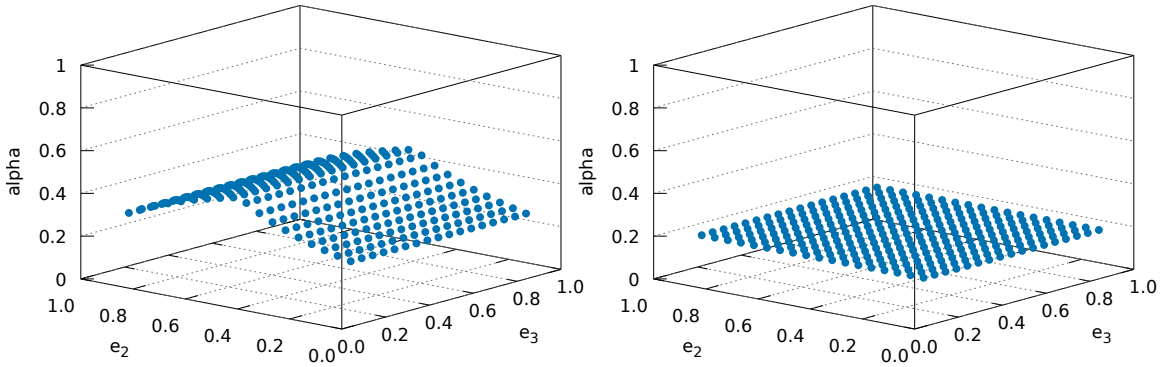
(c) $\varepsilon_1 = 0.55$ (d) $\varepsilon_1 = 0.75$

Figure 2.2: Potential gain α (alpha) of ORP over CSPR (eq. 2.5) for topology in fig. 1.1a



(a) $\varepsilon_1 = 0.1$

(b) $\varepsilon_1 = 0.2$



(c) $\varepsilon_1 = 0.3$

(d) $\varepsilon_1 = 0.5$

Figure 2.3: Potential gain α (alpha) of ORP over CSPR (eq. 2.5) for topology in fig. 2.4

following condition:

$$1 - \varepsilon_3 = \frac{(1-\varepsilon_1)\cdot(1-\varepsilon_2)}{2-\varepsilon_1-\varepsilon_2}. \tag{2.6}$$

It is easy to observe that this condition fulfills when the direct path and the path over the relay are equally good. Say, v_s targets to send the message with size M to v_d with the help of CSPR. If it uses the direct path then the effective reception data rate at v_d equals $d \cdot (1-\varepsilon_3)$. If v_s uses the path with the relay then the effective data rate can be calculated as follows:

$$R' = \frac{M}{\frac{M}{d\cdot(1-\varepsilon_1)} + \frac{M}{d\cdot(1-\varepsilon_2)}} = d \cdot \frac{(1-\varepsilon_1)\cdot(1-\varepsilon_2)}{2-\varepsilon_1-\varepsilon_2}.$$

In figs. 2.3a - 2.3c, we show the potential gain of ORP over CSPR on the example of the diamond network topology (fig. 1.1b). Here, we change the loss ratios between relays only. In this way, we show the influence of asymmetry of the communication channel, which is typical for PLC.

First, we notice that the value of ε_3 does not influence α. It is reasoned by that for the given range of $\varepsilon_1 \in [0.1, 0.5]$ the relay v_2 always has better connection to the destination than v_1. As a result, v_1 should drop all data received from v_2 because v_2 can deliver it faster to v_d than v_1. Second, the gain does not change for any $\varepsilon_2, \varepsilon_3 \in [0.05, 0.95]$ if $\varepsilon_1 = 0.5$. It has an analog reason. When v_1 and v_2 have equally good connection to v_d, they cannot improve the upper bound on the achievable data rate between v_s and v_d through cooperation.

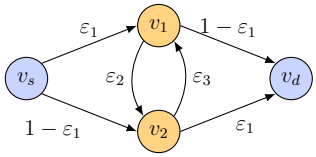
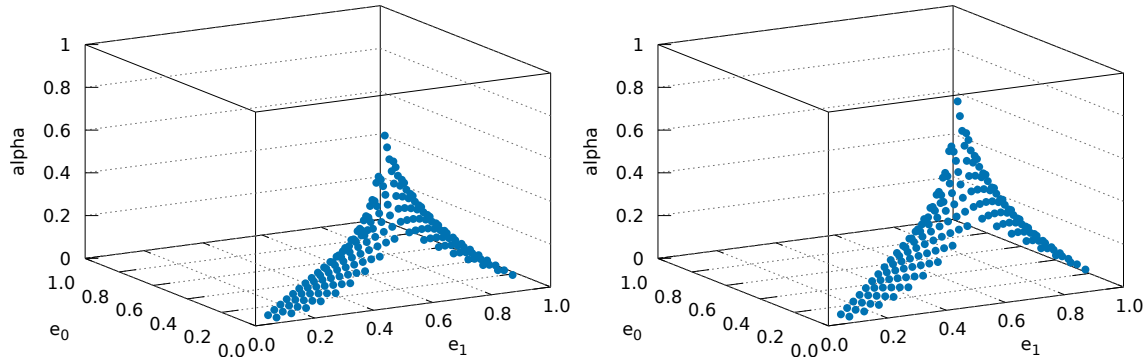


Figure 2.4: Diamond topology. Special case

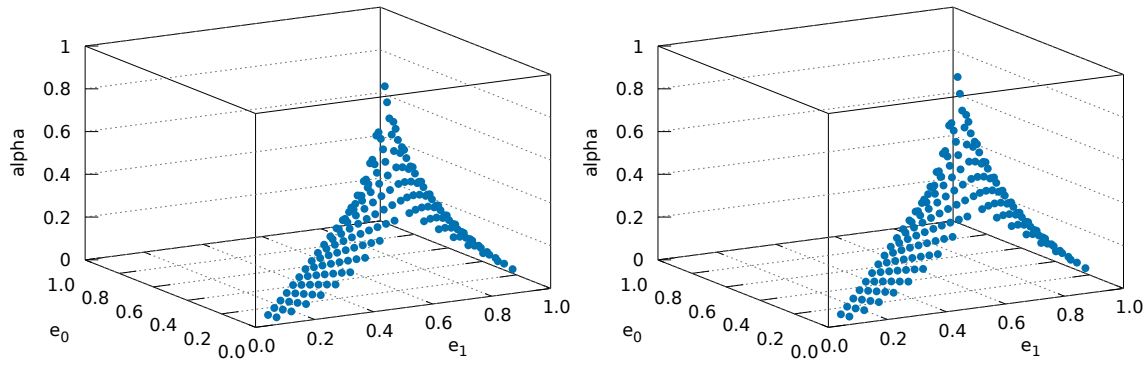
Finally, we calculate the gain α for the topology in fig. 1.1c. Here, we vary not only the loss ratios but also the length of the chain of vertices N (figs. 2.5a - 2.5d).

In order to staying with the realistic model of the long electrical cable, we omit the points when $\varepsilon_0 > \varepsilon_1$. From figs. 2.5a - 2.5d, it is easy to observe that the upper bound on the achievable gain grows when the number of vertices increases. Thus, the bigger the PLC access network is the higher gain can be achieved.



(a) $N = 3$

(b) $N = 5$



(c) $N = 7$

(d) $N = 9$

Figure 2.5: Potential gain α (alpha) of ORP over CSPR (eq. 2.5) for topology in fig. 1.1c. N - number of vertices ($N = |V|$)

Chapter 3

Opportunistic Routing: Realizations and Challenges

CSPRs were adopted in PLC and wireless networks from wired technologies that created artificial communication limits. For example, CSPRs do not exploit the overhearing in the PLC channel. Also, they degrade the performance in presence of packet losses. Therefore, they target to use resilient routes. The resilience is achieved through the selection of more robust modulation and coding schemes that reduces the throughput. On contrary, ORPs use the diversity and harshness of the communication environment as a fortune. They do not have to rely on the resilience of one particular route because they can employ a number of them. Thus, ORPs can also achieve the resilience through the usage of the PLC channel diversity.

Another disadvantage of CSPRs lies in neccessity of the constant maintenance of existing routes. Even such protocols as AODV [19] and B.A.T.M.A.N. [20] that do not transport the routing information to the source or some central vertex but store it locally, suffer from the same problem as well. ORPs do not need each of the selected routes to be highly resilient. Therefore, they are not so sensible to changes of the channel quality.

These advantages of ORPs in comparison to CSPRs were expressed in the previous chapter in terms of the upper bound on the achievable data rate. Evidently, CSPRs do not use the full potential of the PLC channel. Inspired by these results, we investigate ORPs in detail.

For the first time, the idea of opportunistic routing was proposed in a struc-

tured form in ExOR [59]. It was applied for sensor networks and relied on a region-based routing. In accordance to ExOR, the sender specifies a certain distance range and the receiver vertices falling within this range are permitted to forward the received information. Hence, the routing rules in [59] are based on geographical information. In PLC, the actual lengths of power cables do not play a major role in evaluation of CTFs. Therefore, all types of geographical routing cannot achieve the best performance. Nevertheless, the proposed structure of ORP in ExOR can be reused. As also mentioned in [62] and [63], this structure was considered in multiple following protocols like SOAR [32], MORE [31], CodeCast [23], etc. Thus, it creates a development benchmark for ORPs nowadays. We summarize the set of tasks for ORPs as follows:

- each relay/source vertex selects a group of relays eligible to forward the received data;
- each relay/source vertex calculates the coding redundancy;
- upon reception, the selected vertices decide, who should forward the received data;
- acknowledging and retransmissions;
- congestion control.

We analyze this structure step by step in the following sections. First, we introduce the routing metric, the vertex priority, that allows optimizing the performance of ORP for the throughput. The calculation of this metric is tightly connected with the selection of the cooperating candidates. Note that with CSPRs each vertex communicates with one of its neighbors only. With ORP, each vertex selects a subset of all neighbor vertices that can consist of several vertices. Such subset we call the *cooperation group*. Then, we design the Basic Routing Rules (BRR) for ORP that pertain certain idealized assumptions but allow approaching the upper bound on the achievable data rate in the given network, i.e. the optimal data rate. Then, we modify BRR to loosen the theoretical assumptions. Here, we show that using Network Coding it is possible to achieve the near optimal data rate. In order to guarantee the communication reliability, we implement Automatic ReQeust repeat (ARQ)

mechanism. We show that the known ARQs are not applicable with ORP using NC. First, we analyse available solutions and also propose one. In order to avoid buffer overflows, we also implement the congestion control algorithm. Both ARQ and the congestion control are designed keeping the minimization of the protocol overhead (headers and feedbacks) in mind. In order to enhance the stability of the developed protocol, we also extend the procedure in the formation of the cooperation groups and introduce the hysteresis for the comparison operator of the routing metric. The performance analysis of the developed protocol we demonstrate with the simulation results.

3.1 Vertex priority and cooperation group

In each part of the ORP structure, the vertices use a metric to distinguish their importance in comparison to each other. This is also called the *vertex priority*. This metric has an analog importance to the link quality estimator in CSPRs. There are multiple expressions for this metric depending on the field of ORP application. We mention here only those, which base on calculation of the throughput and consider the presence of multiple routes. The examples of such metric are EAX [18] (EATX in [64]), Expected Anypath Transmission Time (EATT) [64]. In our work, we use the metric for the vertex priority that is similar to EAX and EATT. Therefore, we describe them in detail. Here, we use the term of the vertex priority to address both EAX and EATT.

In a tight connection to the vertex priority lies the definition of the cooperation group. Each vertex $v \in V$ selects a group of candidate relays to forward the data received from v. Such subset of vertices we call a cooperation group. In literature it is also known as the forwarders' list, the list of candidate relay or the coalition of vertex v. It can be defined as follows.

Definition 3.1.1. The cooperation group of vertex v is such subset of vertices $L(v) \subset V^+(O(v))$ so that $\forall u \in L(v)$ the priority of v is less than the priority of u. The vertices in $L(v)$ are sorted in descending or ascending order depending on the priority definition. If v_d has the highest (resp. smallest) priority then $u \in L(v)$ are sorted in descending (resp. ascending) order of their priorities.

Analog definitions of the cooperation group can be found in [31, 18, 64, 26, 25], etc.

In [18], the authors define the vertex priority as EAX. Here, $EAX(v, v_d)$

means EAX of vertex v that targets its data to the destination v_d. It can be calculated as follows:

$$EAX(v, v_d) = \frac{1 + \sum\limits_{u \in L(v)} EAX(u, v_d) \cdot p_{(v,u)} \cdot \prod\limits_{u' \in M(u,v)} (1 - p_{(v,u')})}{1 - \prod_{u \in L(v)} (1 - p_{(v,u)})}, \qquad (3.1)$$

where $M(u, v) \subset L(v)$ is such a list of vertices that the priority of each vertex in $M(u, v)$ is higher than the priority of vertex u and $p_{(v,u)}$ is the probability of symbol delivery on the edge $e = (v, u)$.

The evaluation of EAX can be better explained with the help of fig. 3.1. We use the notation v_i to identify the node from this figure and v for the node in general. This figure shows a fragment of an arbitrary network. Each vertex v has a set of input $I(v)$ and output edges $O(v)$. In general case, these sets may intersect. Here, the vertex v_i can receive data from vertices $V^-(I(v_i))$.

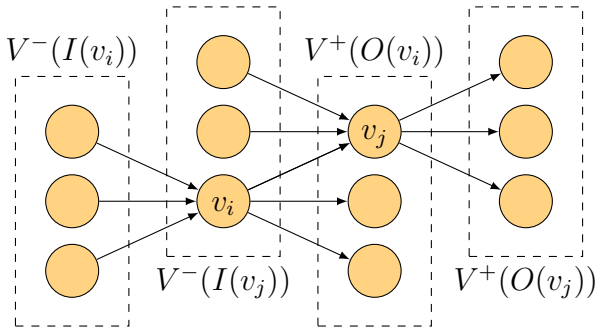


Figure 3.1: Fragment of meshed network. Calculating EAX

Out of the sink vertices of its output edges, it selects the cooperation group $L(v_i) \subseteq V^+(O(v_i))$. This is possible with prior knowledge of the priorities of each vertex in $V^+(O(v_i))$. The vertex v_i also finds out the packet loss ratios for each edge $e = (v_i, u)$ $\forall u \in L(v_i)$ and $EAX(u, v_d)$ $\forall u \in L(v_i)$. Then, it applies eq. 3.1 to calculate its own EAX. Notice that $EAX(u, v_d)$ $\forall u \in L(v_i)$ should be calculated prior $EAX(v_i, v_d)$. Thus, EAX calculation is realized iteratively starting from the vertices nearest to v_d. Later, we show the corresponding algorithm.

The EAX metric considers the same (sending) transmission rate for all vertices. In BB-PLC it is seldomly the case. The EATT metric [64] extends EAX to cover this issue. In [64], the authors express EAX in slightly different

manner. First, they define a weighting coefficient:

$$w(u,v) = \frac{p_{(v,u)} \cdot \prod\limits_{u' \in M(u,v)} (1 - p_{(v,u')})}{1 - \prod_{u \in L(v)} (1 - p_{(v,u)})}. \tag{3.2}$$

They also define d_{iJ} as the average amount of time needed by vertex v_i to transfer a single packet to the vertices in $J = L(v)$:

$$d_{vJ} = 1/(1 - \prod_{u \in L(v)} (1 - p_{(v,u)})). \tag{3.3}$$

Then, the total amount of time needed to transfer a single packet from v to v_d equals $EAX(v, v_d)$:

$$EAX(v, v_d) = d_{vJ} + \sum_{u \in J} w(u,v) \cdot EAX(u, v_d). \tag{3.4}$$

It is easy to observe that with the substitution of eqs. 3.2 and 3.3 into eq. 3.4 we obtain eqs. 3.1.

Afterwards, the authors in [64] extend d_{vJ} to consider the sending data rate:

$$d_{vJ}^{(r)} = d_{vJ} \cdot s/r, \tag{3.5}$$

where r is the sending data rate and s is the maximum size of the packet in the network. Then, EATT can be calculated in analog way:

$$EATT(v, v_d) = d_{vJ}^{(r)} + \sum_{u \in J} w(u,v) \cdot EATT(u, v_d). \tag{3.6}$$

In our work, we derived the analog expression independently of [18] and [64]. Our expression differs from eq. 3.6 basically in that the parameter s is not required and the correlation of packet losses can be considered.

Vertex priority in our work

The vertex priority can be defined as follows [21] (consider the network graph $\mathcal{G}$ and the channel model as defined in chapter 2):

Definition 3.1.2. "The priority of vertex v, $p(v)$, equals the mutual information between the coded message X_v on output of the vertex v and the coded message Y_{v_d} on input of the vertex v_d" [21].

The priority equation we derive as follows. Let vertex v have $m(v)$ symbols to deliver to v_d and $\tau(v)$ be the minimum amount of time required for this purpose when $m(v) \to \infty$. As far as $p(v)$ is the mutual information of the channel between X_v and Y_{v_d}, it equals the upper bound on the communication data rate. Thus, $p(v) = m(v)/\tau(v)$. Notice that here $p(v)$ is measured in symbols per second. Therefore, all logarithms usually appearing in equations for mutual information are to the base 2^l, where l is the size of the symbol in bits.

Let $\mathcal{G} = \{V, E\}$ be the network graph connecting v and v_d. Then, $\tau(v)$ consists of sending durations of all $u \in V$: $\tau(v) = \sum_{u \in V} t(u)$. The upper bound on the achievable data rate between v and the cooperation group $L(v)$ equals $I(X_v; \mathbf{Y}^{(v)}_{L(v)})$. Here, $\mathbf{Y}^{(v)}_{L(v)} = \{\cup_{u \in L(v)} Y^{(v)}_u\}$ and $Y^{(v)}_u$ is the symbol observed by u on its channel output when v sends X_v. Thus, the vertices in $L(v)$ will receive $m(v)$ symbols from v in not less than t seconds on average:

$$t = m(v)/I(X_v; \mathbf{Y}^{(v)}_{L(v)}).$$

During t seconds v can send $n(v)$ symbols:

$$n(v) = t \cdot d(v) = m(v) \cdot d(v)/I(X_v; \mathbf{Y}^{(v)}_{L(v)}),$$

where $d(v)$ is the sending data rate of vertex v.

Each $u \in L(v)$ forwards $Z^{(v)}_u$. This message contains that part of information received by u from v that is not present on any other node with higher priority (higher than $p(u)$). Formally, $Z^{(v)}_u = Y^{(v)}_u | \cup_{u' \in M(v,u)} \{Y^{(v)}_{u'} = y^{(v)}_{u'}\}$, where $M(v, u)$ is such subset of $L(v)$ that $\forall u' \in M(v, u) : \ p(u) < p(u')$. Note that $Z^{(v)}_u = Y^{(v)}_u$ if all vertices $u' \in L(v)$ with higher priority than $p(u)$ receive the symbol $Y^{(v)}_{u'} = \epsilon$ (erasure). Otherwise, $Z^{(v)}_u = \emptyset$. Thus, the vertex u forwards only $m(u)$ symbols from those received from v:

$$m(u) = n(v) \cdot I(Y^{(v)}_u; Z^{(v)}_u)/d(v).$$

Note that u requires in total $\tau(u) = m(u)/p(u)$ seconds to deliver $m(u)$ symbols to v_d (analog to $\tau(v)$).

Let, for notation convenience, $a(v) = I(X_v; \mathbf{Y}^{(v)}_{L(v)})$ and $b(u, v) = I(Y^{(v)}_u; Z^{(v)}_u)/d(v)$. Then, the total time needed to deliver $m(v)$ symbols from v to v_d can be cal-

culated as follows:

$$\tau(v) = t(v) + \sum_{u \in L(v) \setminus v_d} \tau(u) = \frac{m(v)}{a(v)} + \sum_{u \in L(v) \setminus v_d} \frac{m(v) \cdot b(u,v)}{p(u) \cdot a(v)}. \tag{3.7}$$

We use $\tau(v)$ to evaluate the priority of vertex v [21]:

$$p(v) = m(v)/\tau(v) = \frac{a(v)}{1 + \sum\limits_{u \in L(v) \setminus v_d} b(u,v)/p(u)}. \tag{3.8}$$

For PEC without correlation of CTFs the expression for $a(v)$ and $b(u,v)$ can be easily calculated:

$$\begin{aligned} a(v) &= d(v) \cdot (1 - \prod_{u \in L(v)} \varepsilon_{(v,u)}); \\ b(u,v) &= (1 - \varepsilon_{(v,u)}) \cdot \prod_{u' \in M(v,u)} \varepsilon_{(v,u')}, \end{aligned} \tag{3.9}$$

where $M(v,u)$ is a subset of $L(v)$ that $\forall u' \in M(v,u) : \ p(u) < p(u')$. One can observe that for PEC without correlation of CTFs the function $b(u,v)$ is analog to $w(u,v)$ (eq. 3.2) and $p(v)$ is similar to $EATT(v, v_d)$ in eq. 3.6.

Let us evaluate the highest value of the vertex priority. Logically, $p(v)$ should be maximized if the vertex v has a direct connection to v_d and it can communicate with v_d at data rate $d(v)$ without losses. In this case, $b(u,v)$ for all $u \in L(v) \setminus v_d$ equals zero and $a(v)$ equals $d(v)$. Then, the highest value of the vertex priority equals $d(v)$. In this case, v can increase its priority only by increasing $d(v)$. For certain values of $d(v)$ the communication between v and v_d becomes not possible without losses, which does not let $p(v)$ growing proportional to $d(v)$. Eventually, for high data rates the loss probability approaches one and $p(v)$ reduces to zero ($a(v) \to 0$). Thus, the highest value of the vertex priority depends on the tradeoff between the sending data rate $d(v)$ and the loss probabilities on edges in $L(v)$. In chapter 4, we analyze this tradeoff in PLC protocol with realistic channel model.

Evaluation of priority and formation of coalitions

For a given tradeoff between $d(v)$ and ε_e $\forall e \in O(v)$ $\forall v \in V$, it is possible to evaluate the priorities of all vertices and form their cooperation groups with

the help of the algorithm that is presented in [64]. We analyze whether this algorithm suits for our scenario.

In [64], each pair of vertices is connected only with one directed link. In PLC, the broadcast channel connects all vertices with asymmetric bi-directional links. We model such links with pairs of uni-directional edges $[e = (v, u), e' = (u, v)]$, $u, v \in V$. The loss ratios on edges e and e' can be different. It can be expected in PLC due to asymmerty of CTFs. In the following, we check if the algorithm in [64] works for topologies with asymmetric bi-directional edges as well.

For this purpose, we study the example topology in fig. 1.1b. This network contains only one bi-directional link $[(v_1, v_2), (v_2, v_1)]$. For simplicity, other links are uni-directional and the sending datarates of all vertices are equal. In this case, EATT simplifies to EAX. In appendix F we prove that the algorithm in [64] should not create uncertainty for the topologies with bi-directional links. We implemented this algorithm in C++ and checked it for multiple topologies with $|V| \in [3, 15]$ and random loss ratios for each edge. In this way, we concluded that this algorithm is loop-safe. Nevertheless, it has one drawback, which can be shown also on the example of fig. 1.1b. Consider the case $\varepsilon_{23} = \varepsilon_{13}$. Then, neither v_1 includes v_2 in $L(v_1)$ nor v_2 includes v_1 in $L(v_2)$. Nevertheless, it is almost not possible to happen in reality becuase the loss ratios have a floating point type and the probability of $\varepsilon_{23} = \varepsilon_{13}$ is very low.

In [64], the algorithm for calculation of $p(v)$ and $L(v)$ $\forall v \in V$ is developed.

Algorithm 1. Shortest-Multirate-Anypath-First [64]

```
1   for each vertex v in V
2       do p(v) ← 0
3          L(v) ← ∅
4          T(v) ← 0
5          for each rate r in R
6              do p(v)^(r) ← 0
7                 L(v)^(r) ← 0
8   p(v_d) ← ∞; set p(v_d) to the maximum possible value
9   S ← ∅
10  Q ← V
```

```
11  while Q ≠ ∅
12      do u ← Find-Max(Q); see Find-Max(Q) below
13          S ← S ∪ {u}; add vertex u to the set S
13          Q ← Q \ {u}; remove vertex u from the set Q
14          for each incoming edge (v, u) in E
15              do for each rate r in R
16                  do L(v)^(r) ← L(v)^(r) ∪ {u}
17                      if p(v)^(r) < p(u)
18                          then Calculate p(v)^(r) (eq. 3.8)
19                              if p(v) < p(v)^(r)
20                                  then p(v) ← p(v)^(r)
21                                       L(v) ← L(v)^(r)
22                                       T(v) ← r
23                          else L(v)^(r) ← L(v)^(r) \ u
```

Here, the Find-Max(Q) function returns the vertex u from the set Q with the highest priority among other vertices in this set (the destination has the highest priority).

For all scenarios in section 2.5, we use the above algorithm to calculate the priority of the source node and substitute it in the eq. 2.5 as $R = p(v_s)$. The obtained gain values completely concide with the values in figures of section 2.5. This is possible only if $p(v)$ equals the upper bound on the achievable data rate (for the considered scenarios). Hence, the priority is the most accurate routing metric for the routing protocol that aims to maximize the throughput.

Calculation of TDMAP

As follows from the optimization task in eq. 2.3, the achievable data rate is maximized with the certain TDMAP. The highest effective data rate between v and v_d can be also evaluated as $p(v)$ with the help of the algorithm 1. This data rate can be achieved only if v and other vertices from the network graph connecting v and v_d follow the optimal TDMAP. In our simulations, we monitor the real durations for channel access of all vertices. We target to compare these durations with the optimal TDMAP, which can localize the communication problems in the network. Using $p(v), L(v)\ \forall v \in V$, we propose an algorithm to calculate the optimal TDMAP.

Algorithm 2. Calculation of TDMAP (Start from *main* function)

1 **def** *c(Node v):*
2 **return** $d(v)/I(X_v; \mathbf{Y}^{(v)}_{L(v)})$*;* $I(X_v; \mathbf{Y}^{(v)}_{L(v)})$ *is the mutual information (see section 2.3)*
3 **def** *calc(Node v):*
4 $S \leftarrow S \cup \{v\}$*; add vertex* v *to the set* S
5 **for each vertex** $u \in L(v)$
6 $n[u] \leftarrow n[u] + n[v] \cdot b(v,u)/c(v)$
7 **if** $u \notin S$ **then** $A \leftarrow calc(u)$
8 **def** *main:*
9 $V \leftarrow V \setminus Z \; \forall u \in Z : \; |L(u)| = 0$*; exclude the vertices with empty cooperation groups*
10 $n[v_s] \leftarrow 1/c(v_s)$
11 $A \leftarrow calc(v_s)$
12 **while** A **not empty**
13 **do** *action* $\leftarrow A$
14 **execute** *action*
15 **for each vertex** $u \in L(v)$
16 $plan[u] \leftarrow n[u]/d[u]$
17 $T \leftarrow T + plan[u]$
18 **for each vertex** $u \in L(v)$
19 $plan[u] \leftarrow plan[u]/T$

First, we remove such vertices from V that have empty cooperation groups. Then, starting from v_s, we calculate the number of symbols to be sent $n[v]$ $\forall v \in V$. For this purpose, we use the function *calc(Node v)* recursively. The recursion is organized in a special manner. We create the buffer of actions A. This buffer stores the pointers to *calc(Node v)* function for different vertices. Such recursion targets to call the next function only after the actual function exits. When all $n[v]$ $\forall v \in V$ are calculated, we find the fractions of time needed to send these numbers of symbols. The array *plan* stores the TDMAP for each vertex.

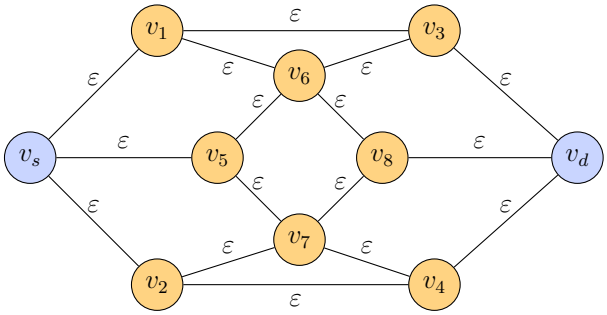


Figure 3.2: Example of topology. Mesh network (all links are bi-directional)

Verification of algorithms

The algorithms for $p(v)$ and TDMAP calculation have been verified for several sample topologies. Note that $p(v_s)$ should be equal to the upper bound on data rate between v_s and v_d given by eq. 2.2. The TDMAP given by the same equation should also concide with the results of algorithm 2. For triangular, diamond and bigger mesh network shown in fig. 3.2, we notice the complete concidence of the results of the algorithms 1 and 2 with the solution of the optimization task in eq. 2.2. We release the corresponding source code in [65] for dissemination.

This result does not only prove the idea and implementation validity. It also highlights that the routing protocol using the same principle as the algorithm for the vertex priority calculation can go beyond the performance of CSPR. If properly designed, this may allow approaching the upper bound data rate in the given communication channel, which is not possible for any CSPRs (see section 2.4).

3.2 Transmission policy in idealized network

In the previous section, we dealt with the first part of the ORP structure proposed in [59], i.e. the selection of the cooperation groups (definition 3.1.1). Meanwhile, we defined the vertex priority that is also used in other ORP parts. In this section, we concentrate on the mechanism of retransmissions and resolving the competition between the vertices in the cooperation groups.

In total, we target to design the routing protocol that uses all available channel resources and minimizes its own overhead, e.g., the traffic for route maintenance, link quality estimation, etc. But at first, we assume all kinds

of overhead produced by ORP to have negligibly small cost. As a result, any vertex can send the feedback to any other vertex for each received packet at zero cost. To our best knowledge, the existing works on ORP do not contain this fundamental study. Many ideas of the known ORP variants are based on the practical observations rather than on the theoretical study. Our analysis starts with the idealized ORP. This allows better understanding of the challenges that appear during the development of ORPs for real networks.

With a perfect feedback, one can take many other idealized assumptions. For example, each vertex knows the quality of each point-to-point link in terms of the packet loss ratio. Thus, each vertex can calculate its priority (assuming single data rate for all vertices and absence of correlation between packet losses). Also, all vertices are instantly informed about the erasure of each symbol that allows to make no useless retransmissions. With the perfect feedback on the Logical Link and Control (LLC) protocol layer, there is no need to code the data because the protocol can perform the retransmission at the zero cost whenever required. In addition, the vertices communicate to avoid sending of replications. Eventually, all vertices can instantly agree on the channel access. Thus, the collisions are eliminated.

Let v_s send the packet set B to v_d in T time slots. The time T consists of discrete number of slots for channel access by each vertex $v \in V$. The feedback sending consumes no time resource. Let untill a certain time $t \in [0, T]$ the vertices receive the fragments of the original packet set ($B_v \subseteq B$). It is clear that for any two vertices $u, v \in V$ the sum of sizes of packet sets on each of them ($|B_v| + |B_u|$) is greater or equal than the size of the set $B_{\{v,u\}} = B_v \cup B_u$. Below, we quantify the value $|B_{\{v,u\}}|$ as the total DOF. The term of DOF has the same meaning as described in section 1.2.

Definition 3.2.1. Total Degree of Freedom (DOF) of the subset of vertices $Z \subseteq V$, denoted as $\vartheta(Z)$, equals the rank of the matrix containig the symbols $\cup_{v \in Z} B_v$.

So, the same symbol present on several vertices in Z counts only once in $\vartheta(Z)$. It is clear that $\vartheta(Z) \leq |B|$. The term of total DOF is applied to find out if a subnetwork of vertices Z can reconstruct the original message, which is possible only if $\vartheta(Z) = |B|$. The distribution of useful information on the vertices in Z is quantified with the unique DOF.

Definition 3.2.2. Unique DOF of vertex $v \in Z$, $Z \subseteq V$, denoted as $\rho(v)$, quantifies the amount of useful (not neccessarily unique) symbols on a given vertex amongst the vertices in the given set and equals the difference between $\vartheta(Z)$ and $\vartheta(Z \setminus v)$.

It can be calculated for all vertices iteratively as follows:

Algorithm 3. Calculation of $\rho(v)$ $\forall v \in Z$

```
1   while Z ≠ ∅
2       do v ← SELECT(Z)
3           Y ← Z \ v
4           ρ(v) ← ϑ(Z) − ϑ(Y)
5           Z ← Y
```

Note that $\rho(v)$, $\forall v \in V$, depends on the SELECT(Z) function. We specify it in the definition of the Basic Routing Rules (BRR). Also note that the sum $\sum_{v \in Z} \rho(v) = \vartheta(Z)$ does not depend on SELECT(Z).

The terms of total and unique DOF are used in definition of BRR.

3.2.1 Basic Routing Rules (BRR)

Our routing protocol targets to maximize the throughput. Therefore, the developed routing rules target to route the data over the fastest path and make no useless retransmissions. The first rule helps to exclude certain sink vertices $V^+(O(v))$ of the vertex $v \in V$ from the list of potential forwarders. Basically, it was already used in the definition of the cooperation group (definition 3.1.1).

Routing rule 1. Exclusion rule.
If $p(v) \leq p(u)$ then u does not include v in its cooperation group $L(u)$, $\forall u, v \in V$.

In appendix A, we prove the necessity of this rule for the triangular network topology.

The exclusion rule in this form is also used in algorithm 1 and in [26],[25] for the construction of the cooperation groups. The novelty lies in the usage of the new vertex priority expression (definition 3.1.2). Remember that using this rule, the solution given by algorithm 1 coincides with the solution of the optimization task in eq. 2.2. Thus, it is necessary for optimal performance.

There are multiple other approaches known in the literature. They are mainly motivated with the real-world circumstances, which are not taken into account by the idealized ORP (BRR). Mostly, they extend the above-defined exclusion rule. For example, in MORE [31] it is proposed to prune additionally such vertices $u \in L(v),\ v \in V$, which plan to perform less than 10% transmissions in $L(v)$. In SOAR [32], $u \in V^+(O(v))$ is added to $L(v)$ if multiple additional conditions fulfill. First, the difference between ETX(v) and ETX(u) should exceed a certain threshold. Second, the vertex u should have a good connection to one or more vertices on the main path between v_s and v_d as defined by CSPR. Third, the size of $L(v)$ should not be equal a certain maximum size. Eventually, the edge (v, u) should significantly improve the quality of the hyperlink $(v, L(v))$. In PRO [27], the authors allow cooperation for relays with a Receive Signal Strength Indicator (RSSI) above a certain threshold. The threshold can be dynamically updated basing on the number of erasures. They target to maintain the RSSI threshold corresponding to 25% of erasures. If the actual number of losses is greater than PRO reduces the threshold allowing more vertices to cooperate, which makes the communication more robust. Some of these ideas are later used in the extension of BRR for real networks. Before that, we state other routing rules for the idealized network.

Routing rule 2. Retransmission rule.
The vertex v sends the symbols, eventually redundant, as long as $\rho(v) > 0$. The unique DOF $\rho(v)$ decrements for each positively acknowledged symbol by at least one vertex $u \in L(v)$.

There are multiple variations of these rule known in the literature. In MORE [31], the source continues sending until it receives an Acknowledgement (ACK) from v_d that it has decoded $\hat{S}^m$. In SOAR [32], each vertex sends the redundant symbols until it either gets the ACK from at least one vertex in $L(v)$ or the maximum retry count is reached. The redundant symbols are sent after the Retransmission TimeOut (RTO) similar to Transmission Control Protocol (TCP). They use piggyback ACK and ACK compression to reduce RP overhead. The number of ACKs is also reduced using cumulative/selective ACKs. CCACK [30] piggybacks the hash vector that describes the coding matrix. Receiving the symbol from the vertex located closer to the destination, the receiving vertex compares the hash vector with its own received and trans-

mitted coding vectors. In this way, it can find out the amount of useful data that it still has to send. Thus, no redundancy is added by default. The redundant symbols are sent only after receiving the hash vectors. PRO [27] sends redundant symbols if it does not receive an ACK. It stops the retransmissions either if the maximum retry counter is reached or a new symbol is received. All these protocols, except CCACK, consider that it is not possible to send the ACK for each received symbol by each vertex. In fact, the perfect feedback in BRR is the optimal solution. It eliminates all unnecessary retransmissions. It meantime, it guarantees the reception of the complete message by v_d.

Each vertex performs the retransmissions only for that data, which has to be forwarded by this vertex. The vertices form the set of symbols that have to be forwarded using the filtering rule.

Routing rule 3. Filtering rule.
Upon reception of symbol y by v from w, if $y \neq \epsilon$ (ϵ stays for erasure), v increments its unique DOF $\rho(v)$ only if y was not received by any $u \in L(w),\ p(u) > p(v)$.

The same rule is presented in [26]. ExOR [59], MORE [31] and SOAR [32] do not distinguish clearly between the redundancy and the filtering rules. MORE defines the "triggering ratio", the number of symbols to be sent per each received symbol. ExOR and SOAR use the delayed transmission start. The sender vertex specifies the forwarders list in the packet header. The closer is the receiving vertex to the begin of the forwarders list, the earlier it triggers the transmission. Hearing this transmission, other vertices cancel their timers. This approach breaks in case of a bad connection between the candidate relays. This problem was addressed in ECONOMY [29]. The authors propose to filter out such vertices from the cooperation group that cannot hear all other vertices in this group. In addition, they advocate the usage of a token to avoid the duplicates completely.

Channel access with BRR

The channel access is organized with a perfect centralized Time Division Multiple Access (TDMA) scheduler. It maintains the list of active vertices and the size of their queues at zero cost of overhead. Each vertex v is active, i.e. wants

to send the symbol, only if $\rho(v) > 0$. At the beginning of each time slot, the scheduler notifies all vertices who sends the symbol in the next time slot and the time slot duration. It selects the transmitting vertex randomly from the set of active vertices. The vertex selection in the next time slot is not affected by the previous selections unless $\rho(v) = 0$.

BRR example

With the following example, we demonstrate the BRR operation. Consider the network topology in fig. 1.1a. Let v_s and v_r use the same constant sending data rate, the symbols have equal size and it takes exactly one time slot of constant duration to deliver one symbol.

The vertices v_s and v_r agree on the usage of the next time slot at the end of the current time slot. If both v_s and v_r, in accordance with the redundancy rule, have data to transmit, only one of them gets the channel access. The sender is selected randomly and instantly.

As before, the message block of G_s symbols is communicated from v_s to v_d. In fig. 3.3, the communication is demonstrated on slot-by-slot basis. One can see that either v_s or v_r can send in each time slot. Each transmission is

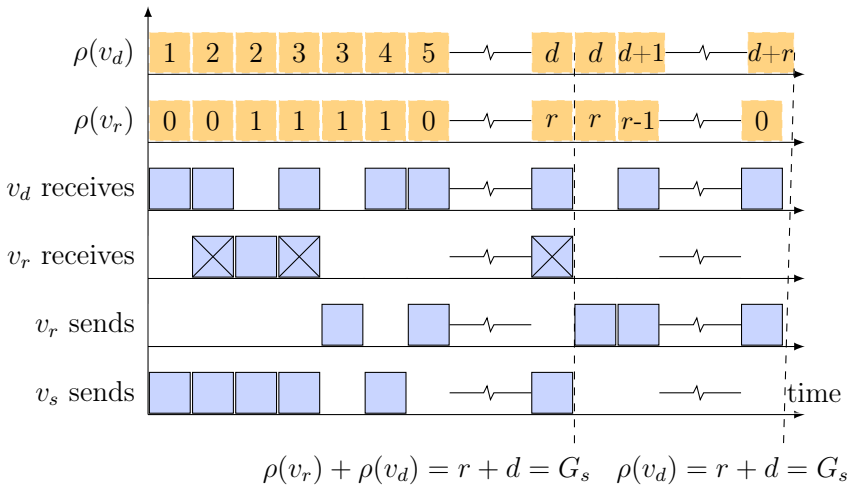


Figure 3.3: BRR in operation (fig. 1.1a). $p(v_r) > p(v_s)$

broadcast. If the symbol is erased, we leave a blank space on "v_r receives" and/or "v_d receives" axis. So, in the first time slot v_s sends and only v_d hears the symbol. In the second time slot, both v_r and v_d hear the symbol. In

accordance with the filtering rule, v_r does not increment $\rho(v_r)$. In the third time slot, v_r gets the symbol, which v_d does not receive. Then, at the end of the 3rd time slot, v_s and v_r decide on that who transmits in the 4th time slot. As soon as v_r succeeds to deliver the symbol, the value $\rho(v_r)$ decrements. Note that ACK is sent by each vertex (both positive and negative) at the end of each time slot. If $\rho(v_r) = 0$ then v_r does not contend for the channel access. It continues untill $\vartheta(\{v_r, v_d\}) = \rho(v_r) + \rho(v_d) = G_s$. At this time point, v_r and v_d have enough symbols to decode the original message. So, v_s should stop sending. After this, v_r sends the remaining symbols until $\rho(v_d) = G_s$.

In accordance to the exclusion rule, this example holds only for $p(v_r) > p(v_s)$.

Monte-Carlo simulation

We constructed a Simple Network Simulator (SNS) that models the channel as described in section 2.2 ([65]). Basically, it allows constructing the network graph and specifying the loss ratios and the sending rates. It also implements BRR and its further extensions. It is an discrete-event simulator and thus its simulation speed does not depend on the computer performance. With its help, we compared BRR with the flooding protocol. This protocol does not issue any strict rules for the relays. Each relay replicates the received symbols an unlimited number of times. The simulation run ends when the destination gets the complete message. The simulation is repeated 1000 times using $G_s = 100$ and the diamond topology as shown in fig. 3.4. This figure shows also the loss ratios on all edges.

It is clear that with flooding much more symbols are forwarded by both relays. Thus, there is much more channel uses required with a view to delivering G_s symbols to v_d. The filtering and exclusion rules play a vital role in BRR for this scenario.

3.3 Transmission policy in real network

The BRR requires the perfect (instant) feedback link. In this section, we extend the BRR to overcome this idealized assumption. The extended basic routing rules are referred to as Basic Routing Rules Extended (BRR-E). Indifferent to BRR, with BRR-E each vertex encodes data using RLNC.

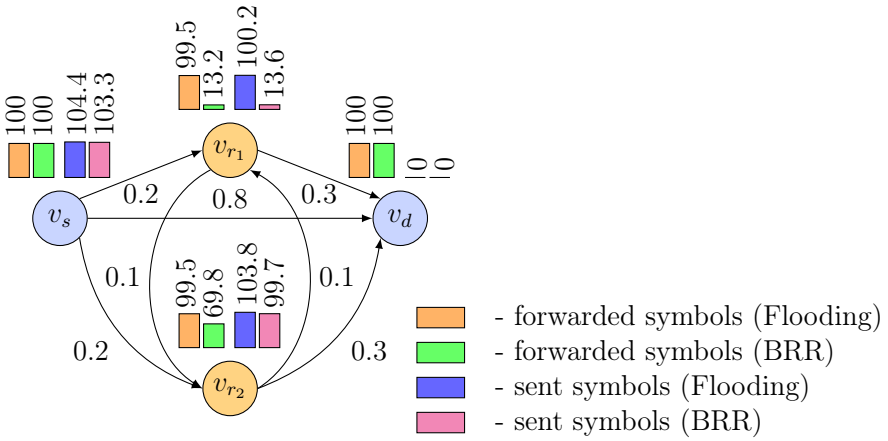


Figure 3.4: Simulation scenario and results

3.3.1 Purpose of NC in ORP

BRR uses the extensive feedback to make sure that each received symbol is forwarded only by one vertex and this vertex has the shortest path to the destination. If we reduce the number of feedbacks, we cannot be sure anymore that the vertices follow the desired strategy. The probability that certain vertex takes the wrong decision is greater than zero. In the following, we show that NC helps to decrease this probability.

Consider the following example. Let v_s generates the message M and segments it in the chunks of equal size resulting in the set of original symbols B_{v_s}. Then, the symbols from B_{v_s} are broadcasted (uncoded) to the network defined with the network graph $\mathcal{G} = \{V, E\}$. All vertices $v \in V$ try to fulfill the targets of the BRR, i.e. they try to let each symbol being replicated only once and the replicating vertex should have the shortest path to the destination. Let the vertices v_0 and v_1 with priorities $p(v_0)$ and $p(v_1)$, $p(v_0) > p(v_1)$, receive the sets of symbols $B_0, B_1 \subset B_{v_s}$, $B_0 \neq B_1$, from v_s. Let also B' denote the set of symbols $B_1 \setminus B_0$ and $B' \neq \emptyset$. In accordance to BRR, v_0 should transmit all symbols from B_0 and v_1 should send only $B' \subseteq B_1$. With the perfect feedback, v_1 can construct B' when v_0 communicates the sequence numbers of the symbols in B_0. In this section, we consider that such feedback is not feasible due to the consumption of substantial channel resources. We look for another way that follows the BRR strategy.

Without receiving the feedback, v_1 tries to guess the set B'. If it knows ε_e, $e = (v_s, v_0)$, it predicts that each symbol $s \in B_{v_s}$ broadcasted by v_s does not

arrive v_0 with probability $p = \varepsilon_e$. Let A_0 and A_1 denote the reception events of the symbol $s \in B_{v_s}$ by v_0 and v_1 correspondingly ($A_0, A_1 = \{0, 1\}$ where 0 stays for loss). Let also D be the decision event of v_1 to forward s ($D = \{0, 1\}$ where 1 stays for the positive decision). The event that v_1 makes a wrong decision can be evaluated as $F = A_1 \cap (A_0 \cap \hat{D} \cup \hat{A}_0 \cap D)$. The probability of F can be calculated for uncorrelated losses as follows:

$$P(F) = (1 - \varepsilon_{e'}) \cdot ((1 - \varepsilon_e) \cdot p + \varepsilon_e \cdot (1 - p)) = 2 \cdot (1 - \varepsilon_{e'}) \cdot \varepsilon_e \cdot (1 - \varepsilon_e), \quad (3.10)$$

where $e' = (v_s, v_1)$. $P(f)$ is maximized with $\varepsilon_{e'} = 0$ and $\varepsilon_e = 0.5$. In this case, it equals 0.5. The vertex v_1 should make the forwarding decision for each received symbols, i.e. $|B_1|$ times. The expectation value of "wrong" symbols in B' can be calculated as $|B_1| \cdot P(F)$. This number is too high for the routing rules to be effective. Therefore, we look for another solution.

Assume that v_1 applies the recoding with the RLNC before sending. In this case, it takes the forwarding decision for all symbols in B_1 only once. In fact, it simply calculates the number of coded symbols, $|B'|$, which have to be produced out of B_1 without adding the redundancy. Let v_1 calculate $|B'|$ as $|B'| \cdot \varepsilon_e$. The vertex v_1 takes the wrong decision (prediction mistake) if $|B'| \neq |B_0 \cup B_1|$. It can be easily shown that $P(|B'| \neq |B_0 \cup B_1|) \to 0$ when $|B_{v_s}| \to \infty$. We verify this statement by means of Monte-Carlo simulation using the topology in fig. 3.5. Each simulation run consists of several transmissions.

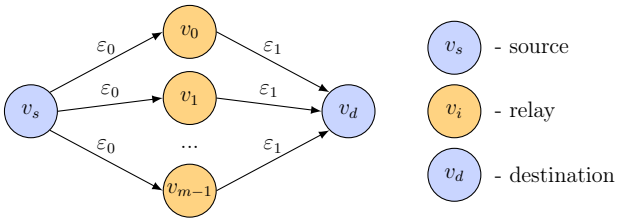


Figure 3.5: Two-hop multi-relay scenario. $\varepsilon_0 = 0.4,\ \varepsilon_1 = 0$

First, the source sends G_s coded symbols using RLNC. Each relay v, $v \in L(v_s)$, receives $k(v)$ coded symbols, recodes them producing $n(v)$ symbols and forwards them to v_d. Here, $L(v_s)$ is the cooperation group of v_s. Thus, the relays are sorted in the descending order according to their priorities ($p(v)$). And let $n(v) = k(v) \cdot \prod_{u \in M(v,u)} \varepsilon_{(v_s,u)}/(1 - \varepsilon_{(u,v_d)})$, where $M(v, u)$ is such set of vertices so that $\forall u \in M(v, u) :\ p(u) < p(v)$ [64, 18]. The sending order of the

relays is arbitrary. When all relays finish sending we calculate $n = \sum n(v)$, the decoder matrix rank r on v_d and the rank of the virtual matrix d, which contains all $k = \sum k(v)$ symbols. If $n > r$ the relays send $s_+ = n - r$ useless symbols. If $r < d$ the relays do not send all information they have: $s_- = d - r$. Note that BRR aims to send no useless symbols ($s_+ = 0$). In the meantime, it ensures the forwarding of the complete message to the destination ($s_- = 0$).

We ran this example 1000 times and plotted $a_+ = s_+/G_s$ and $a_- = s_-/G_s$ in figure 3.6. It is clear that in this scenario the recoding allows approaching

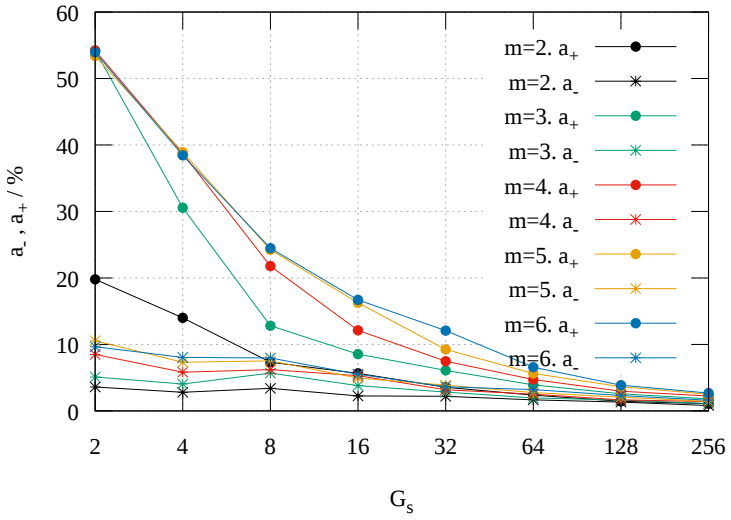


Figure 3.6: Monte-Carlo simulation. m - number of relays; $a_+ = s_+/G_s$ - ratio of useless symbols; $a_- = s_-/G_s$ - ratio of not sent symbols

the BRR target ($s_+ = 0$ and $s_- = 0$) when $G_s \to \infty$. For this purpose, it does not use the feedback but the prediction, i.e. no protocol overhead. For $G_s < \infty$, the feedbacks are required to transfer the missing several percents of data. But a number of feedbacks is much less than the required by BRR.

3.3.2 Extended BRR (BRR-E)

BRR-E notably differs from BRR due to the usage of NC. Another difference lies in the possibility to consider the correlation of packet losses. For this purpose, each vertex maintains a reception map for each sender. We shortly discuss the process of loss ratio calculation.

Packet loss based metrics

Each coded symbol is appended with a CRC block. The receivers use it to verify the consistency of the symbol. After verification they save the status of decoded ("0") or erased ("1") symbols for each sender separately into $n = |I(v)|$ drop-tail queues.

Such queues are also referred to as reception maps in BRR-E. They contain the sequence of receiving status bits and the sequence number of the symbol corresponding to the first status bit. Each map itself allows calculating the packet loss ratio ε_e for the corresponding input edge $e \in I(v)$. Additionally, it is possible to calculate the probability of the packet losses for a set of edges. In PLC, CTFs are often correlated. As far as BRR-E is heavily based on expressions for mutual and conditional information, the knowledge of $\varepsilon_e \, \forall e \in I(v)$ values would be insufficient.

Let m_e is the map of the received and erased symbols for the edge e. Let it be delivered to $v = v^-(e)$ in a feedback from $v^+(e)$. At certain time point, v collected m_e from certain output edges $e \in O(v)$. Let $O(v)$ includes only those edges, which sinks form the coalition of v. The packet loss correlation estimation is possible only if all m_e are synchronized. It means that the first element in each map corresponds to the same symbol broadcasted by v and each map has equal length s. If the maps are originally not synchronized they are trimmed at the start to the latest sequence number and at the end to the shortest length.

After the synchronization, the resulting maps may have an insufficient number of values to give a statistically accurate representation of the symbol losses on corresponding edges. Let the procedure $f(m_e)$ return *true* if the accuracy of ε_e is satisfactory. Let $Q(v)$ be such subset of $O(v)$ so that $\forall e \in Q(v) : \; f(m_e) = 1$ and $Q'(v) = O(v) \setminus Q(v)$. If $Q'(v) \neq \emptyset$ then it is not possible to synchronize all maps. Only the maps corresponding to $Q(v)$ can be synchronized. The edges in $Q'(v)$ can be either discarded or considered as uncorrelated. In the first case, the size of the coalition decreases, which reduces the performance of BRR-E. In the second case, BRR-E becomes inaccurate, which also can reduce the performance.

The reception maps are reset each time when the vertex changes its sending data rate (PHY encoding and/or Bit Allocation Table (BAT)). All maps m_e

with either $v^-(e) = v$ or $v^+(e) = v$ are affected.

Priority calculation

Now, we finalize the equation for the priority calculation. In eq. 3.8, we defined the variables $a(v)$ and $b(v,u)$ for the general case. Here, we specify them using the reception maps. Let the output edges of v, $O(v)$, be divided into two complimentary groups $Q(v)$ and $Q'(v) = O(v) \setminus Q(v)$ as discussed above. Consider the following notation conveniencies: $Z(v) = V^+(Q(v))$ and $Z'(v) = V^+(Q'(v))$. Let the synchronized maps m_e, $\forall e \in Q(v)$, have length s. Then, $a(v)$ and $b(v,u)$ values can be calculated as follows:

$$\begin{aligned} a(v) &= d(v) \cdot \left(1 - \prod_{u \in Z'(v)} \varepsilon_{(v,u)} \cdot \left\| \bigcup_{u \in Z(v)} m_{(v,u)} \right\|^2 / s \right); \\ b(v,u) &= d(v) \cdot (1 - \varepsilon_{(v,u)}) \cdot \prod_{u' \in M'(v,u)} \varepsilon_{(v,u')} \cdot \left\| \bigcup_{u' \in M(v,u)} m_{(v,u')} \right\|^2 / s, \end{aligned} \tag{3.11}$$

where

- $\|x\|^2$ is the vector norm; here x is binary, hence, $\|x\|^2$ is equal the number of ones in x;
- $\varepsilon_{(v,u)} = \left\| m_{(v,u)} \right\|^2 / s$;
- $M(v,u)$ (resp. $M'(v,u)$) is such subset of $Z(v)$ (resp. $Z'(v)$) that $\forall u' \in M(v,u)$ (resp. $M'(v,u)$): $p(u) < p(u')$;
- $d(v)$ - sending data rate of vertex v.

It was considered that for two independent Random Variable (RV)s A and B the following fulfills: $E[A \cap B] = E[A] \cdot E[B]$.

The usage of reception maps also allows updating the routing rules.

Exclusion rule

If the vertices calculate their priorities using eqs. 3.8, 3.11 then the exclusion rule of BRR is automatically extended. Thus, its formulation remains unchanged.

Extended routing rule 1. Exclusion rule.
The same as the routing rule 1 in BRR.

Redundancy rule

Using the coding, we add the redundancy rule. Note that $L(v)$ means the coalition of v, m is the number of symbols be forwarded and $n(v)$ is the number of en-/recoded symbols by v.

Extended routing rule 2. Redundancy rule.
The coding rate of vertex v:

$$m/n(v) = I\left(X^{n(v)}; \bigcup_{u \in L(v)} Y_u^{n(v)}\right)/d(v) = a(v)/d(v).$$

The vertices can also add less or more redundancy than advocated here. In this way, it is possible to manipulate the rertrasnmission probability and the number of send retransmission requests as shown in appendix E. We address this topic in section 3.4.

Towards the filtering coefficient

The next rule requires the definition of the filtering coefficient:

$$p_f(v,u) = \left\| \bigcup_{u' \in M(v,u)} m_{(v,u')} \right\|^2 /s, \tag{3.12}$$

where $M(v,u)$ is as before in eq. 3.11 and s is the size of reception maps after the synchronization. Thus, the data from each sender v is filtered at receiver u separately. The equation of the filtering coefficient is analog to the idea in the definition of the effective forwarding rate in [26]. It differs in that eq. 3.12 considers the correlation of packet losses.

The vertex v needs to calculate $p_f(v,u))$ for each u. For this purpose, it requires rather much information. In fig. 3.7, we give an example for the calculation of $p_f(v,u)$. The position of each vertex on the horizontal axis corresponds to its priority (highest priority to the right). The vertex set $V^-(I(v))$ consists of the vertices $\{u_0, u_1, u_2, u_3\}$. In fact u_3 does not include v to its coalition because $p(v) < p(u_3)$ (BRR-E exclusion rule). So, v drops all data

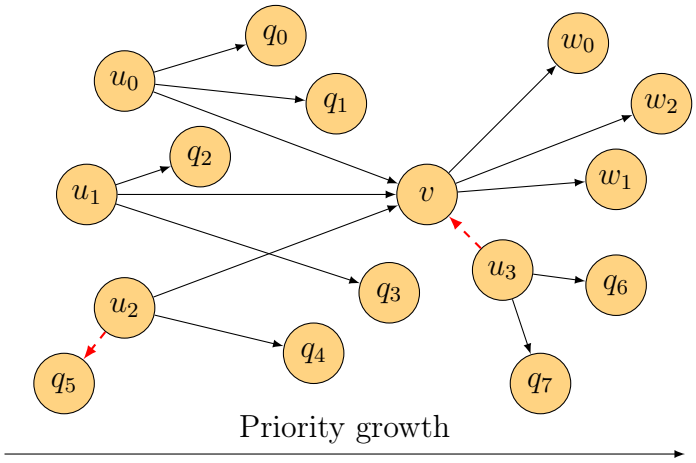


Figure 3.7: Fragment of the mesh network. For all shown edges: $\varepsilon_e < 1$. Red dashed edges are not used for calculation of $\rho(v)$ (eq. 3.13)

received from u_3. The relationship between u_2 and q_5 is analog. For calculation of $p_f(v, u_0)$ the vertex v needs the receiving status maps $m_{(u_0,q_0)}$, $m_{(u_0,q_1)}$ and $m_{(u_0,v)}$. The latter one is calculated by v itself. The other two are calculated by q_0 and q_1 corresponding. All three vertices v, q_0, q_1 should deliver these maps to u_0 (needed by redundancy rule). So, either u_0 or q_0 and q_1 themselves should deliver $m_{(u_0,q_0)}$, $m_{(u_0,q_1)}$ to v. Consider that for $p_f(v, u_1)$ and $p_f(v, u_2)$ extra data have to be communicated.

In overall the vertex v requires receiving status maps from all $q \in V^+(O(u))$ for all $u \in V^-(I(v))$. The communication of all these maps may create substantial overhead. We target to avoid this overhead.

Indeed, v does not need all those maps. It needs only the values $p_f(v, u)$. We let each vertex u calculate it for v. It makes sense becase u has already all required data (u uses the same maps for redundancy calculation). Then u can attach $p_f(v, u_0)$, $p_f(v, u_1)$ and $p_f(v, u_2)$ to the packet header. If MAC packet contains multiple LLC symbols the inclusion of $p_f(v, u)$ values once per MAC packet is sufficient. With quasi-stationary channel the values $p_f(v, u)$ may not change till the next MAC packet is transmitted. So, it is even not obligatory to include them in each MAC packet.

With the simplification above v should communicate only with its neighbours: $V^-I(v)$ and $V^+O(v)$.

Filtering rule

Following the principle of the filtering rule in BRR, we define the new filtering rule that considers the possibility of recoding with RLNC.

Extended routing rule 3. Filtering rule.
Any vertex $u \notin L(v)$ does not increase $\rho(u)$ upon symbol reception from v. If $u \in L(v)$ receives m symbols from v it increases $\rho(u)$ on $k = \lfloor m \cdot (1 - p_f(v,u)) \rfloor$.

Note that filtering in BRR-E refers to calculation of $\rho(u)$ only. The symbols are not filtered as in BRR. All received symbols are added to the coding matrix. All the received symbols are used for recoding. Notice also that $\rho(u)$ in BRR-E becomes a floating point variable.

The filtering rule can be further enhanced as follows. Let v received $N(u)$ symbols from u and sent non symbols since receiving the first one yet. Then, the unique DOF of v can be defined as follows:

$$\rho(v) = min\left\{k, r - r'\right\}, \tag{3.13}$$

where

- $k = \lfloor \sum_{u}^{Z(v)} (N(u) \cdot (1 - p_f(v,u)) \rfloor$;
- $Z(v)$ is such subset of $V^{-}(I(v))$ that $\forall u \in Z(v) :\ p(u) < p(v)$
- r is the coder matrix rank;
- r' is the parameter discussed below.

We explain the parameter r' using [35] as example. The authors in [35] propose a method to reduce the number of linear dependent symbols in the wireless network. The mobile devices $v_0, v_1, ..., v_m$ are located geographically on a straight line from the base station and at the equal distance between each other. Say, the index of the device quantifies its distance to the base station. The base station broadcasts some data wanted by each mobile device (point-to-multipoint). If the device-to-device cooperation is allowed, the mobile devices can work as repeaters. The authors in [35] showed that the best MAC scheduling strategy with NC gives a priority to the repeaters with the biggest distance from the base station. Let v_i recode and send n coded symbols and another repeater

v_j, $j < i$, can overhear this transmission. Note that the device v_j has potentially more data than v_i since it is located closer to the base station. After reception of $s \leq n$ symbols, v_j adds them to its coder matrix and also creates a separate decoding matrix, where it puts the same symbols. Let the rank of the first coder matrix on v_j, r, increased on Δr and the rank of the second coding matrix is r'. When the MAC scheduler gives the channel access to v_j, the vertex v_j forwards only $(r - r')$ symbols instead of r.

The same idea is used in our work. If v receives symbols from any u with higher priority, it does not drop them. Each vertex maintains two matrices: one for symbols received from the vertices with both lower and higher priority and one for the symbols received from the vertices with higher priority only. The former matrix is used to produce the recoded symbols. The latter one is used only to compute r'.

Note that receiving the symbols from higher priority the vertices may increase r but in accordance with the filtering rule the may not increase $\rho(v)$. We motivate it as follows. If v overhears the transmission from u, $p(u) > p(v)$, we consider the probability that the packet received from u by v will be useful for any other vertex w, $p(w) > p(v)$, as very small.

In appendix D, we prove that BRR-E allows reaching the upper bound data rate in the given communication channel.

Channel access with BRR-E

As in BRR, in BRR-E the vertex v tries to access the channel each time when $\rho(v) > 0$. Hereby, there is no threshold value of $\rho(v)$. The routing policy in [44] (PlayNCool) is much similar to BRR-E for triangular topology. In accordance with this work, the relay starts helping only when it gathers a certain amount of DOF. In appendix C, we show that there is a limit on the overall number of symbols to be sent by the relay but there is no limit on that when the relay should start helping. The only condition is that $\rho(v) > 0$. PlayNCool uses the auxiliary restriction of that the relay and the source should stop sending approximately at the same time, which appears unnecessary. The maximum achievable data rate with BRR-E and PlayNCool is the same.

3.4 Automatic ReQuest reply (ARQ)

In our work, we implement a protocol that performs routing, network initialization and manages retransmissions. We refer to this protocol as ANChOR. BRR-E is a part of ANChOR. As also proposed in [31, 38], ANChOR maintains several generations in the active state simultaneously. It means that the source may send the coded symbols from the next generations before the destination can decode the current one. Analog to the Selective-Repeat ARQ, this method improves both latency and data rate in comparison to Go-Back-N and Stop-And-Wait. It becomes especially useful in big networks with high latency and high bandwidth between v_s and v_d, e.g. a BB-PLC network with multiple hops. The number of active generations may be much greater than one in the networks with high latency. It also grows when the generation size decreases.

In order to guarantee the data recovery at the destination, ANChOR performs retransmissions. The probability of successful data reception can be also increased when increasing the coding rate. As we show in appendix E, there is a tradeoff between the retransmission probability and the coding rate. If the RR can be sent with zero cost, it is better to add no coding redundancy. Otherwise, one can select the retransmission probability and the corresponding coding rate depending on the RR cost and the amount of excessive redundancy. Here, we define the excessive redundancy as the number of codewords that were sent and the number of codewords that are sufficient for message decoding. Note that it is a random variable due to the randomness of the codeword loss process.

In ANChOR, RRs are performed on a per-generation basis.

The RRs in ANChOR are sent for the batches of symbols. Each batch corresponds to one generation. ANChOR uses BRR-E for each generation in its buffer. The retransmission rule of BRR-E defines the contents of RR and the RR processing procedure for each single batch. ANChOR adds the following functions:

- which vertices are eligible to originate/forward the retransmission request (RR);
- when RR can be sent;

- which RR receivers are eligible to perform the retransmission.

3.4.1 Retransmission request message contents

The MORE [31] protocol uses a simplistic approach to ensure the reception of the complete generation by v_d. It instructs v_s to send the redundant symbols until it gets ACK from v_d. In the period equal to the end-to-end delay v_s sends useless redundant symbols. This issue can be mitigated letting the relays to send ACKs as well. The participating vertices can use Go-Back-N or Stop-And-Wait to avoid endless waiting. Nevertheless as shown in [38], the classic ARQ schemes like Go-Back-N, Stop-And-Wait, and Selective-Repeat are outperformed (in terms of throughput) by the simple intra-flow NC collaborative scheme GalaRes [38]. Therefore, we compare only those ARQ schemes that use NC.

Basically, we distinguish between two types of ARQ mechanisms. In the first one, the vertices from the cooperation group $L(v)$ of v acknowledge the reception to v. In the second, v sends RR to the vertices in $V^{-}(I(v))$ demanding to send the missing data. The example of the first ARQ type is ARQ using the reception maps as described in the subsection below. Other subsections describe the ARQ mechanisms of the second type.

Using reception maps

The most straightforward approach is the usage of the reception maps. It is the most similar method to the legacy ARQs because it uses Serial Sequence Numbers (SSNs). Still, there is a big difference in the way of SSN usage by the ARQ with and without NC. The ARQ with NC does not need SSNs for sequence control because each two coded symbols of the same generation are equivalent. Nevertheless, it is possible to use SSNs in a different fashion. We propose to use them for the construction of the reception masks as described in section 3.3.2.

Let vertex v broadcast s coded symbols. Say, s includes the redundant symbols and v adds as much redundancy that the expectation value of symbols received by its cooperation group $L(v)$ equals n. Note that in difference to the traditional multicast problem, the vertex v does not target to deliver n symbols to each vertex $u \in L(v)$. Instead, all vertices $u \in L(v)$ together

should receive at least n symbols. Let X be a binomially distributed Random Variable (RV) describing the number of symbols received by all vertices in $L(v)$. Then, $E[X] = n$. Let x be the realization of X. The vertex v targets $x = n$. But in fact, x can be both greater and less than n with 50% probability. Therefore, v needs the feedbacks to find x and retransmit missing information.

For each MPDU sent by v, it can receive up to $|L(v)|$ feedbacks (one MPDU may contain many coded symbols). If no feedbacks are received then v makes a guess: $\hat{x} = E[X] = n$ ($\hat{x}$ is the estimation of x). If all feedbacks are received then $\hat{x} = x$. We analyze if it is possible to reduce the number feedbacks without causing a significant difference between $\hat{x}$ and x. Thus, we calculate $\hat{x}$ when v gets only $0 < k < |L(v)|$ feedbacks.

Consider the following realistic example. Let the vertex v request the channel access to send s symbols but gets the resource to send $s' \leq s$ symbols only. After sending s' symbols, v receives the feedback from $0 \leq k \leq |L(v)|$ vertices that form the set $L'(v) \subseteq L(v)$.

In each feedback, v obtains the reception map $m_{(v,u)}$, $u \in L'(v)$. Then, it synchronizes the maps as described in section 3.3.2. The total number of symbols received by the vertices in $L'(v)$ can be calculated as follows:

$$m = \left|\left| \cup_{u \in L'(v)} m_{(v,u)} \right|\right|^2 . \tag{3.14}$$

where $||x||^2$ is the square of Euclidean norm.

The expectation value of symbols received by the vertices $w \in \{L(v) \setminus L'(v)\}$ can be calculated as follows (assuming a binomial distribution):

$$\hat{x}' = E[X'] = s' \cdot (1 - ||h||^2/|h|), \tag{3.15}$$

where

- binomial RV describing the number of received symbols by the vertices in $L(v) \setminus L'(v)$;
- $h = \cup_{u \in \{L(v) \setminus L'(v)\}} m_{(v,u)}$;
- $|x|$ is the size of vector x.

The maps from vertices $u \in \{L(v) \setminus L'(v)\}$ correspond to older feedbacks. As before, if h is not long enough to obtain the desired statistical accuracy then

the edges in the set $[L(v) \setminus L'(v)]$ are divided in two groups: synchronized and unsynchronized.

The sum of m and $\hat{x}'$ does not give $\hat{x}$ because m and $\hat{x}'$ are the realization and expectation value of two correlated random processes. In the following we find $\hat{x}$.

Let a be the number of the same symbols received by vertices both in $L'(v)$ and $[L(v) \setminus L'(v)]$. The probability that the symbol is not received by vertices in $L'(v)$ is $p_1 = (s' - m)/s'$. An analog probability for the vertices in $[L(v) \setminus L'(v)]$ is $p_2 = ||h||^2/|h|$. Then the value a can be calculated as follows:

$$a = s' \cdot (1 - p_1) \cdot (1 - p_2). \tag{3.16}$$

So, the expectation value of symbols received by the vertices in $L(v)$ when v sends s' symbols is given as follows:

$$\hat{x} = \hat{x}' + m - a. \tag{3.17}$$

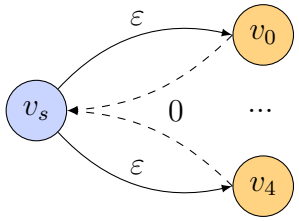


Figure 3.8: Test network scenario for feedback importance

Eq. 3.17 is verified by Monte-Carlo simulation. The network topology under test is shown in fig. 3.8. The source has the full rank coding matrix with size G_s. The coalition size is five: $|L(v_s)| = 5$. The losses on edges $e \in O(v_s)$ are described with independent Bernoulli processes with the same expectation value ε. The source broadcasts $s = G_s/1 - \varepsilon^{|L(v_s)|}$ symbols. Then, v_s receives $0 \leq k \leq 5$ feedbacks and estimates $\hat{x}$ using eq. 3.17. Fig. 3.9 shows the estimation error for different k that is calculated as follows:

$$\Delta = (\hat{x} - x)/G_s. \tag{3.18}$$

We performed the simulation for two values of $\alpha = \varepsilon^{|L(v_s)|} = \{0.01, 0.1\}$. For each α and k, the test was repeated 1000 times. The confidence probability of the confidence intervals is 0.95.

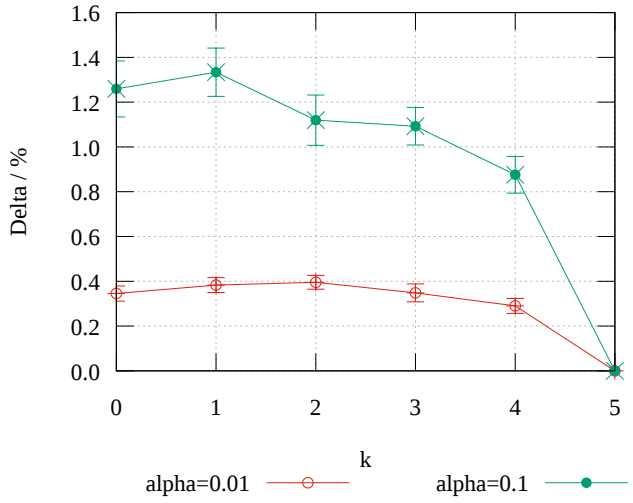


Figure 3.9: Verification of eq. 3.17 using eq. 3.18. $\alpha = \varepsilon^{|L(v_s)|}$

We conclude that there is hardly any difference in the estimation error for the number of feedbacks $0 \leq k < |L(v)|$. The smaller the loss product is the less sense it makes to use the reception maps for $\hat{x}$ estimation. Thus, it makes no sense to the reception maps in multi-relay scenarios with the NC-aided ARQ.

Coding matrix rank

GalaRes [38] proposes another simplistic approach for RR. The RR message in GalaRes contains the amount of missing DOF in the coding matrix of the RR sender. In general, this information is not sufficient to make an efficient retransmission response. We explain it by the following example. Consider a part of the network described by the graph $\mathcal{G}(V', E')$ where $V' = \{v_0, v_1, v_2\}$ and $E' = (v_0, v_2), (v_1, v_2)$. Let $B^{Rx}_{v_i}$ describe the set of received coding vectors by v_i correspondingly. Also, assume that at some moment of time $B^{Rx}_{v_0} \neq \emptyset$, $B^{Rx}_{v_1} \neq \emptyset$, $B^{Rx}_{v_2} = \emptyset$, $B^{Rx}_{v_0} \neq B^{Rx}_{v_1}$ and v_2 forms the group of potential RR responders as $Z = \{v_0, v_1\}$. For simplicity, $\varepsilon_{(v_0,v_2)} = \varepsilon_{(v_1,v_2)} = 0$. Then, v_0 broadcasts all vectors in $B^{Rx}_{v_0}$ that results in $B^{Rx}_{v_2} = B^{Rx}_{v_0}$. Afterwards, v_2 randomly selects the RR responder v_1 from Z. It indicates the rank of its coding matrix in RR message that is equal $Rank(B^{Rx}_{v_2})$. When v_1 receives RR, it calculates the number of symbols to be sent as $\min(Rank(B^{Rx}_{v_1}), G_s - Rank(B^{Rx}_{v_2}))$ where G_s is the generation size. In fact, it can send up to $Rank(B^{Rx}_{v_1} \setminus B^{Rx}_{v_0})$ useful symbols for

v_2. In extreme case when $B_{v_1}^{Rx} \setminus B_{v_0}^{Rx} = B_{v_1}^{Rx}$ and $Rank(B_{v_1}^{Rx}) = Rank(B_{v_0}^{Rx})$, v_1 decides that it has no useful data for v_2, while in reality it has $Rank(B_{v_1}^{Rx})$ DOF that are missing on v_2. Therefore, sending only the rank of the coding matrix in RR can lead to substantial downgrade of the routing protocol performance.

The performance can be improved if v_2 distiguishes between the coding vectors received from different senders. Let the vertex v receives the coded symbols belonging to the same generation from several vertices $u \in V^-(I(v))$ and let the sequence of symbols received from u be denoted as B_u. The vertex v can attach $Rank(B_u) \forall u \in V^-(I(v))$ to the RR message. Each u knows how much DOF it wanted to forward. Let it be denoted as n_u. So, u can calculate the remaining number of symbols to be sent as $\max(0, n_u - Rank(B_u))$.

Number of linear dependent symbols

The above approach can be enhanced further. In [34, 33], the feedback includes the number of Linearly Dependent Symbols (LDS). Our approach differs in that we do not use the number of LDSs as the input variable of the routing protocol. Instead, we use just use a flag to indicate the presence of LDSs

Let v keep the track of the linear dependence of the received vectors. For this purpose, whenever v gets the coded symbol c from u, it feeds it to the coding matrix B that contains all received symbols. If the rank of this matrix does not increase, it sets the flag b_u. In the RR message, v attaches all flags. In this way, the vertex u can find out if any of its sent coded symbols were useless for v. If it is true then u can send no more useful information to v unless it receives new data itself.

All coding vectors

We also show the naive but most accurate approach to form the RR message. It lies in attaching the coding vectors of all innovative coded symbols. In this way, the RR receivers can precisely calculate the amount useful information that they have for the RR sender. Let B_v and B_u denote the coding matrices of a given generation on v and u correspondingly. Let v sends the RR message containing the coding vectors of B_v. Upon RR reception, u creates a virtual decoder and feeds it with the coding vectors from B_u and B_v. The difference

between the virtual decoder rank and $Rank(B_u)$ gives the number of coded symbols, which u produces with recoding the coded symbols in B_u. Nevertheless, this method has an obvious drawback. The size of the RR message is large. In appendix G we calculate the protocol overhead that can be created by such messages on a particular example of G.hn protocol. In the best case, it equals 7.4%. Finally, we implement this approach in SNS as the most accurate method to form the RR contents. Note that SNS does not consider the RR message overhead. In our G.hn implementation, the RR message overhead is taken into account. In order to reducing the protocol overhead, we study other approaches.

Null Space Basis (Null Space Base (NSB)) approach

The CodeCast [23] protocol proposes to compress the information about the coding vectors in a single vector of size G_s, i.e. the same size as each vector in the coding matrix. It is named the hash vector. CodeCast calculates it as the random vector belonging to the null space of the coding matrix.

Definition 3.4.1. Let S^0 be the space of all possible vectors y of size G_s and each element of the vector y be the element of the field $GF(q)$ of size q. Say, $z \times y = 0$, where $z \in S^0$ and $y \in S \subseteq S^0$. The space of all possible vectors z fulfilling this equation, S^1, is called the null space of vector y ($S^1 \subset S^0$). Any random vector from S^1 is the *hash vector* of space S.

In this way, a RR message carries the information about the vector space of the RR sender u. The RR receiver v multiplies each of coding vectors in its own coding matrix by the hash vector. The number of coding vectors that do not give the zero in the product with the hash vector is used as the number of useful coded symbols that v has for u.

With this method, v can reliably detect that u has received c if u really has received this coding vector. Otherwise, there is a non-zero *false-positive* probability, i.e. the probability that v thinks that u received c although it did not. This is the price for the "compression" of the RR message. The false-positive probability can be up to $1/q$ [23], which is high for small field sizes, e.g. $1/q = 0.5$ for the binary Galois field.

We demonstrate the calculation of the false-positive probability in Code-

Cast that is useful in further analysis of CCACK approach [30].

The encoder can produce up to $|S^0| = q^{G_s}$ different coding vectors. For any element of y, $y_i \in [0, q]$, $i \in [0, Gs-2]$, there exists exactly one possible $y_{Gs-1} \in [0, q]$ so that $y \times z = 0$ (if z is strictly non-zero). Thus, $|S^1| = q^{Gs-1}$. Say, S^2 be the linear space spanned by vectors in a certain matrix $M_{G_s \times G_s}$. The degrees of freedom of this space can be calculated as $d = G_s - Rank(S^2)$. It means that any vector z that is produced with linear operation on vectors in S^2 has d free variables. Thus, there exist $|S^2| = q^{Rank(S^2)}$ vectors in this space. So, there can be up to $|S^3| = |S^0 \setminus S^2|$ vectors on the encoder that do not belong to S^2 ($y \notin S^2$). And, there are up to $|S^4| = |S^1 \setminus S^2|$ different vectors that are orthogonal to the hash vector $z \in S^2$ and do not belong to S^2. So, the probability that $y \times z = 0$, $y \in S^0$ and $y \neq S^2$ can be calculated as follows:

$$P = \frac{|S^3|}{|S^4|}. \tag{3.19}$$

As far as $S^2 \subseteq S^1 \subset S^0$ the above probability can be written as follows:

$$P = \frac{q^{G_s-1} - q^{Rank(S^2)}}{q^{G_s} - q^{Rank(S^2)}} < \frac{1}{q}. \tag{3.20}$$

Cumulative Coded ACK (CCACK) approach

The authors of CCACK [30] eliminated several shortcomings of CodeCast. For example, the false-positive probability was reduced to $(1/q)^n$ where $n > 1$. Thus, CCACK seems to have the much smaller size of the feedback than the size of all vectors in the coding matrix. And, in meantime, it can have very small false-positive probability, which makes CCACK very attractive. Therefore, we analyze it in detail.

The hash vector we denote as V'. In CCACK, V' can be calculated with the following algorithm (see used notations in table 3.1).

Algorithm 4. Generation of V' [30]

1 Select M^{Rx}

2 Generate $H^1, ..., H^n$

3 Calculate B

4 Generate V and calculate V'

5 Validate V'

This algorithm includes several steps that require a detailed description. The selection of acknowledging matrix M^{Rx} and generation of hash matrices $H^1, ..., H^n$ is explained with the algorithms below.

Algorithm 5. Selection of M^{Rx} [30]

1 **while** $|M^{Rx}| \neq k$
2 **do** $a_t \leftarrow \min(a_i \; \forall i \in [1, |B^{Rx}|])$
3 $M^{Rx} \leftarrow [M^{Rx}, B_t^{Rx}]$ (concatenation)

Algorithm 6. Generation of $H^1, ..., H^n$ [30]

1 **while** $Rank(H'_J) \neq n$ **for any** $J = \{j_1, ..., j_n\}$, $j_i \neq j_l$, $j_i, j_l \in [1, G_s]$
2 **for each** $j \in [1, n]$ **for each** $i, l \in [1, G_s]\, i \neq l$
3 **do** $h_{il}^j = 0$
4 **for each** $j \in [1, n]$ **for each** $i, l \in [1, G_s]\, i = l$
5 **do** $h_{il}^j = rand(GF(q^n) \setminus 0)$

The vector from the null space V' has the following property: $\forall i \in [1, k]\; m_i \times V' = 0,\; m_i = \{m_{i1}, ..., m_{iG_s}\}$. We use it to verify the validity of the implementation (the last step in the algorithm 4).

In addition, we quantify the false-positive probability of the feedback algorithm. Let the vertex v send the symbol with the coding vector c and the reception event of c by u be denoted as $A_c = \{0, 1\}$ where $A_c = 0$ stays for loss. If $A_c = 1$, u acknowledges all vectors in its receive buffer $B^{Rx\,u}$ to v, $c \in B^{Rx\,u}$ (not only the vector c [30]). Upon feedback reception, v may decide if the vector c', $c' \in [B^{Rx\,v}, B^{Tx\,v}]$, belongs to $B^{Rx\,u}$ using V'. Let us denote this event as $D_{c'} = \{0, 1\}$ where $D_{c'} = 1$ stays for the decision of v that $c' \in B^{Rx\,u}$. If $A_c = 1$ then $P[D_c = 1] = 1$ [30]. If $A = 0$ then $P[D_c = 1] = (1/q)^n$, where q is the prime of the field size and n is the number of hash matrices. It means that there exist a non-zero probability of the false decision. It can be reduced increasing the number of hash matrices n. It is called the false-positive probability of the feedback algorithm in [30]. Later we refer to this definition as $P[D_c = 1 | A_c = 0]$ or simply $P[D = 1 | A = 0]$.

Table 3.1: Notations

$B^{Rx}_{t\times G_s}$	The set of the received coding vectors
$B^{Tx}_{t\times G_s}$	The set of the sent coding vectors
n	Number of hash matrices
$M^{Rx}_{k\times G_s},\ k = \min\{\frac{n}{G_s} - 1, \lvert B^{Rx}\rvert\},\ M^{Rx} \subseteq B^{Rx}$	The set of the selected received coding vectors to be acknowledged
$m_{ij} \in GF(q^n)$	The element of M^{Rx} at i-th row and j-th column
a_i	Number of times the vector (row) i from B^{Rx} was selected for acknowledgement
$H^1_{G_s\times G_s}, ..., H^n_{G_s\times G_s}$	Hash matrices
$h^j_{il} \in GF(q^n)$	The element of H^j at i-th row and l-th column
$\Phi^j = M^{Rx} \times H^j,\ j \in [1, n]$	M^{Rx} rotated by H^j
$\Phi = [\Phi^j\ \forall j \in [1, n]\,]$	Concatenated matrices Φ^j
Φ'	Φ in echelon form
B	Basis of the null space of Φ' (also Φ and M^{Rx})
$V_{1\times G_s} = \{v_1, ..., v_{G_s}\}, v_i \in GF(q^n)$	Random vector
$V'_{1\times G_s} = B^T \times V$	Hash vector from the null space of M^{Rx}
$H'_{J=\{j_1,..,j_n\}} = \begin{pmatrix} h^{(1)}_{j_1} & h^{(2)}_{j_1} & \cdots & h^{(n)}_{j_1} \\ h^{(1)}_{j_2} & h^{(2)}_{j_2} & \cdots & h^{(n)}_{j_2} \\ \vdots & \vdots & \ddots & \vdots \\ h^{(1)}_{j_n} & h^{(2)}_{j_n} & \cdots & h^{(n)}_{j_n} \end{pmatrix}$	Check matrix for hash matrices (see algorithm 4)

We evaluate this probability in our implementation with the help of Monte-Carlo simulation. For this purpose, we create one encoder and one decoder

connected with a lossy channel ($\varepsilon = 0.5$). The encoder is connected to the traffic generator. Firstly, $G_s = 100$ original symbols are generated. They are encoded with the full-vector codec (systematic encoding is switched off) from the c++ NC library KODO [48] using the generation size G_s and the field size 2^3. The encoder produces coded symbols that are transferred over the lossy channel to the decoder. Upon each reception, the decoder sends the feedback message containing the hash vector V' produced with the algorithm 4. The feedback is delivered to the encoder without losses. For each feedback, the encoder tries to identify the coding vectors received by the decoder. For this purpose for each $c \in B^{Rx\,v}, B^{Tx\,v}$ it applies the following algorithm:

Algorithm 7. Feedback assessment for $c \in B^{Rx\,v}, B^{Tx\,v}$ [30]

```
1   for each j ∈ [1, n]
2       do if c × H^j × V' = 0
3           then return false
4   then return true
```

Remark. In fact for the example with one encoder-decoder, it is sufficient to check $c \in B^{Tx}$ only. The vectors $c \in B^{Rx}$ should be checked if alternative paths exist (scenario with helpers).

If algorithm 7 returns "true" then the corresponding c is marked as heard by the decoder. In accordance to [30], the encoder continues sending as long as the following condition fulfills:

$$Rank(B_h^{Rx\,v} \cup B_h^{Tx\,v}) < Rank(B^{in\,v}), \tag{3.21}$$

where $B_h^{Rx\,v} \subseteq B^{Rx\,v}$ is the subset of the heard coding vectors (analog for $B_h^{Tx\,v}$) and $B^{in\,v}$ is the set of innovative coding vectors. In our scenario, $Rank(B^{in\,v}) = G_s$. The encoder should stop sending (the condition in eq. 3.21 should be false) when the rank of the decoder becomes full. In fact, the encoder cannot know that for sure. Instead, it uses the hash vector to make the guess. It estimates the number of symbols $\hat{k} = [Rank(B^{in\,v}) - Rank(B_h^{Rx\,v} \cup B_h^{Tx\,v})]$ that should be innovative for the decoder. On real, this number equals $k = [Rank(B^{in\,v} \cup B^{in\,u}) - Rank(B^{in\,u})]$. But the encoder does not have $Rank(B^{in\,u})$. Thus, it

can calculate only the estimate $\hat{k}$. Besides $P[D = 1|A = 0]$, we calculate also the following false-positive probability:

$$P[Rank(B^{in\,v}) - Rank(B_h^{Rx\,v} \cup B_h^{Tx\,v}) \neq Rank(B^{in\,v} \cup B^{in\,u}) - Rank(B^{in\,u})]. \tag{3.22}$$

This probability has a closer relation to the routing rules than $P[D = 1|A = 0]$. In fact, $\hat{k}$ specifies the number of symbols that the encoder should retransmit.

Again, in one encoder-decoder scenario this equation can be simplified considering $Rank(B^{in\,v} \cup B^{in\,u}) = Rank(B^{in\,v})$.

CCACK drawback

CCACK proposes to use the feedback containing the hash vector of the receiver and the transmitter matrices. The receivers of the feedback can identify with a high accuracy if they have any useful data for the sender of the feedback. Nevertheless, it may be insufficient to have only this information. Say, v sends $N \gg 1$ symbols and u receives $M < N$ of them. If we aim to reduce a number of feedbacks, we may want to send the feedback only once per M received symbols. In this case, v should be able to learn from the feedback not only if it has any useful data for u but also how many symbols are useful (innovative). Unfortunately, CCACK approach does not allow to give such information for all feedback receivers. We prove this statement with the following example.

Consider the triangular scenario. Let the source v_s transmit data belonging to one particular generation. Its coding matrix has the full rank. Let all Coding Vector (CV)s in the coding matrix of v_s be denoted as $B^{in\,v_s}$. In general, $B^{Tx\,v_s} \subseteq B^{in\,v_s}$, $B^{Rx\,v_r} \subseteq B^{Tx\,v_s}$, $B^{Tx\,v_r} \subseteq B^{Rx\,v_r}$, $B^{Rx\,v_d} \subseteq B^{Tx\,v_r}, B^{Tx\,v_s}$. We examine a particular case. Consider that v_s transmits all symbols with CVs from $B^{Tx\,v_s}$ before v_r starts transmissions. We also assume that $Rank(B^{Tx\,v_s}) = |B^{Tx\,v_s}|$, i.e. all CVs in $B^{Tx\,v_s}$ are linear independent, and $Rank(B^{Tx\,v_s}) = Rank(B^{in\,v_s})$. Let the relay gets all symbols sent by v_s, i.e. $B^{Rx\,v_r} = B^{Tx\,v_s}$, and the destination gets none of them inducing $B^{Rx\,v_d} \subseteq B^{Tx\,v_r}$. Then, the relay transmits all symbols from $B^{Tx\,v_r}$, while $B^{Tx\,v_r}$ spans the vectors in $B^{Rx\,v_r}$, i.e. v_r transmits all information it has. The destination v_d observes the losses. Thus, $B^{Rx\,v_d} \subset B^{Tx\,v_r}$. For simplicity, let v_d calculate the hash vector V as the NSB vector for complete matrix $B^{Rx\,v_d}$ and without rotation by hash matrices as proposed in CCACK [30]. Such method was also used in Code-

Cast [23]. Let us denote the vector space of all NSB vectors of matrix B as $\mathcal{N}(B)$. The vertex v_d selects the NSB vector V randomly from $\mathcal{N}(B^{Rx\,v_d})$. There are in total $q^{G_s - Rank(B^{Rx\,v_d})}$ possible choices where q is the finit field size. Any vector $V \in \mathcal{N}(B^{Rx\,v_d})$ is orthogonal to $|B^{Rx\,v_d}|$ vectors from those present in $B^{Tx\,v_r}$. It is reasoned by that $B^{Rx\,v_d} \subset B^{Tx\,v_r}$. Let us denote the set of such orthogonal vectors as B'^{v_r}. If the false-positive probability (discussed in [30]) is not counted then $B'^{v_r} = B^{Rx\,v_d}$. Therefore, when v_r gets V in the feedback from v_d, it can calculate the number of useful symbols for v_d as $n_{v_r} = Rank(B^{Tx\,v_r}) - Rank(B'^{v_r})$. If $B'^{v_r} = B^{Rx\,v_d}$ the value n_{v_r} is accurate. CCACK shows that the probability $P[B'^{v_r} \neq B^{Rx\,v_d}]$ can be made arbitrarily small ($P[D = 1|A = 0] = 1/q^n$). When v_s gets V in the feedback from v_d it tries to calculate n_{v_s} in the same fashion as v_r calculates n_{v_r}. For this purpose, v_s forms B'^{v_s} as the set of vectors from $B^{Tx\,v_s}$ that are orthogonal to V. Since v_d does not receive any symbols from v_s directly then $B^{Rx\,v_d} \not\subset B^{Tx\,v_s}$. Therefore, $B'^{v_s} \neq B^{Rx\,v_d}$ as it could have been assumed for v_r. In order to evaluate n_{v_s}, we calculate the probability p that the vector $U \in B^{Tx\,v_s}$ is orthogonal to V. The size of $\mathcal{N}(B^{Tx\,v_s})$ equals $q^{G_s - Rank(B^{Tx\,v_s})}$. As far as $\phi(B^{Tx\,v_s}) \in \phi(B^{Rx\,v_d})$, where $\phi(B)$ is the space of vectors spanned by those in B, then $\mathcal{N}(B^{Tx\,v_s}) \in \mathcal{N}(B^{Rx\,v_d})$. Thus, the probability p can be calculated as follows:

$$p = \frac{|\mathcal{N}(B^{Tx\,v_s})|}{|\mathcal{N}(B^{Rx\,v_d})|} = \frac{q^{G_s - Rank(B^{Tx\,v_s})}}{q^{G_s - Rank(B^{Rx\,v_d})}} = \frac{1}{q^{Rank(B^{Tx\,v_s}) - Rank(B^{Rx\,v_d})}}. \tag{3.23}$$

Then, the size of B'^{v_s} can be calculated as follows:

$$|B'^{v_s}| = |B^{Tx\,v_s}| \cdot p. \tag{3.24}$$

As assumed before, $B^{Tx\,v_s}$ contains no linear dependent CVs. Then, $|B^{Tx\,v_s}| = Rank(B^{Tx\,v_s})$ and $|B'^{v_s}| = Rank(B'^{v_s})$. The source calculates the number of innovative symbols for the destination as follows:

$$\begin{aligned} &n_{v_s} = Rank(B^{Tx\,v_s}) - Rank(B'^{v_s}) = Rank(B^{Tx\,v_s}) \cdot (1 - p) = \\ &Rank(B^{Tx\,v_s}) \cdot (1 - \frac{1}{q^{Rank(B^{Tx\,v_s}) - Rank(B^{Rx\,v_d})}}). \end{aligned} \tag{3.25}$$

It is clear that if $Rank(B^{Rx\,v_d}) \neq Rank(B^{Rx\,v_s})$ then $n_{v_s} > Rank(B^{Tx\,v_s}) - Rank(B^{Rx\,v_d})$. The value n_{v_s} is accurate only if $Rank(B^{Rx\,v_d}) = Rank(B^{Rx\,v_s})$

($n_{v_s} = 0$).

If v_s uses n_{v_s} to send the redundant symbols then it surely sends more symbols than needed. Thus, the CCACK approach can let v_s know if it has any innovative symbols for v_d but it cannot tell v_s how many innovative symbols it has.

We again verify this result by means of Monte-Carlo simulation. We setup the simulation script to guarantee $Rank(B^{Tx\,v_s}) - Rank(B^{Rx\,v_d}) = 1$ and $Rank(B^{Tx\,v_s}) = G_s = 32$. Then, the expectation value of n_{v_s} ($\hat{n}_{v_s}$) should be equal $G_s \cdot \frac{q-1}{q}$ (using CodeCast approach). With $q = 256$, $n_{v_s} = 32 \cdot 255/256 = 31.875$. The simulation with 10000 iterations gives $\hat{n}_{v_s} = 31.88$ (using *test/test-ccack-manual-relay.h* [65]). Then, we set $Rank(B^{Tx\,v_s}) = Rank(B^{Rx\,v_d}) = G_s - 1$. In this case, $\hat{n}_{v_s} = 0$. As already stated above, the CodeCast approach can let the receiver of the feedback know, if it has any useful data for the sender of the feedback. But it does not give the amount useful data. This simulation result confirms the statement and validates the analytic formula for n_{v_s}.

Although CCACK solves several problems of CodeCast, it has another drawback that is not present in CodeCast. Using CCACK, it is possible to acknowledge with one hash vector maximum $Rank(B^{Rx\,v_d})-1$ of the received vectors. Thus, even in case when $Rank(B^{Tx\,v_s}) = Rank(B^{Rx\,v_d})$, the destination acknowledges the subset $B \subset B^{Rx\,v_d}$ with the rank at most $Rank(B^{Rx\,v_d}) - 1$. As a result, it is even not possible to tell if the source has any useful data for the destination. With CCACK it is possible to acknowledge the received vectors only to those vertices, from whom these vectors are received. In our scenario, v_d gets no vectors from v_s directly. Thus, the acknowledgement of v_d is useless for v_s.

Note that in the considered scenario a simpler approach is to send the rank of the coding matrix on v_d ($Rank(B^{Rx\,v_d})$) instead of the hash vector in the feedback. But this approach will not work if we consider the triangular topology scenario as a part of a bigger network. Then, both v_s and v_r are the relay vertices and v_s may not have the full rank coding matrix. As far as in general case, v_s does not know if DOF at v_d are obtained directly from v_s or from v_r, it cannot calculate n_{v_s} as $Rank(B^{Tx\,v_s}) - Rank(B^{Rx\,v_d})$. Doing that the real number of innovative symbols on v_s is underestimated.

We conclude that CCACK has a substantial inaccuracy in a multi-relay

network.

Min-Max approach

Here, we design another approach that maintains the advantage of the small size of the RR message.

The vertices can give more information about their coding matrices than their ranks and NSB vectors. They also can add to the RR message the flags of the pivot and uncoded symbols. The coded symbols with the unity coding vectors are called uncoded. Each packet passed to the encoder is associated with the certain index corresponding to the place in the coding matrix. The uncoded symbols have the same indices in the decoder matrix as they have in the encoder matrix. Thus, the decoder attaches only the uncoded/not uncoded flags (without indices).

When the symbols are passed to the decoder it performs a partial Gaussian elimination.

Definition 1. *Let the coding vector corresponding to the row of the coding matrix with index i be denoted as c_i and its j-th element as c_{ij}. If $\forall j < i : c_{ij} = 0$ then the coded symbol on the row with index i is said to be the "pivot" or "seen" symbols by the decoder.*

Using the coding matrix rank and the maps of uncoded and pivot symbols we propose to calculate the lower and upper bound on the useful information that the RR receiver has for the RR sender.

I. Upper bound

The RR sender v needs at most $(G_s - r)$ symbols, where G_s is the generation size and r is the decoder matrix rank at v. The RR receiver u can send at most $\min\{r' - n,\ G_s - r\}$ useful symbols, where r' is the decoder matrix rank of u and n is the number of symbols decoded at v and seen at u.

II. Lower bound

If the symbol is not seen by v but seen by u, it will definitely increase the coding matrix of v. Note that the greater is the index of symbols the harder is to mark it as "seen". So, u has at least m useful coded symbols, where m is the number of symbols not seen by v but seen by u. From another side, if the rank of u is higher than the rank of v then u has at least $(r' - r)$ useful symbols.

So, the lower bound can be stated as $\max\{m,\ r'-r\}$. It is interesting to note that if $r' < r$ then there is still a chance that $m > 0$.

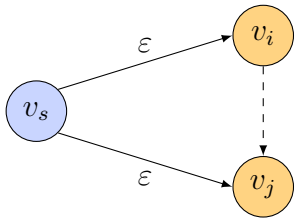


Figure 3.10: Test network scenario for feedback estimation

III. Expectation value of useful symbols

The expectation value of useful symbols lies between the lower and upper bounds. For simplicity, we let N be in the middle of this interval:

$$N = \begin{cases} \min\{r'-n,\ G_s-r\} \text{ if } \min\{r'-n,\ G_s-r\} < \max\{m,\ r'-r\}. \\ \max\{m,\ r'-r\} + \frac{\min\{r'-n,\ G_s-r\} - \max\{m,\ r'-r\}}{2} \text{ otherwise} \end{cases} \tag{3.26}$$

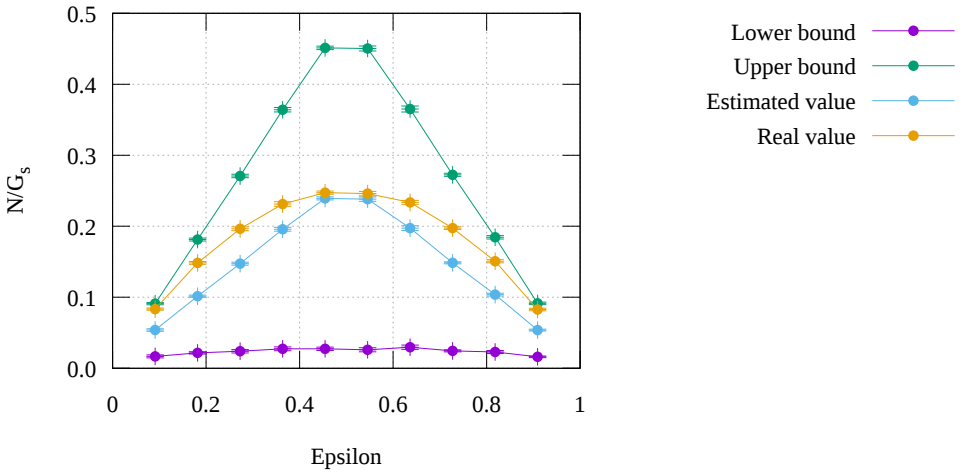


Figure 3.11: Verification of eq. 3.26

Using the full-RLNC codec in KODO library [48] we tested eq. 3.26 in a simple artificial scenario. The network consists of three vertices: one source and two relays (fig. 3.10). The source broadcasts G_s symbols. Some symbols are lost at v_j and at v_i due to the Bernoulli independent loss process on edges (v_s, v_j) and (v_s, v_i) with the same expectation value ε. The source transmits all G_s symbols uninterrupted by the transmissions of v_j and v_i. After sending

the last packet v_s stops and will not transmit any packet till the end of the test (even though v_j and v_i may be not able to decode the complete generation). At this moment v_i sends the RR message to v_j. The communication between v_j and v_i is error-free. Upon the RR reception, the vertex v_j estimates N using eq. 3.26. In fig. 3.11, the value N normalized by G_s is shown as "Estimated value". Afterwards, v_j sends the recoded symbols to v_i as long as the rank of v_i grows. The difference of ranks on v_i in moments before and after the transmissions of v_j gives the number of useful symbols received from v_j. This value is shown as "Real value" in fig. 3.11. The value N is the estimation of it.

This test is repeated 1000 times for several values of ε. Confidence intervals with 95% confidence probability are shown on the figure as well (very small). The test was also repeated for different field and generation sizes but the results remained the same (see fig. 3.11). One can see that eq. 3.26 usually underestimates the real N.

3.4.2 RR origination and forwarding

GalaRes [38] is implemented on LLC layer. RR can be originated by any vertex after the timeout expiration, which is adaptively calculated basing on the average time that it takes the vertex to increase the rank of its coding matrix by one. RRs are not forwarded.

In our implementation, the destination is the only vertex that can originate RRs. Let it has n generations with indices $i, i+1, ..., i+n-1$ in its buffer. If it receives the coded symbol from the generation with index $j \geq i+n$ and the coding matrix of any generation with index $l \in \{i, i+1, j-k\}$ does not have the full rank, it sends RR for such generations. When the relay gets RR, it notes the oldest generation ID (The Oldest Generation IDentifier (OGID)), that is mentioned in the RR message. Then, it checks if it is selected by the RR sender as the responder to RR. If yes, it evaluates the number of useful information that it can forward using the retransmission rule of BRR-E and updates its forwarding plan. Afterwards, it decides if the RR messages should be forwarded. For this purpose, the following conditions should fulfill:

- the RR receiver has lower priority than the RR sender;
- the number of generations in buffer is greater than one;

- the RR receiver does not have all information requested;
- the generation ID of the received coded symbol is older than OGID.

If the relay decides to forward RR, it subtracts from the original RR the information that it can send itself. The RR message can be piggybacked or sent as a separate message if the forwarding plan is empty.

In order to prune the network flooding with RRs, ANChOR selects the RR forwarder and specifies it in the RR message. We propose several ideas for the selection:

- random selection among the vertices in $V^-(I(v))$;
- prefer the vertices $u \in V^-(I(v))$ with lower $\varepsilon_(u, v)$ (no randomness);
- prefer the vertices in $V^-(I(v))$ with higher priority (no randomness);
- prefer the vertices in $V^-(I(v))$ that send less linear dependent symbols (random selection).

Additionally, we implement the rule for the cancelation of the old RRs. If either the destination or the relay receives RR from the vertex with lower priority with an OGID that is older than the oldest generation ID on the receiving vertex, then such vertex broadcasts the ACK to abort the retransmissions of the old generations.

3.4.3 Retransmission response

In GalaRes [38], the RR sender v specifies the vertex that is eligible to respond on RR in the RR message. It selects such vertex randomly from the group of upstream vertices (closer to v_s) $Z \subseteq V^-(I(v))$. The group Z is formed basing on the loss ratios on edges $e \in I(v)$. The vertices $u \in V^-(I(v))$ with the best connection to v are preferred. The size of Z is limited by four. This approach allows reducing the amount of uselessly sent information.

GalaRes was designed for NB-PLC. In BB-PLC, the Logical link control layer Protocol Data Unit (LPDU) are sent in batches in a single MPDU. Therefore, LPDUs arrive also in batches, which can impair the estimation of the RR timeout proposed in [38]. We use ANChOR with G.hn, which is a BB-PLC protocol. Therefore, we follow another approach.

Basically, ANChOR does not forbid any vertex to respond on RR. Instead, it uses the following ACK mechanism. If the destination v_d receives the symbols of the generation that has already the full rank, it sends ACK. The receivers of ACK note the IDs of generations that are already completely received by v_d. If any of such relays receives the symbols of the generation that is already acknowledged by v_d or another relay with higher priority or this generation has the full rank, it sends ACK as well.

Fast retransmission response

We implement a mechanism that reduces the amount of sent RRs called the fast retransmission response. Consider the following example. The vertices v and w receives RR from the vertex u, whereby $p(v) > p(u)$ and $p(w) < p(u)$. Let vertex v have previously marked all generations that are listed in this RR as acknowledged and some of those generations on w are not acknowledged yet. It means that all information that is present on w is also present on v. Then, w sends additional coded symbols, which violates the BRR filtering rule. In accordance with it, if two vertices v and w has the same packet then the vertex with the highest priority should forward it. In our example, $p(v) > p(w)$. Thus, w should not send any additional symbols.

In order to avoid such a situation, in above, v immediately sends the message with the actual map of the acknowledged generations. It will not prevent w from sending the additional symbols. But it will prevent u from sending additional RRs.

3.5 Congestion control

Each vertex $v \in V$ can select the sending data rate, $d(v)$, to maximize the receiving rate of the vertices in the cooperation group $L(v)$:

$$R_{L(v)} \leq I(X_v; \cup_{u \in L(v)} Y_u^{(v)}), \tag{3.27}$$

where $I(X_v; \cup_{u \in L(v)} Y_u^{(v)})$ is the mutual information. The tradeoff between $d(v)$ and loss processes on edges $e = (v, u)$ $\forall u \in L(v)$ is considered in the calculation of the mutual information. But this strategy is often not sufficient for reliable communication. If the incoming flow data rate of vertex v, $\sum_{u \in V^-(I(v))} f_{uv}$, exceeds the outcoming flow data rate, $\sum_{u \in L(v)} f_{vu}$, then some data will be lost

on v due to receiving buffer overflow:

$$\sum_{u \in V^-(I(v))} f_{uv} \leq \sum_{u \in L(v)} f_{vu}. \tag{3.28}$$

The destination has a chance to revoke all data sent by v_s only if the data rate over each disjoint cut, R_S $S \subseteq \{V \setminus v_d\}$ $S \in \mathcal{C}$, does not exceed the upper bound data rate between v_s and v_d, R:

$$\begin{aligned} &R_S \leq R \; \forall S \in \mathcal{C} \\ &R_S = \sum_{v \in S} t_v \cdot I(X_v; \cup_{u \in L(v)} Y_u^{(v)}). \end{aligned} \tag{3.29}$$

For uncorrelated CTFs:

$$I(X_v; \cup_{u \in L(v)} Y_u^{(v)}) = d(v) \cdot (1 - \prod_{u \in L(v)} \varepsilon_{(v,u)}).$$

Thus, $d(v)$ should be calculated as the data rate that maximizes $R_{L(v)}$ (eq. 3.27) conditioned by eq. 3.29.

The algorithms 1 and 2 (section 3.1) give the optimal solution of this problem using the global information about the quality of the links. If the global information is not available, the vertices have to implement congestion control mechanism to guarantee the reliable communication (fulfilling eq. 3.28).

It is worth to mention that the congestion control is already a part of TCP. But it allows the control of the source only (since TCP is end-to-end). From eq. 3.29, we see that in the general case the congestion control on each vertex can be required. Thus, the delegation of this function to TCP does not solve the problem on lower layers. Therefore, we look at the protocols working on Data Link Layer (DLL).

In SOAR [32], the end-to-end ACK for source rate adaption is used. The sending rate of other vertices is not adapted. In this sense, though implemented on DLL, SOAR works similar to TCP. The destination sends the end-to-end ACKs at regular intervals over the shortest path. In this way, v_d indicates, how many codewords were received in the previous interval. The source updates the sending rate using this information.

In CCACK [30], each vertex adds the hash vector describing the coding matrix to each codeword. Each receiving vertex can modify the received hash vector with his own forming a cumulative ACK. Thus, the cumulative ACKs

are transferred abundantly in the network, which allows evaluating the differential backlog. The vertex $v \in V$ calculates it as the difference between the number of innovative codewords on v and the number of innovative codewords on all vertices $u \in V$ closer to v_d than v. The differential backlog is used to derive another metric serving as the trigger to send. It also allows the source finding out if the network already has sufficient data to deliver the current message block to the destination.

In MC^2 [28], the congestion control is realized with a credit based system. The credits interpret the number of codewords that certain vertex should send in the certain generation. The credit data is transferred separately from the codewords using the robust communication links.

ORCD [24] uses the backlog information of downstream vertices to reduce the network congestion. It develops the flow congestion metric that takes in regard the actual traffic load of each downstream vertex. Using this metric, the candidate relay is selected. Thus, the routing is always realized on the path with the smallest congestion. Therefore, if the source sending data rate does not exceed the out bound achievable data rate of the given network, the congestion does not take place. Despite optimality of this method, it requires an extensive feedback transmission, which makes it ineffective in most networks.

3.5.1 Congestion control in our work

We realize the congestion control through the neighbor overhearing. We assume that the buffer of each vertex can accommodate up to n, $n > 1$, generations. The actual buffer size depends on the Round Trip Time (RTT), RTO, PER, ACK window size, etc. TCP uses the end-to-end RTT, RTO, and PER in the formula of the sender rate calculation. We consider that in a big mesh network this can severely increase the latency. Therefore we implement the mechanism working on the per-hop basis. On each hop, the ORP uses hyperlinks (with the output of v and inputs of $u \in L(v)$) instead of unicast links. Therefore, in contrast to the unicast-TCP, the congestion control has to be realized for the multicast problem. And, differently, to the multicast-TCP, there is no reliability requirement for each unicast link separately.

We aim to reduce the protocol overhead for the congestion control. For this purpose, instead of ACKs transmission, we normally use the information already available in the headers of codewords. Thanks to the broadcast prop-

erty of the PLC channel, both up- and downstream vertices can receive the PLC signal. Surely, the downstream vertices do not need the contents of the received data. But they can use the header information.

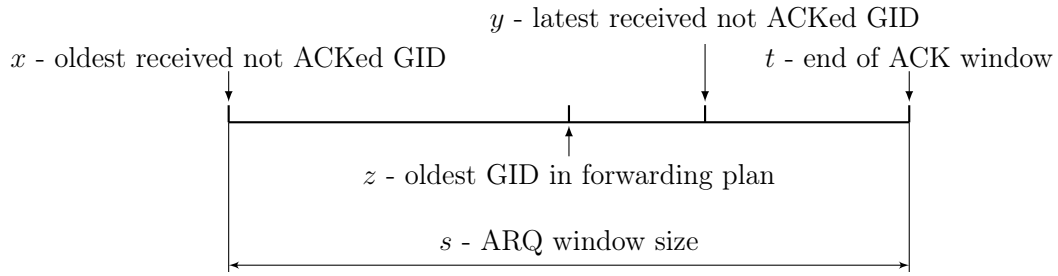


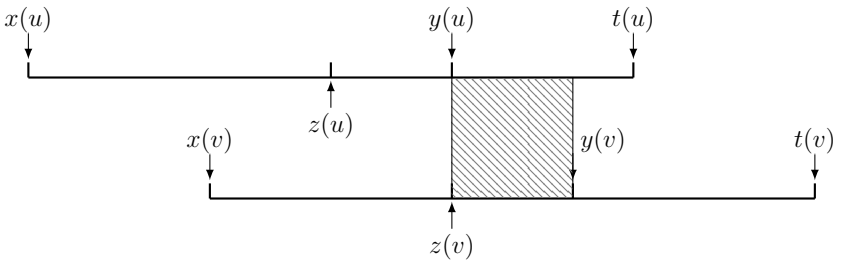
Figure 3.12: ARQ window. GID

Each vertex maintains its own ACK window and gathers the information about the ACK windows of the neighbor vertices reading the headers. The ACK window can be described with five values as shown in fig. 3.12. The size of the ARQ window, s, is measured in the number of generations. It is constant for all vertices and does not change at runtime. The end of ACK window, t, is defined through the oldest received GID that is not acknowledged yet and the ARQ window size. The vertex can keep in the buffer not more than s generations simultaneously. Therefore, $y < x + s$. Using BRR-E, the vertex finds out a number of symbols to be sent for each generation. The oldest GID in the forward plan, z, indicates that all symbols for the generations with IDs starting from x and up to $(z-1)$ were already sent. Thus, the generation with GID equal z is the current generation to transmit. The vertex can send the symbols of at most $(y - z)$ generations.

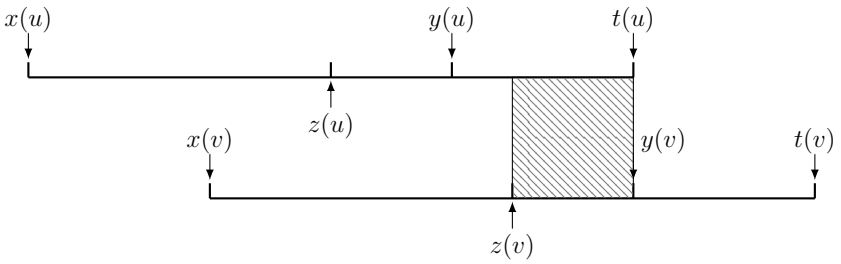
Definition 3.5.1. The number of generations that the vertex can send is called the *transmission window*.

Definition 3.5.2. The number of generations between $x(v)$ and $y(v)$ is called the *reception window*.

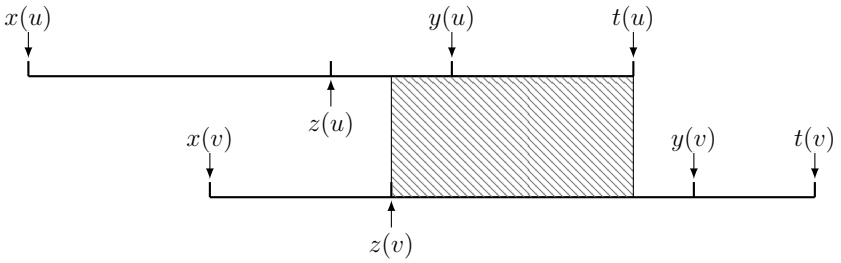
If the vertex receives symbols from the generation with ID $i > y$ and there is no more space in the ACK window $(y = t - 1)$ then it drops the oldest received generation (data may be lost). Thus, the vertex prefers new information to the old one. Although data drops are possible, they are heavily eliminated with the congestion control mechanism. The transmission window is limited



(a) Normal forwarding ($z(v) = y(u)$)



(b) Speeded up forwarding ($z(v) > y(u)$)



(c) Slowed down forwarding ($z(v) < y(u)$)

Figure 3.13: Transmission window dependence on neighbour ARQ windows. $p(u) > p(v)$. Transmission window of vertex v is shaded

not only by $(y - z)$, i.e. own ARQ window but also by the ARQ windows of the vertices in the cooperation group.

In the following, notations $x(v)$, $y(v)$, $z(v)$ and $t(v)$ denote x, y, z and t (see fig. 3.12) for vertex v. In figs. 3.13a - 3.13c, we show three possible cases of relationship of ARQ windows between two vertices v and u. Here, we let u be in the cooperation group of v. The dashed rectangles show the transmission window of v. The start of the transmission window is always defined by $z(v)$. The end of the transmission window is minimum GID of $t(u)$ and $y(v)$. If the size of the generation is much bigger than the size of the cooperation groups and the vertices send the coded symbols as separate data units to the physical layer then the cases in fig. 3.13b and fig. 3.13c will rarely occur. But in BB-PLC, we can send a large number of coded symbols in one PHY frame. As a result, the start of the transmission window of v may be not equal to the end of the reception window of u. The speeded up the case (fig. 3.13b) is possible when due to the burst of noise a bunch of packets on the way from v to u is erased. The slowed down case (fig. 3.13c) happens very often. When one of the vertices $V^-(I(u))$ have already sent its data before the vertex v then u already has some data from the generations, which v did not send yet. From these figures, you can see that in the case of slowed down forwarding, the vertex v is allowed to transfer more data than in the normal case. In this way, the vertices with older data are allowed to send more data. In the speeded up case, the transmission window of v is limited by the end of the ACK window of u that decreases the transmission window of v. In this way, the congestion is avoided.

The transmission window, m, can be also bounded by the maximum size of MPDU, n_{max} (unit - number of LPDUs). In general, the size of the transmission window can be calculated as follows:

$$\begin{aligned} m(v) &= \min\{n_{max}, k\}, \\ k(v) &= \sum_{i \in Y} k_i(v), \\ Y(v) &= \{z(v), z(v)+1, ..., \min\{y(v), t'(v)\}\}, \\ t'(v) &= \min_{u \in L(v)} t(u), \end{aligned} \tag{3.30}$$

where $k_i(v)$ is the number of symbols to send from the generation with ID i on vertex v and $Y(v)$ is the range of GIDs falling into the transmission window

of v.

Eq. 3.30 defines the main rule for the congestion control in ANChOR. There are also a few auxiliary rules that improve the performance. For example, if v receives the RR message and it has no request data to send then it refuses from sending until one of the following happens: it receives the requested data from the vertices with lower priority or it receives the RR message with newer requested data that it already has, or it receives ACK from the vertices with higher priority for the requested data. Also, we limit the data rate of the source vertex to its priority. Remember that the vertex priority describes the upper bound on the effective reception data rate of the destination. Thus, the network cannot support a bigger data flow than defined by the source vertex priority.

3.6 Network initialization

Consider the network graph $\mathcal{G}(V, E)$, which is stationary in terms that the set of vertices V, the set of edges E and associated with the edges $e \in E$ packet loss ratios ε_e are time-invariant. Considering a fully meshed scenario, each vertex $v \in V$ is connected to each vertex in $u \in V \setminus v$ with an edge $e = (v, u)$ and each v can receive a packet from each u with a probability $(1 - \varepsilon_e)$. It stays true also for big networks, where ε_e between distant pairs of vertices approaches one.

We assume that initially no data or control information has been exchanged yet and no link quality has been estimated yet anywhere in the network. At this moment, ANChOR cannot use all its functions because $\mathcal{G}(V, E)$ cannot be constructed. The period from the initial time point until the full activation of ANChOR functions is called the initialization phase. In this phase, the vertices $v \in V$ construct $V^-(I(v))$, $V^+(O(v))$, build reception maps for edges $e \in I(v)$ and learn the reception maps for edges $e \in O(v)$, calculate the filtering coefficient, learn their own priorities and the priorities of their neighbours and form the cooperation groups $L(v)$.

First, any vertex has the empty cooperation group. Only the source v_s having the empty cooperation group is allowed to send data. When the relays receive the coded symbols from v_s, their broadcast the Network Discovery Message (NDM). Each receiving relay sends it with certain probability p. In

the header of NDM, the vertices include the Time To Live (TTL) that indicates the maximum number of times that NDM can be repeated. NDM is not repeated if it reaches the vertex v with $L(v) \neq \emptyset$. Also, the NDM header contains the sender vertex ID and its priority. Initially, all vertices have a very low priority that equals the half of the smallest expected sending data rate. Only the destination v_d has initially high priority. In fact, the priority of v_d does not change. It is always greater than the priority of any other vertex. When NDM reaches the destination and it repeats NDM, the v_d neighbors learn about the presence of the destination in their neighborhood. It allows increasing the priorities of these vertices. As one can see in eqs. 3.8 and 3.11, the vertices also need the loss ratios to calculate the priority. Initially, all loss ratios equal zero. Only when the vertices gather sufficient amount of statistics, they update the loss ratios, which influences the priorities as well. When the loss ratios are zero, in accordance with eqs. 3.8 and 3.11, the priorities of the vertices in the neighborhood of v_d are equal their sending data rates. When the vertices with the increased priority receive NDM again, they relay it indicating their neighbors the presence of neighbor vertices with high priority. Such relays increase their priorities as well. In this way, all vertices initialize their priorities.

When the vertex priority grows, other vertices can include them in their cooperation groups. So first, the neighbours of the destination $v \in V^{-}(I(v_d))$ add v_d to their cooperation groups $L(v)$. When the priorities of all vertices are initialized, the most of the vertices have non-empty cooperation groups. When $L(v) \neq \emptyset$, v can relay the received coded symbols. After receiving certain number n of coded symbols (both erased and not erased), the loss ratios are updated to the first estimation of real loss ratios. It stimulates the recalculation of the priorities, filtering coefficients and updating of the cooperation groups.

Assuming that $\mathcal{G}(V, E)$ is stationary, the initialization phase ends when all loss ratios, filtering coefficients, priorities and cooperation groups become stationary. We determine the duration of initialization phase observing the graphs of these routing metrics.

3.7 Formation of the cooperation groups (coalitions)

In accordance to algorithm 1, the vertex $v \in V$ includes in its cooperation group $L(v)$ all neighbour vertices with higher priority (it also follows from the definition 3.1.1). In a large and dense network, the size of $L(v)$ can grow fast. As a result, the filtering coefficients for all $u \in L(v)$ increase and each vertex broadcasts the smaller number of symbols on average. This explains the increased inaccuracy of BRR-E as shown in fig. 3.6. Therefore, it makes sense to have a method that can limit the size of $L(v)$.

Let the current coalition of v is $L(v)$ and the priority is $p(v)$. Let u with higher priority $p(u) > p(v)$ broadcasts a message that carries the value $p(u)$ in its header, v receives this message and decides if u should be added to $L(v)$. We propose to weight the contribution of u when it enters $L(v)$ by the difference $(p'(v) - p(v))$ where $p'(v)$ is calculated in same manner as $p(v)$ (eq. 3.8) but for the bigger cooperation group $L'(v) = L(v) \cup u$. The vertex v adds u to $L(v)$ if the following condition fulfills:

$$p'(v) - p(v) > b, \tag{3.31}$$

where $b > 0$ is some constant. As far as $p(v)$ is analog to the effective data rate between v to v_d, the above equation issues a threshold for any new vertex entering the coalition that bases on the potential increase in the data rate.

If u enters the coalition of v, the vertex v refines $L(v)$. If the vertex u has higher priority than at least one vertex in $L(v) \setminus u$ then some vertices $w \in L(v) \setminus u$ may be pushed out of it. The refinement algorithm works as follows.

Algorithm 8. Refinement of $L(v)$

1 Remove all vertices from $L(v)$ with lower priority then $p(v)$
2 $L(v) \leftarrow L(v) \setminus w$, where w is the last vertex in $L(v)$
3 $p'(v) \leftarrow p(v)$
4 Calculate $p(v)$ (using $L(v)$)
5 **if** $p'(v) - p(v) > b$ **exit**
6 **else** go to step 2

Remember that the vertices in $L(v)$ are sorted in the descending order of their priorities (definition 3.1.1). Each vertex is self-responsible for the formation of its coalition. Any vertex u can join the coalition of any other neighbor vertex if it broadcasts any ANChOR message since each ANChOR message includes the sender ID and its priority.

So, the coalition formation is decentralized, which is most suitable for mesh networks.

3.8 ANChOR header

The ANChOR header consists of the core and optional parts. Each message of the ANChOR protocol and also the coded symbols are appended to the ANChOR header. In cases when the underlying protocol packs the ANChOR messages/coded symbols in one PDU (e.g. MPDU in G.hn, IEEE1901, UPA), it makes sense to split the ANChOR core header into parts. The first one is attached to each message/coded symbol and the latter one only once per the PDU of the underlying layer. The optional part of the ANChOR header is also attached only once per the PDU of the underlying layer. The optional part contains the RR message and RPI. The header structure so as the size of the header fields is shown in table 3.2.

In table 3.3, we show the example calculation of the ANChOR header size for the setup as shown under the table.

In accordance to table 3.3, in this example the header of 39 bytes is added to each coded symbol and 4 bytes can be added only once per the PDU of the underlying layer. In G.hn, the MPDU can accommodate up to 376 LPDUs of 540 bytes size. Then, the main overhead is created by the first part of the ANChOR header core. The size of RPI and RR with the hash vector or the data for the min-max RR method is less than 540 bytes. Thus, these optional parts fit into one LPDU.

3.9 Communication of protocol information

ANChOR has several kinds of management information: core information, NDM, RR and RPI. As shown in the ANChOR header description (table 3.2), these information is transferred in the ANChOR header. The core information corresponds to the header core parts. In comparison to the core information,

Table 3.2: ANChOR header structure

Field name	Size / bits
	Core part 1
Coding vector	$G_s \cdot m^{(1)}$
Generation ID	8
CRC block	32
Symbol sequence number	16
	$\Sigma_1 = G_s \cdot m + 56$
	Core part 2
Priority	16
Sender ID	8
RPI flag	1
RR flag	1
Network discovery Flag	1
	$\Sigma_2 = 27$
	RPI
Filtering probabilities $p_f(v,u)$ for all $u \in V^+(O(v))$	$\lvert V^+(O(v))\rvert \cdot 8$
Receiving maps m_e for all $e \in I(v)$	$\lvert I(v)\rvert \cdot M^{(2)}$
Map of acknowledged generations	k
	$\Sigma_3 = \lvert V^+(O(v))\rvert \cdot 8 + \lvert I(v)\rvert \cdot M + k$
	RR message
Use all vectors	1
Use hash vector	1
Use min-max method	1
Coding vectors	$G_s^2 \cdot m \cdot k^{(3)}$
Hash vector	$G_s \cdot m \cdot k$
Rank of coding matrices	$8 \cdot k$
Map of uncoded symbols	$G_s \cdot k$
Map of pivot symbols	$G_s \cdot k$
RR forwarder	8
$\Sigma_4 = G_s^2 \cdot m \cdot k + 11$, $\Sigma_5 = G_s \cdot m \cdot k + 11$, $\Sigma_6 = (2 \cdot G_s + 8) \cdot k + 11$	
	$\Sigma^{1/2/3} = \Sigma_1 + \Sigma_2 + \Sigma_3 + \Sigma_{4/5/6}{}^{(4)}$

(1) $m = log_2(q)$ - logarithm of the field size

(2) M - maximum size of the receiving map

(3) k - actual number of generations in the buffer

(4) RR contains either the coding vectors or the hash vector or the data for min-max method

Table 3.3: Size of ANChOR header parts

Sum values from table 3.2	Σ_1	Σ_2	Σ_3	Σ_4	Σ_5	Σ_6	Σ^1	Σ^2	Σ^3
Size / bytes	39	4	191	4098	130	38	4332	364	272

$G_s = 32$, $q = 256$, $M = 500$, $k = 4$, $|V^+(O(v))| = 3$, $|I(v)| = 3$

NDM, RR, and RPI are attached relatively rarely. If no new vertices join the network in the runtime, NDM flag is set only during the initialization phase (see section 3.6). RR is attached only several times per a generation (see section 3.4.2). RPI is normally attached when the map of acknowledged generations is updated, i.e. once per generation. In addition, RPI can be attached with certain probability p when any coded symbol is sent. Increasing p, it is possible to control the frequency of RPI updates on the neighbor vertices. If the channel is not stationary then the frequency has to be high enough to make ANChOR react on channel changes. The exact value of p should be obtained resolving the tradeoff between the increased protocol overhead (RPI size is 191 bytes for example in table 3.3) and the increased number of useless redundant symbols that appear due to slow updating of the filtering coefficients and the vertex priorities.

We distinguish between several kinds of messages, that can be sent with ANChOR. Each ANChOR message consists of the ANChOR header (table 3.2) and the body field. The body may contain the coded symbol or be empty. If the body field is not empty, we refer to the ANChOR message as the data message regardless of the contents of the ANChOR header. If the body is empty, the message is named in accordance with its main purpose. The message with the activated network discovery flag we call NDM (NDM). If NDM flag is not set and the header contains RR then the message is called the RRM. Eventually, if both NDM and RR flags are unset then the header contains RPI and the corresponding message is referred to as Routing Protocol Information Message (RPIM). The messages with deactivated NDM, RR and RPI are not sent unless the body field is not empty.

3.10 ANChOR simulation

As previously mentioned, we developed the discrete-event simulator SNS that implements the broadcast packet erasure channel and allows setting up of ar-

bitrary network topologies. It also implements a greedy traffic generator. It is most suitable for analysis of the maximum achievable data rate that can be sustained by the network. When using the greedy traffic generator, the ANChOR connects to the application layer containing this traffic generator with a callback function that can trigger the generation of new data. The source vertex uses this function each time when its queue is not full. In this way, it is guaranteed that the source always has data to send. In the same time, the queue overflow is excluded. The channel access is realized using the principle of slotted ALOHA. The duration of the current slot equals the duration of the current packet transmission. At the beginning of each slot, all vertices that have data for transmission get the channel access with probability $1/n$ where n is the number of vertices. In difference to slotted ALOHA, collisions are avoided with central coordinator function.

On basis of SNS, we implement ANChOR and the evaluator of the maximum achievable data rate (see section 2.3).

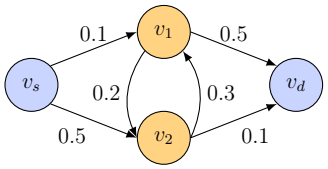


Figure 3.14: Diamond network with erasure ratios. $d(v) = 1$ Mbps $\forall v \in \{v_s, v_1, v_2\}$

. v_s and v_d are also referred to as v_0 and v_3 respectively.

3.10.1 ANChOR information in real time

Using SNS, we run simulations for several topologies. First, we show all information that is gathered by ANChOR in runtime on the example of the diamond network (see fig. 3.14). Table. 3.4 we gives the simulation parameters. These simulation parameters remain constant throughout all simulations.

In fig. 3.15, the loss ratios on the edges $(2, 1)$ and $(2, 3)$ are shown. They are calculated using the reception maps that the vertices v_1 and v_3 send to v_2 in RPIM. The horizontal axis gives the time. We evaluate the network performance counting the number of the received symbols by the destination to the total number of all sent symbols by all vertices N. If all sending vertices use the same sending data rate and all sent packets have equal size then it takes

Table 3.4: Simulation parameters

Parameter	Value
Maximum number of generations in a buffer	10
G_s	100
$d(v)\ \forall v \in V$	1 Mbps
Number of buffered generations	2
q	2^8
CCACK levels	2

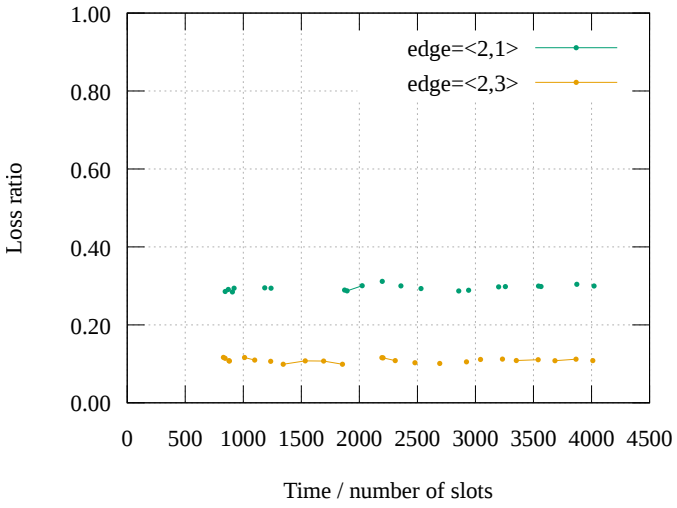


Figure 3.15: Loss ratios (topology in fig. 3.14)

the same time to send each packet. We set the sending data rate of each vertex equal $d = 1Mbps$ and the size of each coded symbol is equal. Therefore, the value N can be used for the time metric in fig. 3.15.

We can see the points in fig. 3.15 with a period equal to sending of approx. 500 symbols. At these moments of time, the vertex v_2 gets RPI from v_1 and v_3. Surely, such period can be too large on networks with a fast changing channel. Then, the frequency of RPI sending should be increased as advocated in section 3.9.

Using the receiving maps, the vertices v_1 and v_2 can calculate their priorities (see fig. 3.16).

The vertex v_2 has better connection to the destination than v_1. Therefore, the priority of v_2 should be higher. It equals $p(v_2) = d \cdot (1 - \varepsilon_{(v_2,v_3)}) = 0.9Mbps$.

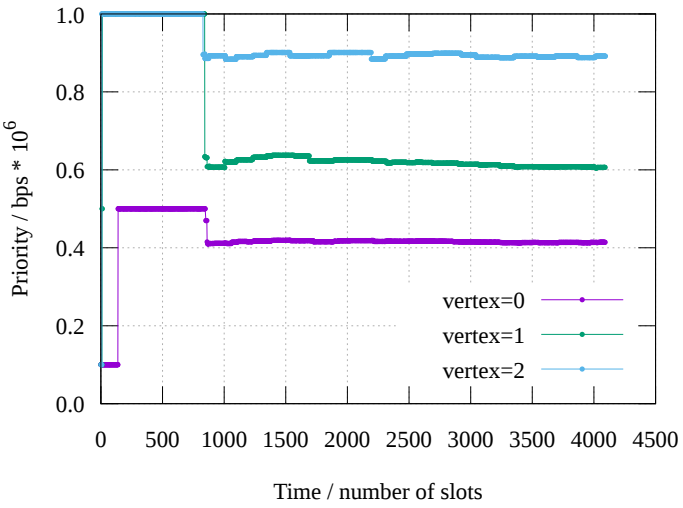


Figure 3.16: Priorities (topology in fig. 3.14)

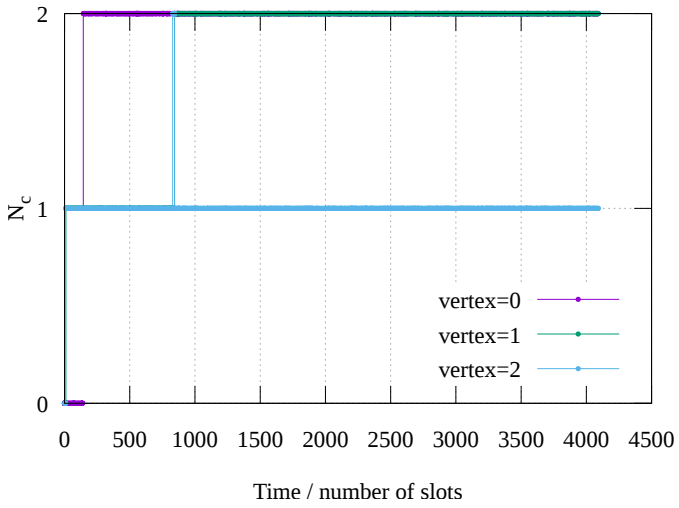


Figure 3.17: Coalitions (topology in fig. 3.14)

The priority of v_1 is a bit higher than $d \cdot (1 - \varepsilon_{(v_1,v_3)})$ because it recognizes that $p(v_2) > p(v_1)$. This motivates v_1 to cooperate not only with the destination but also with v_2. Thus, v_1 uses not only the direct path but also the path over the vertex v_2, which allows the increasing of its priority. For $p(v_1)$ calculation, v_1 needs not only $\varepsilon_{(v_1,v_2)}$) and $\varepsilon_{(v_1,v_3)}$) but also $p(v_2)$.

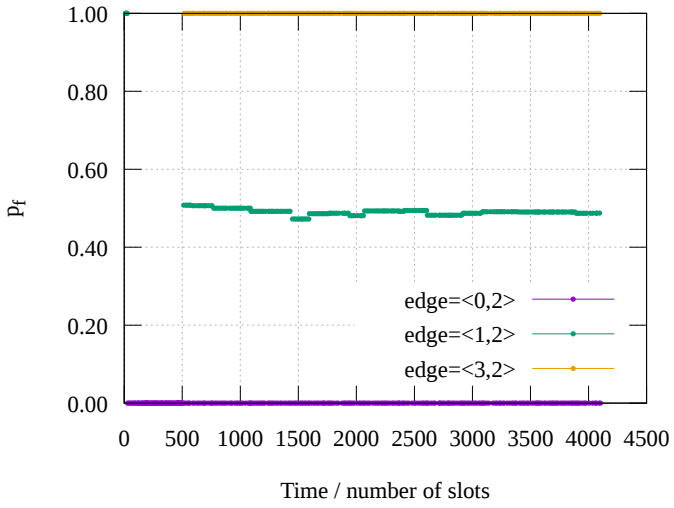


Figure 3.18: Filtering probabilities (topology in fig. 3.14)

Simultaneously with priority calculation, the vertices form their coalitions (see algorithm 1). In fig. 3.17, the sizes of the cooperation groups N_c are shown for each vertex separately. As far as $p(v_1) < p(v_2), p(v_3)$ and $p(v_s) < p(v_1), p(v_2)$, $N_c = 2$ for v_s and v_1.

Basing on the reception maps, the vertices can also calculate the filtering probabilities p_f (fig. 3.18). When v_s is sending, since $p(v_1) < p(v_2)$, v_2 forwards all the received information, i.e. $p_f = 0$. When v_1 is sending then v_2 forwards only that part, which is not received but v_d. So, $p_f = \varepsilon_{(v_1,v_d)} = 0.5$. Although in fig. 3.14 all vertices are connected with uni-directional edges, in SNS we consider all edges to be bi-directional and symmetric unless different quality if different direction is specified, like between v_1 and v_2. For this reason, this figure also includes p_f of v_2 for data comming from v_d. As far as $p(v_d) > p(v_2)$, $p_f = 1$. Although in fact, v_d sends no data.

In all figures above, the ANChOR variables (coalitions, priorities, filtering coefficients) stabilize after about 2000 time slots. This period can be reduced by increasing the sending frequency of RPI as proposed in section 3.9.

Running the simulation for 20000 time slots, we evaluate the achievable data rate (see fig. 3.20a).

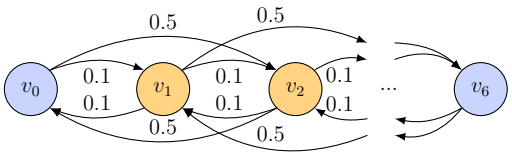
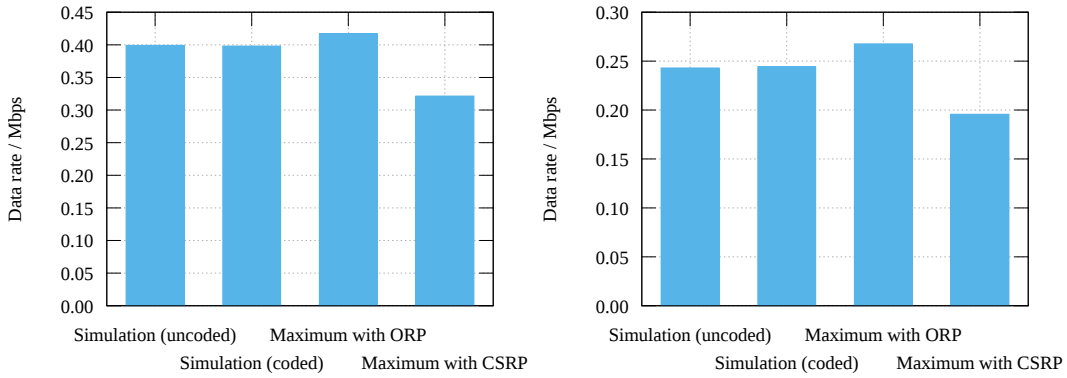


Figure 3.19: Long line topology with erasure ratios. $d(v) = 1$ Mbps $\forall v \in V$



(a) Topology in fig. 3.14

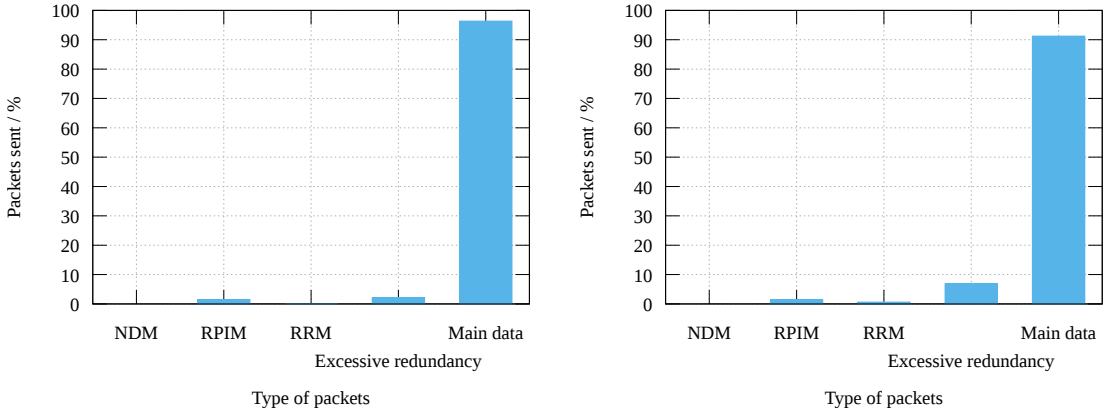
(b) Topology in fig. 3.19

Figure 3.20: Data rates

The maximum achievable data rate with ORP and with CSPR are evaluated as shown in section 2.3. We present two types of the data rate that are obtained by means of simulation. The coded data rate D_c corresponds to the rank of all generations that are received by the destination multiplied by the size of the uncoded symbol. The uncoded data rate D_u is the effective data rate that is perceived by the user on the application layer. If the protocol does not guarantee the full data decoding then D_c exceeds D_u. In fig. 3.20a, $D_u \approx D_c$, which is a sign of complete decoding of the original data. D_u is just slightly smaller than D_c, which is a result of the simulation stopping rule. We specify the simulation duration as the total number of sent symbols. As a result, the last generations may be not fully decoded.

Although D_u exceeds the maximum achievable data rate with CSPR by approx. 18%, it is about 6% below the maximum achievable data rate with

ORP. In fig. 3.21a, we show all types of sent messages that consume the channel resources.



(a) Topology in fig. 3.14

(b) Topology in fig. 3.19

Figure 3.21: Types of sent ANChOR messages

Remember that ORPs are not widely applied yet because many of them suffer from extensive feedback and sending of excessive redundant data. The feedbacks in ANChOR are presented with the group of NDM, RPIM and RRM messages. ANChOR manages to spend only 3% of the channel resources for all them in total. Also, about 3% is spent for sending of the excessive redundancy. We calculate this as follows. Let n_s is the total number of sent messages, n_r is the total number of uncoded symbols by the destination and n_f is the total number of NDM, RPIM and RRM messages. Using eq. 2.3, we calculate the upper bound of the achievable data rate d'_r. If all senders have equal sending data rate d_s then $n_r/n'_s = d'_r/d_s$, where n'_s is the average optimal number of total sent messages. We calculate it as $n'_s = (d_s \cdot n_r)/d'_r$. Then, the number of the excessive redundant symbols n_e can be calculated as follows:

$$n_e = n_s - n_f - n'_s.$$

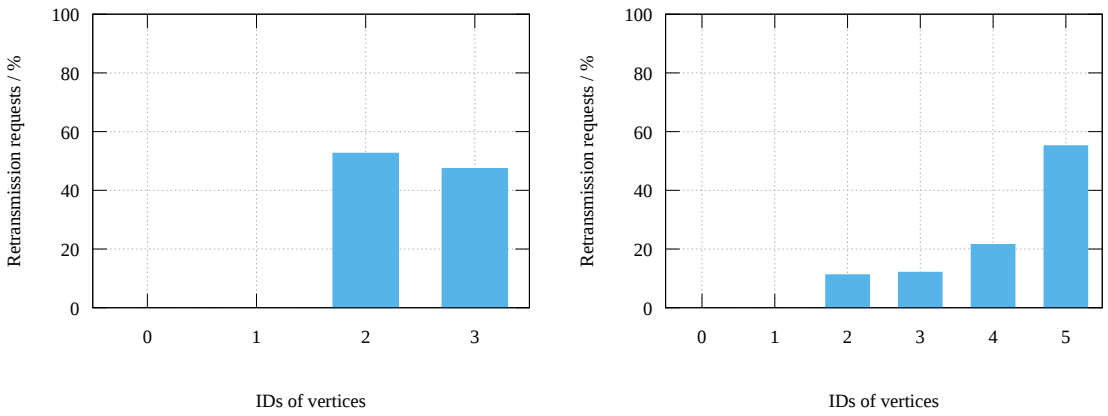
There are two reason for $n_e \neq 0$. First, BRR-E bases on the estimations of the expectation values of multiple random variables (RVs), which leads to a certain inaccuracy. Second, the BRR-E redundancy rule adds the proposed amount of redundant symbols. And, even if estimations of the expectation values of RVs that are required for the coding rate calculation are very accurate, it is not possible to avoid the excessive redundancy. We explain it by the following

example. Say, the vertex v targets to deliver m symbols to its cooperation group $L(v)$. Let the probability that each separate symbol will be received by at least one of $u \in L(v)$ equals p (see the redundancy rule in section 3.3.2). Assuming the channel to be memoryless, the number of received symbols by $L(v)$ can be described with the binomially distributed RV M. Let v add the redundant symbols to increase the probability that $L(v)$ receives m symbols. Thus, v sends in total $n > m$ coded symbols. The number of excessive redundant symbols is also described with RV $K = \max(M - m, 0)$. The expectation value of K, $\overline{K}$, can be calculated as follows:

$$\overline{K} = \sum_{i=1}^{n-m} i \cdot \mathcal{B}(n, m+i, p), \tag{3.32}$$

where $\mathcal{B}(x, y, p) = C_x^y \cdot p^y \cdot (1-p)^{x-y}$ and $C_x^y = \frac{x!}{y! \cdot (x-y)!}$. It is clear that as long as $n > m$, i.e. the vertex v adds any redundancy, $\overline{K} > 0$.

We also show the percentage of the sent RRMs by all vertices (see fig. 3.22a).



(a) Topology in fig. 3.14

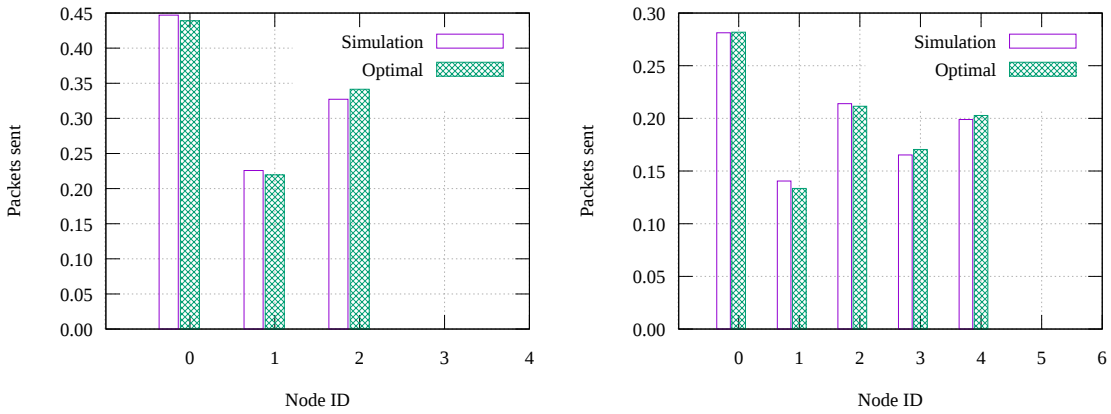
(b) Topology in fig. 3.19

Figure 3.22: Distribution of RRMs

Remember that only the destination can originate RR in ANChOR. Other vertices can only forward it (see section 3.4.2). Still, the relay v_2 sends more RRMs than the destination v_3. In fact, the relays can forward and replicate RRMs. The relay replicates RR when it receives a coded symbols from a vertex with the smaller priority that belongs to the generation with newer ID than the oldest generation ID that is stated in RRM.

Eventually, in fig. 3.23a we compare the optimal and actual TDMAPs.

Remember that the optimal access plan corresponds to the upper bound on the achievable data rate (section 2.3). The actual TDM access plan we obtain as follows. We count the number of sent data messages by each vertex and normalize these values by the total number of sent data messages. In the network with the same sending data rate for all vertices, these values are analog to $T = \{t_0, \ldots, t_{|V|-1}\}$ in eq. 2.3.



(a) Topology in fig. 3.14

(b) Topology in fig. 3.19

Figure 3.23: Comparison of the optimal and actual TDMAPs

The simulation and analytical values almost coincide, which demostrates the efficiency of all BRR-E rules together.

3.10.2 Selection of the coding rate

The BRR-E redundancy rule advocates adding as many redundant symbols as it is expected to be lost on average. Nevertheless, as shown in eq. 3.32, the more redundancy is added the greater is the expectation amount of the excessive symbols, $\overline{K}$. Targeting to decrease $\overline{K}$, we can reduce the number of redundant symbols. Surely, it causes the intensifying of RRs. But if RR overhead is not big in comparison to the sending of the excessive symbols then the overall throughput can increase. For example, using min-max RR approach in the example calculation of table 3.3, the RR contents comprises 38 bytes only.

In PLC, RRM can have a substantial overhead created by PHY if RRM is sent in a single PHY PDU. The overhead consists of PHY header, preamble, interframe gaps, etc. In this case, it can be useful to increase the redundancy above the expected number of symbols to be lost on average. We realize it as

proposed in appendix E.

In SNS, the size of all ANChOR message is assumed to be the same. Therefore, the sending of one data message and one RRM consumes the same amount of channel resources. Nevertheless, it still can make sense to manipulate the coding rate in SNS because $\overline{K}$ and the number of RRMs n_{RRM} are connected with a complex function. By means of simulation, we solve the following optimization task:

$$\begin{aligned} \min_{N,\beta} \quad & N = \overline{K} + n_{RRM} \\ \text{s. t.} \quad & c' = \left(1 + \frac{1-c}{c} \cdot \beta\right)^{-1}, \\ & \beta \geq 0, \end{aligned} \tag{3.33}$$

where c is the coding rate that is specified in the redundancy rule of BRR-E and c' is the coding rate applied in the simulation.

Again, we use the topology in fig. 3.14 as an example. Note that $\beta = 0$ means adding no redundancy and $\beta = 1$ means $c' = c$. In fig. 3.24, we show the

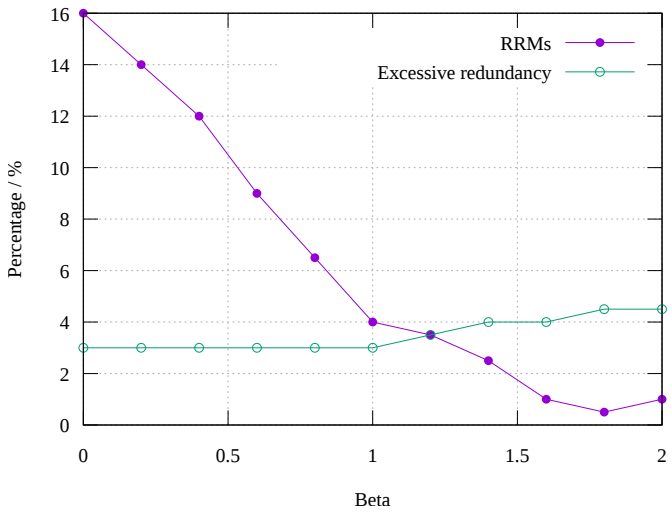


Figure 3.24: Percentage of RRMs and excessive redundancy (see topology in fig. 3.14)

percentage of RRMs and excessive redundancy (analog to data in fig. 3.21a). Increasing β we add more redundancy that reasons the reduction of n_{RRM}. It is interesting to observe that $\overline{K}$ does not change a lot even when $\beta = 2$. The answer lies in ANChOR ARQ. Even though all vertices add two times more

redundant symbols they do not send all of them. The vertices with higher priorities are constantly acknowledging the generations with the full rank and the generations that are acknowledged by the vertices with higher priority. As soon as the vertex gets the ACK for certain generation, it drops all outstanding data. Such performance is typical for protocols that can send each ANChOR message as a separate PHY PDU. In BB-PLC, it is not the case that can result in greater $\overline{K}$. Fig. 3.25 shows the data rate (uncoded) that complies with the

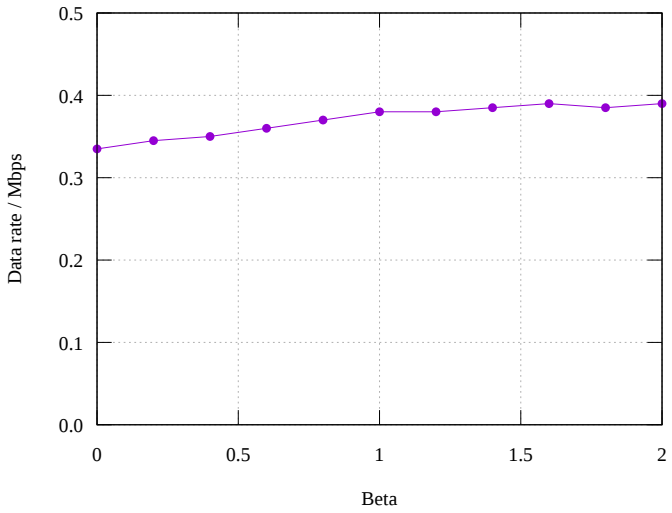


Figure 3.25: Data rate uncoded (see topology in fig. 3.14). Changing coding rate

results in fig. 3.24. The data rate grows significantly only for $\beta < 1$.

3.10.3 RPI broadcasting frequency

Increasing the frequency of RPI broadcasting p improves the speed of ANChOR adaptation to the changing channel conditions. It makes ANChOR more precise in routing decisions in order to increase the data rate. At the same time, the number of RPIMs increases that reduces the data rate. We analyze the data rate achievable with ANChOR as a function of p.

The SNS allows configuring the network graph only once before the simulation start. Thus, it does not change the channel conditions. With SNS, the only one transient phase is at the network start up. Remember that initially, the vertices do not know the network graph and the qualities of the links. It is analog to say that initially in the network acts a source of strong noise that

makes all vertices unreachable to each other. Then, instantly, this noise disappears. Therefore it is valid to say that the network startup is equivalent to significant changes on the channel.

In the initialization phase (section 3.6), first the network graph is constructed. Then, the receiving maps are gathered, which allows calculating the priorities, filtering coefficients and forming the coalitions. The duration of the initialization phase can be visually defined in figs. 3.15 - 3.18 as the period from the simulation start that it takes to stabilize all ANChOR metrics. It equals approx. 2000 time slots. In the example in figs. 3.15 - 3.18, we set $p = 0$. In SNS, it is equal the probability of attaching RPI to the ANChOR header. For greater p, the initialization phase is shorter. Remember that RPI is also added to each RRM and NDM. Also, it is unconditionally attached to the ANChOR header when the map of acknowledged generations is updated.

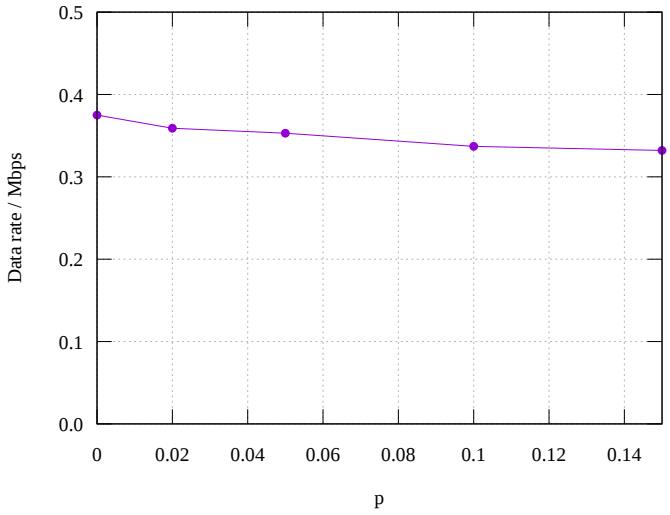


Figure 3.26: Coded data rate (see topology in fig. 3.14) versus the probability p of attaching RPI to the ANChOR header

We set the simulation duration to 2000 time slots. Then, we run serveral simulations for different p observing the coded data rate at the destination. For $p = 0$, the simulation duration equals the initialization phase. For larger p the initialization phase finishes sooner. The observed coded data rate evaluates the impact of different p in the transient phase. As far as the simulation duration is rather short, we repeat it 15 times and plot the average values in fig. 3.26. From this figure, we see that while increasing the RPI broadcasting frequency,

the data rate at the destination decreases. Thus, in this example it makes no sense to send RPIMs additionally to the obligatory cases.

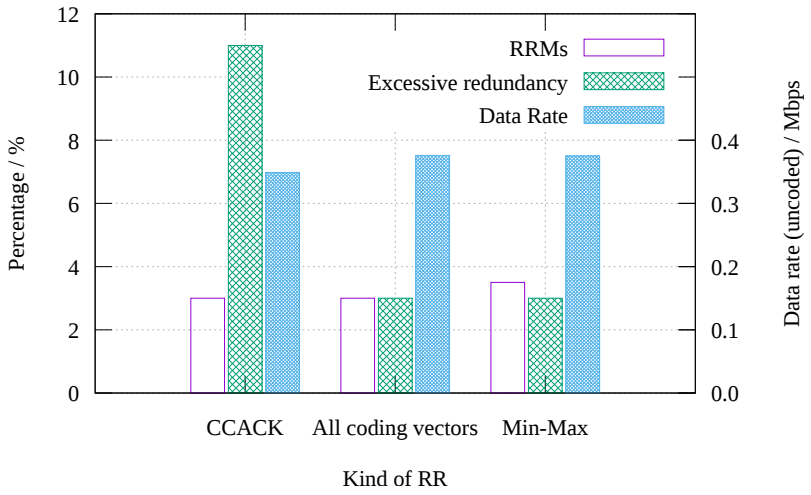


Figure 3.27: Selecting the kind of RR contents (see topology in fig. 3.14)

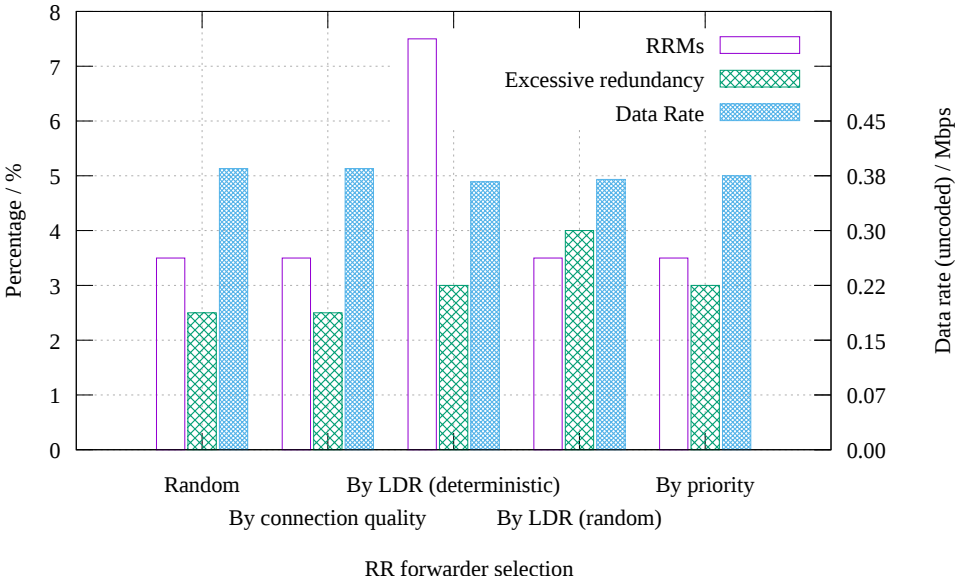


Figure 3.28: Selection of RR forwarder (see topology in fig. 3.14)

3.10.4 RR contents

The effectiveness of different ARQ strategies can be evaluated with the SNS as well. We compare three strategies: sending all coding vectors, CCACK

approach, and min-max approach. In fig. 3.27 there are two axis. The left one corresponds to the percentage of RRMs and excessive redundancy. The right one shows the scale for the data rate. The most accurate RR contents are offered when the coding matrices are completely attached to RRM. But it creates significant overhead (see table 3.3 and appendix G). Min-max method gives almost the same performance as attaching all coding vectors at very small overhead cost (see table 3.3).

3.10.5 Selection of RR forwarder

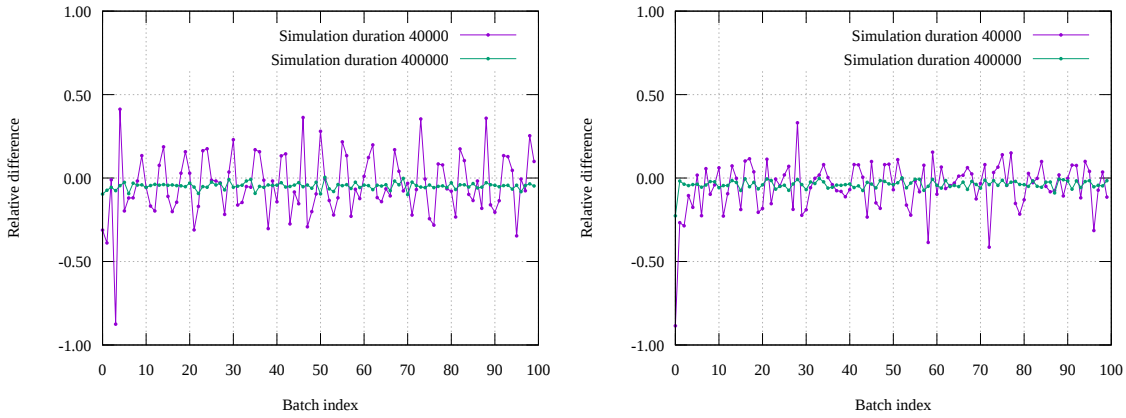
The receiver of RRM checks can find in RR info if the RR sender specified him as the RR forwarder. The forwarder selection can be realized in a different fashion as described in section 3.4.2. In fig. 3.28 one can see the same performance metrics as before. There is just a slight difference of the data rate. The best results are achieved by the random selection among the vertices in $V^-(I(v))$ or by the connection quality on the links between the RR sender and the vertices in $V^-(I(v))$.

3.10.6 ANChOR stability

First, we evaluate the ANChOR stability observing the reception data rate at the destination. The maximum achievable data rate at the destination with ANChOR equals the upper bound on the achievable data rate between the source and the destination that can be calculated with eq. 2.3. In fig. 3.29a, we show the relative difference between the actual data rate R' obtained from the simulation and the upper bound on the achievable data rate R:

$$\lambda = (R' - R)/R. \tag{3.34}$$

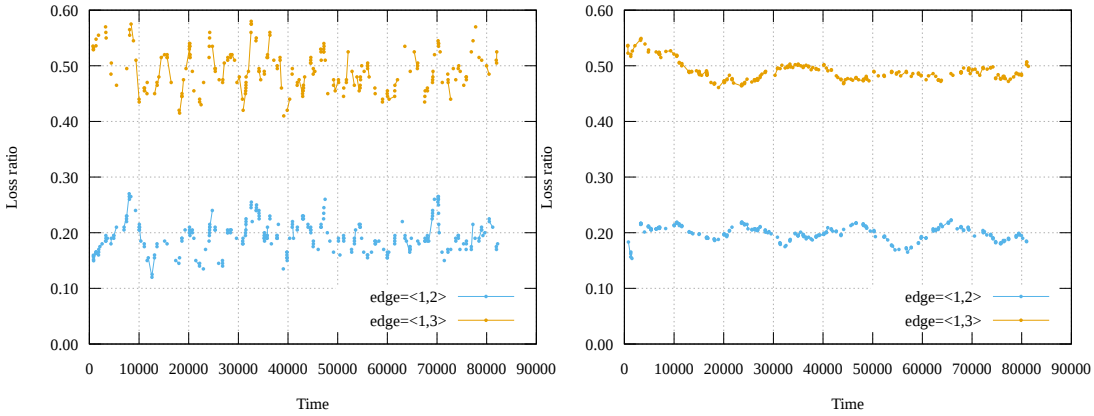
For the shorter simulation duration, we observe oscillations of λ. In fact, they can be caused not by ANChOR but by the channel access mechanism. In order to evaluating the ANChOR stability independent of the channel access mechanism, we analyze the variation of the ANChOR metrics. It is clear that as long as the channel remains stationary, the erasure probabilities converge to their mean values. Nevertheless, ANChOR uses only n of the last reception statuses (with a view to adapt to the channel changes), which causes the oscillations of the measured erasure probabilities with the period dependent



(a) Topology in fig. 3.14 (b) Topology in fig. 3.19

Figure 3.29: Stability of the output data rate (using eq. 3.34)

on n.



(a) $n = 200$ (b) $n = 1000$

Figure 3.30: Stability of the measured erasure probabilities (using the topology in fig. 3.14)

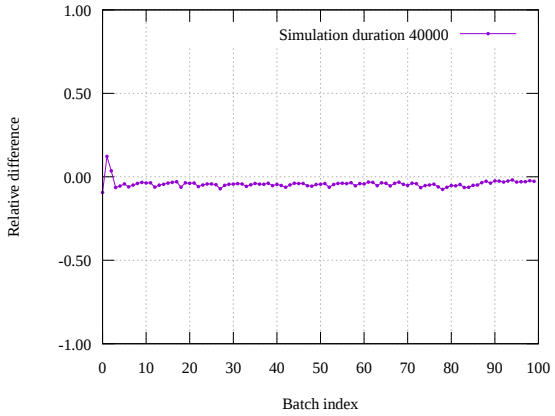
All previous and following simulation results we obtain with $n = 1000$. For this size of the averaging window the fluctuations of the measured values a relatively small (see figs. 3.30a, 3.30b). Anyway, due to the channel stationarity, the measured erasure probabilities are also stationary (observe the oscillation around the certain average values in figs. 3.30a, 3.30b).

The priorities of the vertices depend not only on the measured values of the erasure probabilities but also on the communication between each other. Therefore, the stability of priorities has to be investigated as well. The priority

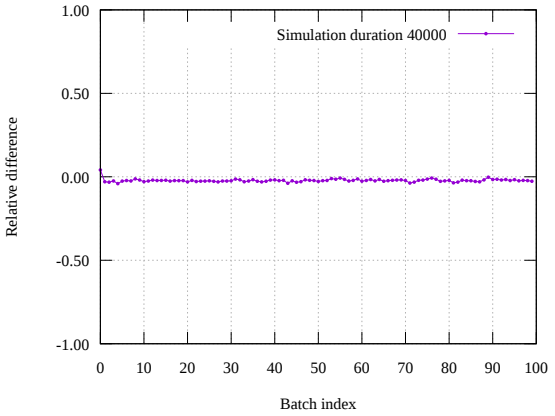
of the source is obtained as a function of all other priorities. Therefore, it is sufficient to analyze the source priority only. For this purpose, we adopt an analog metric:

$$\lambda' = (p(v_s)' - R)/R, \tag{3.35}$$

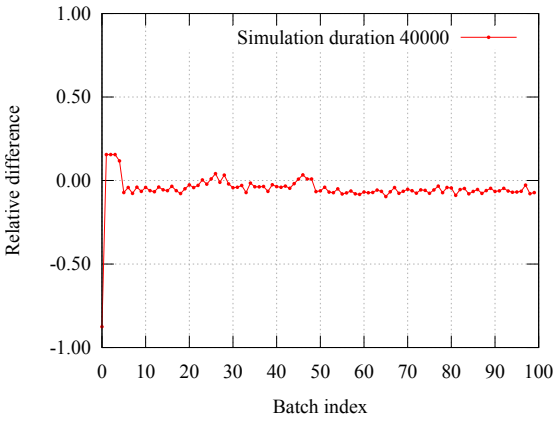
where $p(v_s)'$ is the actual priority estimated by ANChOR and R as before.



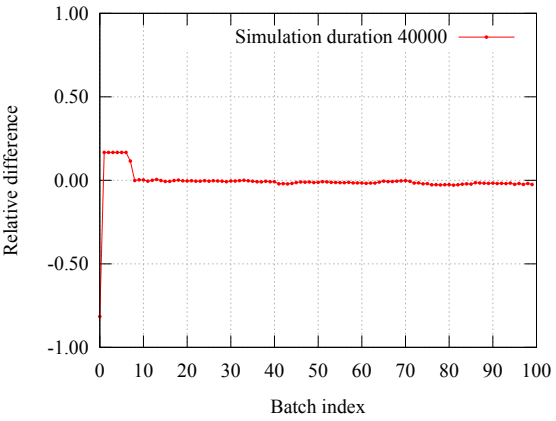
(a) Topology in fig. 3.14



(b) Topology in fig. 3.19



(c) Topology in fig. 3.5 ($\varepsilon = 0.7$, $\varepsilon_1 = 0.2$)



(d) Topology in fig. 3.2 ($\varepsilon = 0.2$)

Figure 3.31: Stability of the source priority (using eq. 3.35)

In fig. 3.31a, the relative difference λ' is presented. For the same simulation duration (40000), it has much less variation than λ in fig. 3.29a, which means that the big variation of the data rate at the destination is not caused by ANChOR. The data rate may also vary due to the randomness of the channel access. Remember that SNS implements slotted ALOHA scheduler with

collision resolution.

The duration of the warm-up phase can be evaluated as the length of the period at the simulation start with significant deviation of λ' from zero. The value of λ' for the network in fig. 3.2 (fig. 3.31d) has the longest warm-up period because it has the greatest number of hops between v_s and v_d.

The value of λ' for the network in fig. 3.5 (fig. 3.31c) has the highest variance. It is reasoned by the big size of the coalition of the source vertex.

The filtering coefficients and coding rates depend on the priorities and the erasure probabilities. As far as a composition of stationary processes gives a stationary process, the stability of the filtering coefficients and the coding rates does not need a proof.

3.11 Summary

Today there are several ORP solutions, like ExOR, MORE, SOAR, CCACK, CodeCast, etc. In difference to these solutions, we provided the theoretically optimal routing rules (BRR) that allow the throughput maximization. We also analyzed the challenges connected with the optimal solution. It requires the central coordination function and an enormous amount of feedbacks. Then, we proposed a near-optimal solution, BRR-E, that is free of these problems. In BRR-E, a vertex v does not need to have a global knowledge of the protocol information. It is sufficient to learn about:

- priorities of neighbor vertices;
- receiving maps on the edges between v and the vertices in its cooperation group $L(v) \subseteq V^+(O(v))$;
- the filtering coefficients for all vertices in $V^-(I(v))$.

Also, the amount of the feedbacks with BRR-E is substantially reduced since the vertices are capable to predict the feedback information. The prediction has become possible thanks to the usage NC. With the Monte-Carlo simulation of BRR-E in SNS, we demonstrate that the negligibly small prediction error can be achieved by selection of the generation size.

Thus, BRR-E allows creating of the decentralized routing protocol ANChOR. In big mesh networks, it dramatically reduces the management communication. ANChOR is a complete routing protocol that defines the network

operation during both the regular and the initialization phase. It guarantees the full data recovery and pertains no idealized assumptions. Thus, it is suitable for the usage in real networks. It is proven to be stable in several communication scenarios, and it is able to adapt to the current traffic demand and channel conditions. The latter is possible through the tuning of the averaging window size for the loss ratio calculation. The faster adaptation is possible at the cost of the protocol accuracy. In stationary scenarios, ANChOR approaches the optimal performance.

For retransmission of the lost data fragments, we implemented the ARQ mechanism. In difference to ARQs used with CSPRs, the ARQ with ORP and NC requires a complex metric to describe the feedback information. After analyzing several common methods that are described in the literature, we concluded that all of them have disadvantages. It motivated us to design the min-max approach that uses all available coder information. Despite of its simplicity, it outperformes other solutions.

All required information for the maintenance of the ANChOR routing rules can be attached to the ANChOR header. It has a variable size. And, it can be optimized for transmission of data in packet trains (several coded symbols in one PHY frame). The influence of the header size on the overall system performance is analyzed for an example of the BB-PLC protocol in the next chapter.

Chapter 4

ANChOR in the Gigabit Home Network (G.hn) Protocol

We implement ANChOR inside of the G.hn protocol with BB-PLC physical layer specification [6, 4] using the ns-3 simulator [66]. We run a full-stack simulation using the protocol stack as shown in fig. 4.1. Our source code is available at [67]. The application layer generates greedy traffic that allows evaluating the maximum achievable data rate with the given technology under test. The greedy traffic generator is realized with the help of the feedback between the DLL and the application layer. Whenever the DLL has a vacant place in its queue, the application layer generates a packet. In this way, the size of the input queue of DLL is always maintained to be full. At the same time, a DLL queue overflow is avoided.

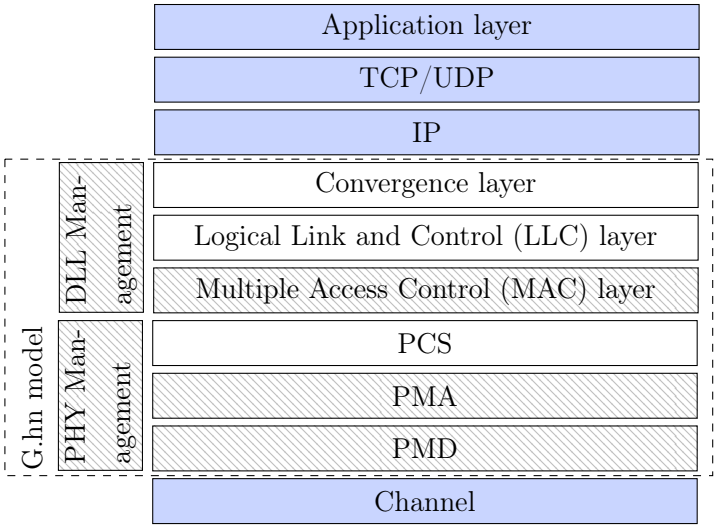


Figure 4.1: Protocol stack. Shaded blocks do not follow G.hn completely [1]

The application layer connects directly to the transport layer. For this purpose, we use the ns-3 module of User Datagram Protocol (UDP) protocol. The network layer is represented by IPv4, which is a part of the official ns-3 repository.

The network layer connects to the DLL of the G.hn protocol [6]. This layer consists of the convergence layer, which is basically responsible for traffic classification and translation of IP to MAC addresses. The underlying LLC layer manages the DLL connections. This layer is the main focus in our study. It creates an ARQ manager separately for each connection. Alongside with the legacy ARQ, we implement the ARQ of ANChOR. The ANChOR protocol itself is implemented as a standalone module. Our ns-3 implementation of G.hn uses the same source code as SNS [65]. We also implement the CSPR protocol that selects the best path using Dijkstra's algorithm. Basically, the ns-3 simulator can work in two modes: the G.hn protocol with CSPR and the G.hn protocol with ANChOR. In the latter case, the ARQ mechanism and the routing protocol are switched to use ANChOR.

The DLL connects to the G.hn PHY [4]. Following the concept of the Open Systems Interconnection model (OSI) model, the main functions of the PHY and the DLL do not influence each other. This eases the protocol design and implementation. In our work, the PHY and the DLL must cooperate since both PHY and LLC encode the data. Their cooperation lies in the mutual selection of the coding redundancy and the sending rate aiming to maximize the data rate on the application layer.

The PHY layer sends data through the channel that models the physical medium of power lines. In contrary to the SNS, the ns-3 G.hn model provides simulation results with realistic packet loss patterns and correlation. Remember that the SNS models the losses of coded symbols as a Bernoulli process. The G.hn model evaluates the losses as the random process influenced by the fading of the PLC signal and the power of different noise components. Thus, the SNS provides the generic study of ANChOR functions, while the ns-3 G.hn model shows the performance that is very close to a real PLC network.

4.1 Compatibility with the PLC protocol stack

The network vertices can support ANChOR if capable to implement the following minimum set of functions:

- broadcast the PHY frames to any set of vertices;
- receive, decode and forward to the NC layer (ISO/OSI layer 2 or 3) the packets sent from any vertex;
- replace or extend the original ARQ mechanisms with the ANChOR HARQ;
- disable relaying required by the traditional RP; other functions of this protocol can be preserved.

The first two functions are necessary to use alternative paths, which is the basic requirement of ANChOR. Though sounding simple, they are not trivial to be incorporated within the existing BB-PLC solutions. The main problem lies in the ability of PLC signal demodulation. The modem can demodulate the PLC signal only if it knows the corresponding tone map (modulation of each subcarrier of the Orthogonal Frequency Division Multiplexing (OFDM) symbol). The tone maps are calculated by the sender modem using the measured SNR of the receiver. During the connection setup, the tone maps are communicated between each pair of modems that build the route for the connection. In a CSPR, each sender dedicates all its traffic to one receiver only. Therefore, it is not foreseen that the neighbors should save the tone maps that are communicated during the connection setup if they do not belong to the route.

In fact, this problem can be easily solved. During the connection setup procedure, the modems use the pre-defined tone map that is known to all network members. Thus, everybody can demodulate this signal and save the overheard tone maps. So, each modem collects a list of tone maps that it associates with different senders. Upon PLC signal reception, the modem can demodulate the PHY header that uses the pre-defined BAT as well (G.hn [4]) and read the sender ID. Then, it demodulates the received signal if the entry in the list of tone maps for the given sender ID exists. The function of decoding

the overheard signal is also called the promiscuous operation mode.

Besides this mode, there exist also other solutions, e.g. multicast. A range of MAC addresses is reserved in the PLC modems for multicast communication. Unfortunately, the multicast mode in most PLC modems is realized as a sequential unicast. We derive this from the laboratory tests with Elcon UPA (Marvell chipset), Elcon G.hn (Marvell chipset), PPC IEEE1901 (Qualcomm chipset) and Devolo HomePlug AV (Qualcomm chipset) modems and on the discussions with the manufacturers. Such implementation is reasoned with the high diversity of the PLC channel. As far as multicast aims to deliver the complete message to each member in the multicast group, the PLC signal at each receiver should be strong enough to decode it. Due to the channel diversity, the optimal tone map for each point-to-point connection may differ a lot. The easiest solution would be to use the most robust modulation and coding schemes. The solution with the sequential unicast can allow higher data rate if the size of the multicast group is not big.

ANChOR does not aim to deliver the complete message to each member of the multicast group. Therefore, the higher data rate can be achieved if selecting the tone map that works as a compromise for all point-to-point connections and maximizes the packet success rate of all modems in the multicast group. This solution differs from the promiscuous mode in the way of the tone map calculation. Note that in the promiscuous mode we use overhearing but the tone map is calculated for a point-to-point connection. Thus, the solution with multicast can allow improving the data rate.

Currently, both promiscuous mode and multicast are implemented by none of the above-mentioned manufacturers in a way that can be used by ANChOR. The only possible solution with existing hardware lies in the usage of the broadcast mode. In this mode, all modems use the same pre-defined tone map. As far as this tone map uses robust modulation and coding schemes, it is difficult to find the scenario where ANChOR can improve the communication. In most cases, the communication is error-free, i.e. no packets are lost. With zero packet loss ratio, the size of the cooperation groups in ANChOR is exactly one. This means that no cooperation is possible.

Therefore, we see a necessity in the extension of the available chipset firmware by modification of either multicast or promiscuous mode.

ANChOR HARQ is designed to interact with the surrounding protocol layers similarly to the original G.hn ARQ. It allows a smooth integration in the available protocol stack. Moreover, it can co-exist with the original ARQ. Fortunately, the implementation of multicast or promiscuous mode will not imply any modifications to the original ARQ since the ANChOR HARQ takes care of all retransmissions.

The routing with ANChOR, similar to the original G.hn routing protocol, is a part of the DLL management block. Though its functions are used on LLC. The LLC layer manages the logical connections. For each source and destination node, it creates a separate instance of managers that work independently of other connections. These managers include the routing rules and ARQ. ANChOR replaces the original G.hn managers. In general, the overall idea about the LLC layer is maintained.

4.2 Channel and noise model

We model the PLC channel using the ABCD-matrix concatenation approach that is implemented as a ns-3 module provided by [68]. It allows defining the network topology specifying the positions of the PLC modems and connecting them with a cable of a certain type. It also contains the framework for attaching of different noise and impedance types of electrical loads. Based on the selected cable structure, positions of PLC modems and loads, the network is exploded in a cascade of sub-networks, for which ABCD-matrices can be easily calculated. The CTF for each pair of PLC modems is computed by serial multiplication of the ABCD-matrices of such sub-networks that build up the network graph between the selected pair of PLC modems. In this way, the singal mitigation and the reflections in PLC are modeled. The impedance and noise power produced by the loads can be specified as time-variant and frequency-selective, which is typical for real PLC channels. In the frequency domain, the CTF is modeled as a collection of sub-channel transfer functions for each subcarrier of the OFDM symbol. In the time domain, the CTF is recalculated with a certain time period that matches the time-variance of the real channel. The duration of this period should be motivated with relatively small changes in the CTF. In accordance with [49], over the period of about 400 microseconds, the PLC channel can be regarded as stationary.

We analyze the performance of the G.hn protocol with and without ANChOR on the example of two types of channel scenarios. First, we setup the in-home scenario. In-home PLC plays an important role on many markets proposing an alternative solution to Wifi. Second, we simulate the scenario with a long electrical line that is typical for electricity metering systems (medium and low voltage access power network). Presently, providing the electricity metering system with communication capability is supported by most European countries. It was previously investigated in [8] and [9]. In the Netherlands, France, Italy, it was planned to use PLC as the communication solution in the majority of cases [10, p.72]. The recent PLC pilot projects by EnBW NETZE, ENSO NETZ (Germany), Salzburg AG, and Wiener NETZE (Austria) show the interest in PLC technology as well. It motivates us to concentrate on the corresponding channel scenarios.

4.2.1 In-home scenario

We use the approach from [69] to generate the in-home topology of the electrical network. This is implemented as a ns-3 module and made available at [70]. We also create a number of electrical devices using the ns-3 module described in [71] that is made available online at [72]. This model allows specifying the impulsive synchronous noise component and the impedance of the electrical devices using the random selection of parameters basing on diverse measurement results (see references in [71]). We also create the colored noise using the model from [5].

4.2.2 Access network scenario

The simulation setup for this scenario differs from the in-home scenario in the topology of power cables and the type of noise. Here, we install no electrical devices. Thus, the only noise component is the colored noise [5]. In this scenario, we install the PLC modems equidistantly with a step size of about 50 meters, which corresponds to the average distance between households in Germany.

4.3 PHY layer implementation

In appendix H we present the functional blocks of the implemented PHY. We simplified the original G.hn PHY, while meeting certain assumptions. First,

we neglect the inter-carrier interference and assume the perfect receiver clock synchronization that allows removing the scrambler. Also, we assume to have an ideal Maximum Distance Separation code (MDS) coder that performs the hard decoding. Note that originally, G.hn specifies a QC-LDPC FEC and a repetition coder. With the assumption of the hard decoding, no "confidence" bits are used, i.e. the output alphabet of the Analog to Digital Converter (ADC) is binary. MDS allows successful decoding (with probability equal 1) when the number of error bits is less or equal to the number of the redundancy bits. Additionally, the positions of the error bits do not influence the decoding probability, which is a fair assumption for an AWGN channel.

We also assumed the perfect implementation of the analog part of PLC modems. So, the decoded sequence does not depend on the finite number of ADC and DAC quantization steps. The inter-symbol interference can be completely mitigated by adding the cyclic prefix. And, the subcarriers are sufficiently separated to avoid the inter-carrier interference. With these assumptions, we omitted the implementation of the blocks between the delta-reference point and MDI (see appendix H).

In our implementation, we can use a wide range of sending data rates that are present in the G.hn. This standard uses 1024 OFDM subcarriers with modulations from BPSK to 4096QAM, FEC with coding rates from 1/2 to 20/21 and the repetition encoder with up to 7 repetitions. Using these setup possibilities, the channel estimation protocol can select the sending data rate with a high granularity. This is typical also for other BB-PLC protocols like IEEE1901 and UPA. Due to the versatility of the PLC channel, the sending data rate of different vertices may differ a lot.

Bit errors that can be still present after decoding on PHY cause the packet erasures on the layer above. The amount of bit errors is highly dependent on the BAT and the transmission PSD of the PLC signal. G.hn defines the BAA to be a part of PHY. We describe it here in detail.

4.3.1 Bit Allocation Algorithm (BAA)

G.hn specifies the protocol for the channel assessment that evaluates the Signal to Noise Ratio (SNR) and defines a way for communicating the BAT, but it does not recommend any BAA. We use the BAA from [2]. It optimizes the BAT for bit rate taking the tolerable BER as a parameter and does not

perform the transmission power balancing. In this section, we motivate our selection and analyze the BAA performance.

The G.hn standard specifies the transmission PSD envelope. So, the transmission signal power in each OFDM channel can be equal or less than the specified power by the standard. Since ANChOR can perform better when the cooperation groups are bigger, it is favorable to use the maximum possible transmission power, which increases the communication range. Therefore, the considered BAA without power level balancing suits to our goals.

The size of the cooperation groups is also influenced by the selection of the tolerable BER parameter. With a CSPR, the BER value should be small enough to eliminate the retransmissions. Each sender has one dedicated receiver that normally takes care of the consistency of the received message. Each receiver performs retransmission requests to ensure 100% reception of the message. With an ORP, the tolerable PER for each separate Pt2Pt link can be much higher. Each sender selects the group of receiving vertices and takes care only of that the group receives the whole message, while each group member receives just a part of it. In fact, ANChOR performs even better when the losses are high. Say, the vertex v has the cooperation group $L(v) \subseteq V^+(O(v))$. Using the refinement procedure of the cooperation group (algorithm 8) we filter such vertices that have not significant impact on $p(v)$. This procedure removes less vertices (i.e. $|L(v)|$ increases) if the loss ratios on edges $e \in O(v)$ are greater.

For explanation, we give the following example. Let the loss ratio on certain edge $e = (v, u)$, $u \in L(v)$, be zero ($\varepsilon_e = 0$). In accordance to the priority calculation procedure (eq. 3.11), all vertices $w \in L(v)$ with lower priority ($p(w) < p(u)$) do not increase $p(v)$. It is analog to say that all such vertices w bring no gain when cooperating with v. Therefore, the refinement procedure removes them from $L(v)$. But if v increases the sending data rate $d(v)$ yielding $\varepsilon_e > 0$ then $p(v)$ increases. If the priority increase exceeds a certain threshold b (eq. 3.11) then the sink vertex of edge e can be added to $L(v)$, i.e. ANChOR has more opportunities to cooperate, which can increase the achievable data rate between v_s and v_d.

The selection of the higher tolerable BER also allows increasing of the sending data rate $d(v)$. But the change of the effective reception data rate

$R(L(v))$ of the cooperation group $L(v)$ is not necessarily positive because of the increasing erasure probability. If $R(L(v))$ does not increase with the selection of the higher tolerable BER then the best strategy is to use the same small tolerable BER that is used by CSPR. In this case, each sender v selects the sending rate that eliminates the losses on the best path between v and the destination. On example of the triangular network (fig. 1.1a), we show that even in this case the gain of ORP against CSPR decreases but is not zero. When the direct path between the source and the destination in fig. 1.1a is better, the gain of ORP against CSPR is zero (using eq. B.3 with $\varepsilon_3 = 0$). But when the path over the relay is better, the gain is calculated as follows (using eq. B.3):

$$\alpha = \frac{(1 - \varepsilon_3) \cdot d(v_0)}{d(v_0) \cdot \varepsilon_3 + d(v_1)}. \tag{4.1}$$

Surely, α increases if the vertices select higher sending data rates that results in $\varepsilon_1, \varepsilon_2, \varepsilon_3 > 0$.

Besides the transmission PSD and the tolerable BER, the BAA requires also the SNR for a certain Pt2Pt link. In ANChOR, each vertex can have several sink vertices because $|L(v)| \geq 1$. Therefore, optimally, ANChOR should use the sending data rate that is evaluated for the Point-To-MultiPoint (Pt2MPt) link that maximizes the achievable data rate in the broadcast channel ([53, p.570]). In this case, v requires the SNR for each Pt2Pt link in the Pt2MPt link. Nevertheless, in BB-PLC, the multicast transmission is normally realized with sequential unicast [17] or broadcast [6]. A probable reason lies in the protocol overhead to keep all SNRs up-to-date and the substantial frequency selectiveness of PLC channels [73]. Therefore, we propose to select only one vertex $u \in L(v)$ and apply the BAA for the edge $e = (v, u)$. The resulting BAT and PHY coding rate should be communicated to all vertices in $L(v)$. Then, any vertex $w \in L(v)$ can demodulate the PLC signal sent by v. This approach is well harmonized with G.hn. In this standard, when v wants to communicate with u, it sends a request to u to calculate the corresponding BAT basing on the SNR, the transmission PSD, and the tolerable BER. This communication is realized during the establishment of the LLC connection, i.e. before any data communication takes place. For this purpose, a set of control messages between v and u is exchanged. The control messages are mapped into PHY frames and eventually in OFDM symbols using a pre-defined BAT.

Therefore, when u responds v with certain BAT for data communication, each vertex $w \in V^+(O(u))$ can demodulate this signal and find out the BAT for data communication that will be used by v (promiscuous mode). Thus, any vertex $z \in (V^+(O(v)) \cap V^+(O(u)))$ is able to demodulate the data communicated between v and u. We use the same principle in the simulation model.

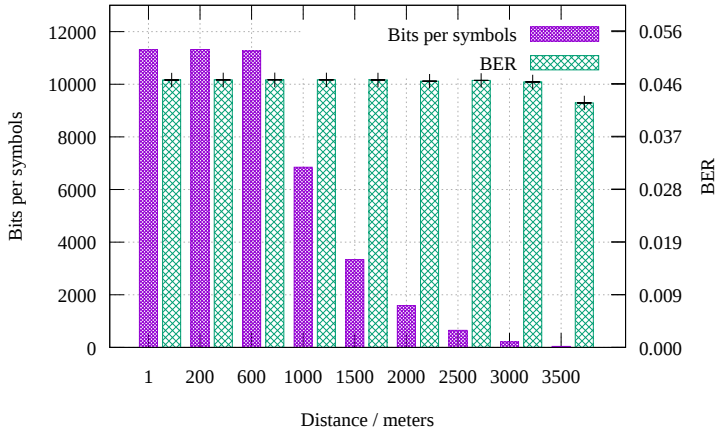
The vertex v has a freedom to select any vertex from $L(v)$ for the calculation of the BAT. We select the sink vertex with a mind to maximize the number of cooperation opportunities. Thus, our idea concerns the updating of the cooperation groups. Here, we extend the rules for the construction of cooperation groups provided in section 3.7. We propose to calculate the sending data rate of v each time when v learns about the presence of new neighbors. Let the vertex v receive the information about the presence of a new neighbour u with priority $p(u) < p(w)\ \forall w \in L(v)$. Let also in accordance to BRR-E u can join $L(v)$. Then, we apply the BAA for the edge $e = (v, u)$ using a tolerable BER that corresponds to $\varepsilon_e = 0.2$ (see mapping in appendix I). The resulting BAT and PHY coding rate are recalculated into the effective sending data rate $d(v)$. This value is passed to ANChOR. Receiving the new value of the sending rate, ANChOR resets its statistics to initial values (see section 3.6). Consequently, $\forall u \in L(v) : \varepsilon_{(v,u)} = 0$. After performing the refinement procedure (algorithm 8), $L(v)$ will consist of one vertex only, the vertex with the highest priority. Then, v enters the state similar to the initialization phase. It needs time to accumulate substantial statistics on the links quality before it can add all remaining vertices to its cooperation group again.

Note that in PLC the channel changes when the electrical devices change their operation mode, which may happen even more seldom than several seconds. As a result, for this period, each vertex has a fixed set of neighbors that can be physically reached. Therefore, the recalculation of the sending data rate does not cause the instability in the network. Moreover, the sending data rate is not recalculated when the vertex v receives the information about the presence of a new neighbour u with priority $p(u) \geq p(w)$ for some $w \in L(v)$.

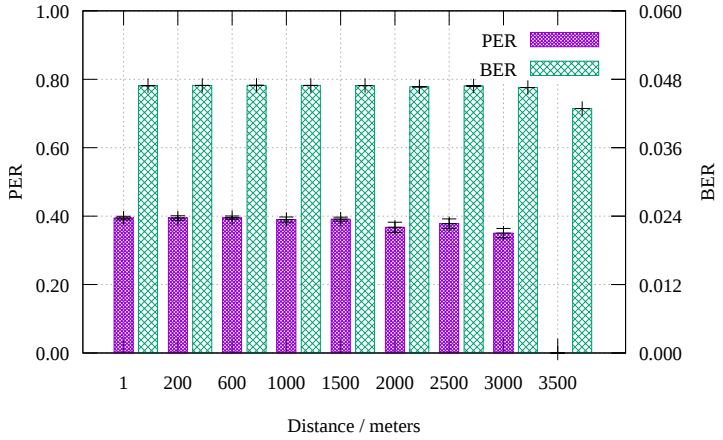
With regard to the mentioned above, the BAA from [2] completely satisfies our needs. The only extension that we supply is the PER to BER mapping that is described in appendix I.

As far as ANChOR targets to set the target PER for a given edge, we an-

alyze the accuracy of the PER setting. Consider two PLC modems connected with the cable NAYY150SE [74] (using channel model from [68] and colored noise model from [75] (worst case)). The modems use G.hn PHY with the simplifications as shown in appendix H and the BAA from [2]. Increasing the



(a) Dependence of bits per symbols and BER from distance



(b) Dependence of bits per symbols and BER from distance

Figure 4.2: Analysis of BAA [2] on Pt2Pt scenario. Target PER=0.4. Corresponding BER=0.04695 for $c = 20/21$ (see appendix I)

distance between the modems, the Shannon channel capacity decreases. Using the adaptive bit loading, the BAA keeps BER almost unchanged for the range of distances between 1 and 2000 meters (fig. 4.2a). Nevertheless, BER values are slightly lower than BER=0.04695 that is corresponding to PER=0.4 and coding rate $c = 20/21$ (see appendix I). The significant decrease of BER is typical for big distances when the BAA uses a small subset of carriers due to

a low SNR in a wide frequency range. In fig. 4.2b, we show the resulting PER. It almost equals the target PER for the range of distances between 1 and 2000 meters.

As mentioned above, we calculate the BAT for such receiver vertex from the cooperation group, which connects to the sender vertex with the worst channel. The result in fig. 4.2b shows, that such receiver vertex can be located at a distance of 2000 meters from the sender. All vertices in-between can overhear the PLC signal. Although our simulation model is realistic, it uses a number of simplifications. If we replace the ideal MDS coder with the QC-LDPC coder then this distance can decrease.

In the following, we present simulation results to demonstrate the BAA operation. For this purpose, we use the simulator from [67]. It implements the channel model from [68], NAYY150SE cable model, G.hn PHY and coloured noise [75] (best case conditions). We connect two vertices v and u with the cable and apply the BAA. In fig. 4.3, the BAT for different distances between the pair of vertices is shown. It was calculated for $\varepsilon_e = 0.1$ ($e = (v, u)$), which corresponds to BER=0.04381 (see fig. I.1). As far as the electrical cable has no junctions/branches and the PSD of the coloured noise has no peaks, the number of bits per subcarrier gradually decrease with the frequency increase.

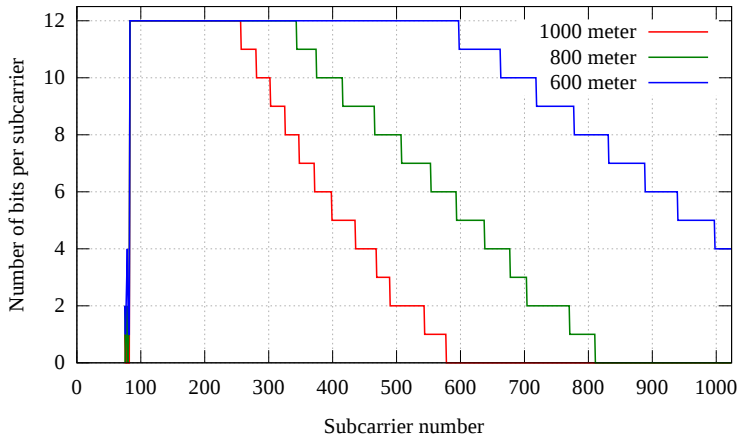


Figure 4.3: Adaptive bit loading algorithm from [2]. BER=0.04381 (corresponds to PER=0.1 with 20/21 ideal coder). Example of bit loading for different distances between Tx and Rx vertices (connected with the cable of different length)

4.4 MAC layer

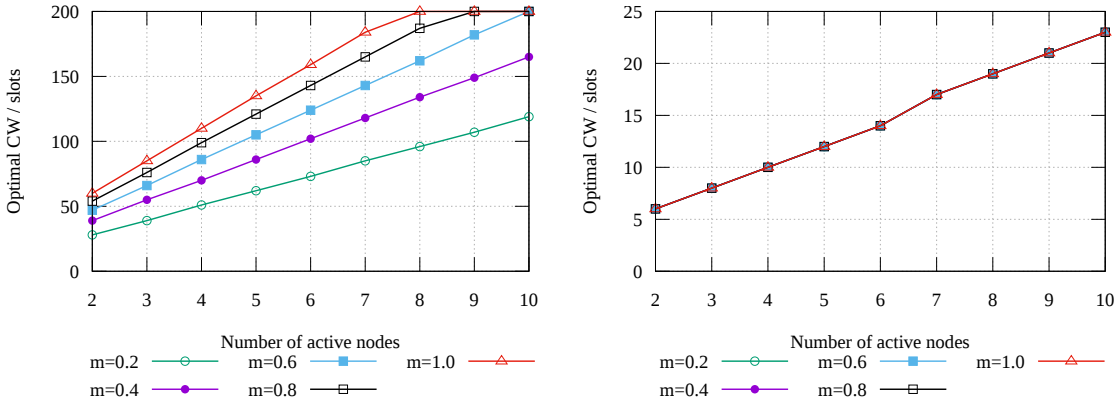
In ORPs, the coordination between the modems is regarded as the main problem for a wide usage. ANChOR operates in a decentralized mode that allows reducing the number of control messages. The decentralization impacts the selection of the MAC scheduler.

G.hn implements TDMA and Carrier Sense Multiple Access (CSMA). TDMA entitles the central (master) node to perform the scheduling. This node reserves the time slots for each network member in accordance with the requested service (type and amount of traffic). Periodically, it broadcasts the Multiple Access Plan (MAP) that announces the time points and durations for transmission of each network member. Thus, the transmission planning is realized before the data transmission actually starts. This concept contradicts the idea of ORPs, where the vertices decide on forwarding the data upon the fact of the data reception. Therefore, TDMA scheduling is not suitable for ANChOR.

We implement CSMA. Differently to TDMA, the channel access with CSMA is decentralized. Each vertex runs the contention procedure that gives it a chance for the channel access autonomously from other modems. In this procedure, the modem generates a random number that initializes the *backoff timer*. When this timer expires, the modem senses the channel. It measures the total power of signal and noise currently present in the channel. If the power exceeds the certain threshold, the channel is regarded as "busy". In this case, the modems generate the new value to initialize the backoff timer and repeat the procedure. In this way, the simultaneous start of transmission by several modems can be avoided if the backoff timer has sufficient granularity. This type of CSMA is also called the CSMA with Collision Avoidance (CA) (CSMA/CA). The collision happens if the backoff timer of two or more modems expires at the same time. Then, all such modems sense the channel as "free" and start transmission. The received PLC signal becomes a superposition of the channel noise and all other sent PLC signals. In most cases (depending on the power of each signal component), it is impossible to decode any of the sent signals. Thus, a collision produces a waste of channel resources. The collision probability can be reduced by increasing the number of possible

initial values of backoff timer. The range of possible values is also called the CW size. The selection of the large CW size yields the wastage of the channel resources as well because the modems have longer pauses between attempts to start transmission. For a given number of active (willing to start transmission) modems and the protocol overhead for sending the message of the given size there exist the optimal CW size [76].

The optimal CW size can be reduced when using the CSMA with Collision Detection (CSMA/Collision Detection (CD)). As a result, the achievable throughput can be increased [1]. Moreover, the optimal CW size with CSMA/CD is not that much sensible to the number of active modems as the optimal CW size with CSMA/CA. In figs. 4.4a and 4.4b, one can see the opti-



(a) CSMA/CA (b) CSMA/CD

Figure 4.4: CW size that maximize the MAC efficiency [1]

mal CW size as the function of the number of active modems and the duration of the PHY frame (m). Here, $m = 1.0$ means the maximum duration of 40 milliseconds (G.hn). The range of variation of the optimal CW size is much less with CSMA/CD. It means that adding more active modems does not degrade the throughput significantly. With ORPs, we have potentially more active modems than with CSPRs. Using CSMA/CD, the overall performance with ORP will not be impaired due to the increased time for the channel access.

G.hn specifies CSMA/CA with the backoff algorithm that operates as follows. The CW size can be selected as one of the values $\{8, 16, 32, 64\}$. The actual CW size depends on the number of previous collisions and "busy" channel assessment. The CW size is doubled whenever a collision is inferred. Also,

it doubles after a certain number of "busy" channel assessments in a row that can be one of the values $\{1, 2, 4, 16\}$. This number is doubled each time when either the collision happens or the channel is detected as "busy" for the current number of "busy" channel assessments in a row. In the case of successful transmission, the CW size defaults to the minimum value.

The maximum CW size in G.hn, $CW_{max} = 64$, is not sufficient for a network with a big number of nodes (see fig. 4.4a). Therefore, it cannot be used in access network scenarios (up to several hundreds of modems) without modification.

We do not use CSMA/CA at all. Instead, we use CSMA/CD with a very simple backoff algorithm. The simplicity becomes possible thanks to the weak dependence of the optimal CW size on the number of active modems and the size of PHY frame (see fig. 4.4b). In fact, we set the constant value $CW = 20$ (1 slot equals 35.84 microseconds in G.hn [6]).

4.5 LLC layer

In this section, the basic principle of HARQ in ANChOR is described and compared to the ARQ mechanism of the G.hn protocol.

G.hn creates separate ARQ buffers for each data flow. The source has one ARQ buffer, the relay has one for reception and one for sending and the destination has only one ARQ buffer for the reception.

In G.hn, ARQ on the receiver side does not mix data from several senders, i.e. it is associated only with one sender. In ANChOR, as long as data was originated by the same source node, each receiver puts all received data to the same receiver ARQ buffer. In this way, the multi-path data transmission is enabled.

4.5.1 Reference ARQ

In this work, G.hn ARQ is taken as a reference. It is a selective repeat ARQ with no error correction on link layer. Each packet of the ARQ buffer is appended with a CRC block for error check only (see fig. 4.5). In fig. 4.5, the construction of the MAC layer PDU (MPDU) is demonstrated. The PDUs from the convergence layer are converted to LLC frames and stored in the buffer. When the modem has any packets on LLC layer, it signals the MAC

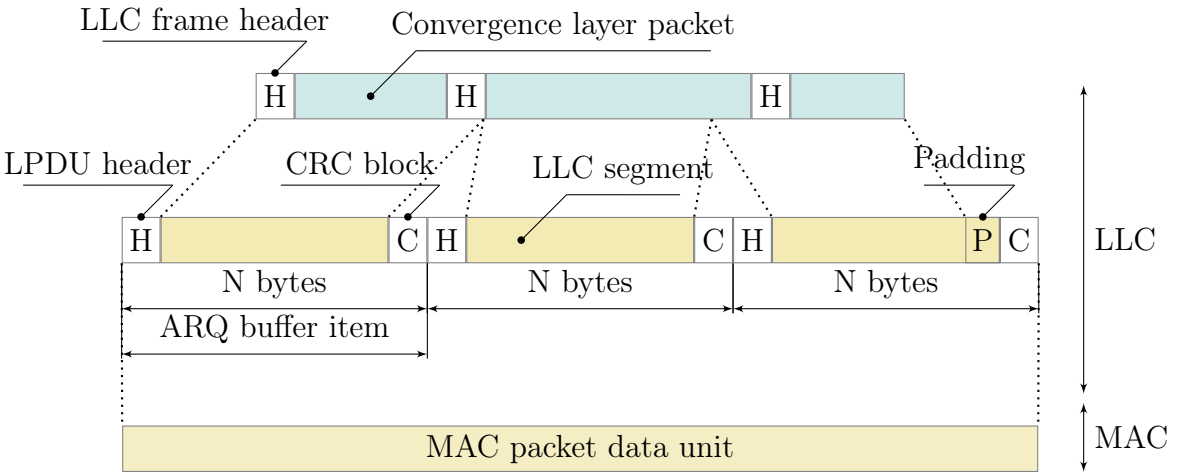


Figure 4.5: MPDU construction in G.hn. $N = 540$ or $N = 120$ bytes

layer to gain the channel access. When the MAC layer gets the channel access, the LLC frames are segmented to PDUs of equal size. The segments receive a header containing their Serial Sequence Number (Serial Sequence Number (SSN)). They are also named as LLC PDUs (LPDUs). The LLC also tells the MAC layer the amount of outstanding data in the buffer. The MAC layer takes either all available data on the LLC or a part of it corresponding to the available channel access duration. Anyway, a MPDU contains several LPDUs. Although all of them are delivered to the PHY layer as a single packet, the PHY layer divides it into subframes with the size that equals the LPDU size. Each subframe is encoded into a separate FEC block. In this way, each LPDU is encoded by FEC on the PHY layer separately. It means that the loss of LPDU is equivalent to the loss of a FEC block.

The LPDUs must be acknowledged. They are stored in the second buffer. It is also called the ARQ buffer. The range of LPDUs that can be sent within one MPDU is called the ARQ window. It is defined with the oldest SSN in the ARQ buffer and the ARQ window size. The oldest SSN is updated each time when the corresponding LPDU with current oldest SSN leaves the ARQ buffer. Such an event is also called the shift of the ARQ window. Thus, the window can be shifted only if the oldest LPDU is acknowledged or the number of its retransmissions expires. The packet losses can block the window shift. In this case, the "newer" already acknowledged LPDUs are blocked in the ARQ buffer by the oldest LPDU, which is also known as the head-of-the-line problem.

We demonstrate the operation of the sender ARQ in fig. 4.6. Each cell

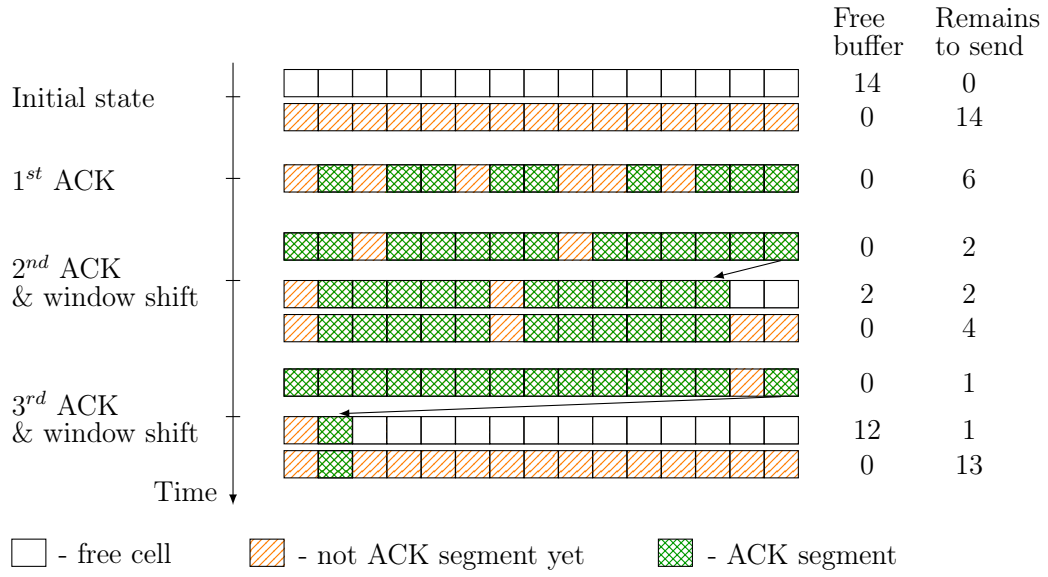


Figure 4.6: Example: G.hn ARQ; sender side; window (buffer) size - 14 segments

denotes the place for LPDU. The cells are marked differently to notify the usage of the cell. If the cell is empty, new LPDUs can be pushed into the buffer. The cell with LPDU can be in two states: already acknowledged and not acknowledged yet. Note that LPDUs marked as acknowledged are not retransmitted anymore (selective ARQ). Still, they possess the places in the buffer till the window shift happens. In fig. 4.6, after the 1^{st} acknowledgment no shift happens because some segments were not acknowledged. It can happen if these segments were not decoded by the receiver due to low SNR. Later, the window is shifted at first by 2 and then by 12 cells to the left.

With this example, it is shown that in G.hn the number of packets remaining to be sent from the ARQ buffer is equal or less than the ARQ window size. The LLC layer passes to the MAC layer only those LPDUs that fall within the ARQ window. As a result, the size of MPDU can be limited with the available data in the ARQ window even though the ARQ buffer and the LLC frame buffer may contain more packets.

This results in two shortcomings. First, it is not possible to shift the ARQ window by the same number of cells after each transmission that impairs the jitter. Second, the decreased MPDU size can reduce the throughput due to the growth of the percentage of the protocol overhead. One can propose removing the acknowledged segments from the buffer immediately. But it will cause a

massive re-ordering of LPDUs. Since the LPDUs have to be segmented, the order should be maintained.

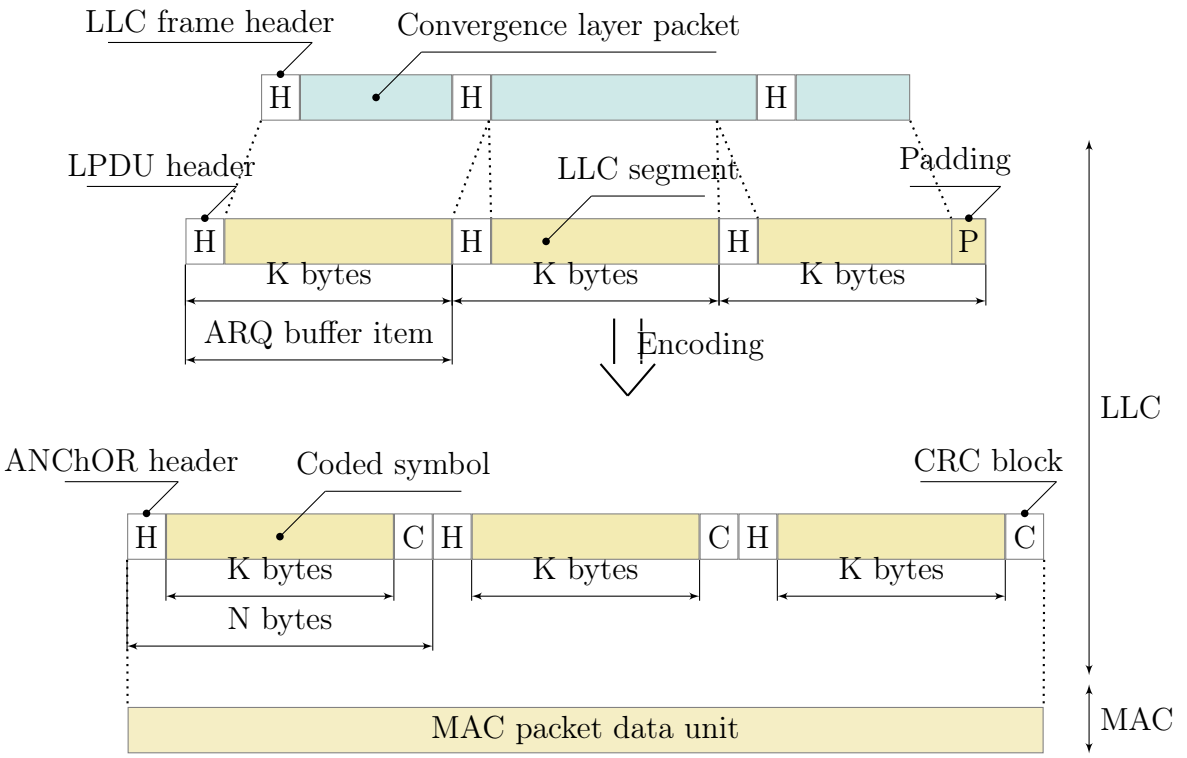


Figure 4.7: MPDU construction in ANChOR on v_s. $N = 540$ or $N = 120$ bytes

4.5.2 HARQ in ANChOR

The above problems can be partially solved with a hybrid ARQ. First, we demonstrate the construction of MPDU when using ANChOR HARQ. It is similar to the MPDU construction in G.hn (compare fig. 4.5 and fig. 4.7). The packets of the convergence layer are prepended with LLC frame header, segmented and converted to LPDUs. The size of LPDUs is smaller than the LPDU size in G.hn on the ANChOR header size and CRC block size. Afterward, LPDUs are encoded with RLNC. The number of encoded symbols is defined by ANChOR. In fig. 4.7, no redundant symbols are created. The ANChOR packet consists of the coded symbol, the ANChOR header, and the CRC block. Its size equals the size of LPDU in G.hn. Note that the scheme shown in fig. 4.7 is applicable only for the source vertex. The relay vertex does not receive data from the convergence layer. It recodes the received coded

symbols.

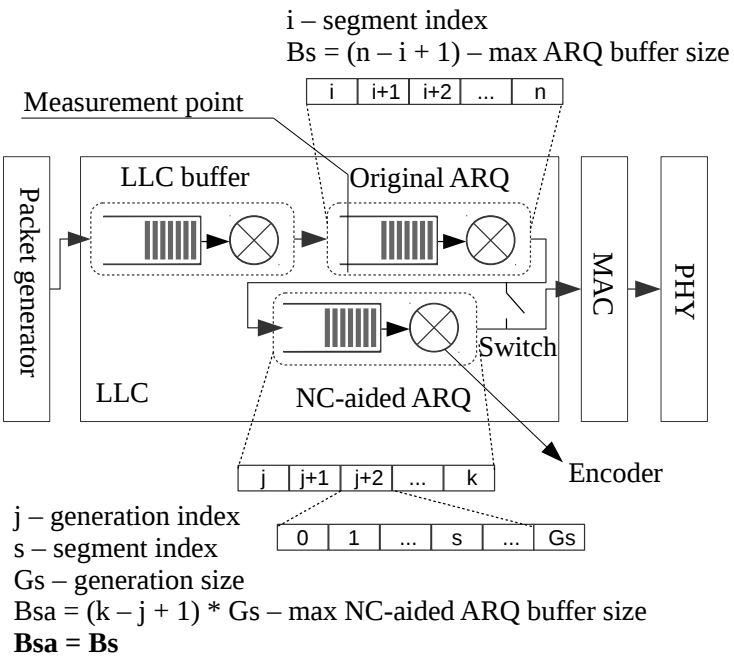


Figure 4.8: Queueing model of the G.hn Logical Link layer [3]

On the PHY layer, the ANChOR packet is encoded with FEC in an analog fashion as LPDUs in the original G.hn. Note that the RLNC in our implementation is not capable of correcting bit errors. Therefore, the coded symbol is discarded if the CRC check fails. ANChOR HARQ never resends the same coded symbols and does not combine several received erroneous symbols in order to improve the SNR. Thus, it belongs to incremental redundancy hybrid ARQ in selective repeat mode.

Fig. 4.7 shows the MPDU construction of the ANChOR packets that belong to the same generation. Actually, the ARQ buffer can contain a number of generations. Different to the original ARQ, here the generations are the subject for acknowledgment. But it also uses selective ARQ. It also has the ARQ window as shown in fig. 3.12. And, it retransmits the coded symbols of the requested generations.

We also advocate the ANChOR HARQ implementation that does not require the replacement of the original ARQ. Instead, the original ARQ and ANChOR HARQ are placed in tandem (see fig. 4.8). In [3] we studied the per-

formance of such an implementation. It was shown that a switch between the original and ANChOR ARQ can be helpful. The original ARQ works better for a small packet loss ratio and small ARQ window sizes.

Eventually, we showed the optimal generation size in relation to the buffer size considering throughput and latency. The conclusion was to make the generation size as big as possible. But in reality, it is limited by the computational power of the hardware. As a result, the HARQ buffer contains a number of generations.

4.5.3 Modeling PDU erasures on LLC

In the simulator, we make no bit processing and no Cyclic Redundancy Code (CRC) check. The erasure event creation and detection are organized as follows. Upon a reception event, the receiving vertex v obtains the signal Power Spectral Density (PSD) and noise PSD, both time and frequency variant. First, v calculates N_r, the total amount of information bits received, using the Shannon capacity, which is calculated separately for each of K subchannels with width Δf and each of A time intervals with duration t. From the PHY frame header, it is possible to read out the coding rate cr, PHY frame size N_s (without redundant bits) and Forward Error Correction (FEC) block size bs (PHY frame consists of several FEC blocks). The value N_r does not count the redundancy bits.

Algorithm 9. Erasure creation/detection

1 $N_r \leftarrow cr \cdot \sum_i^A \sum_j^K C_{ij} \cdot t \cdot \Delta f$

2 **if** $N_r > N_s$ **then** $BER \leftarrow 0$ **else** $BER \leftarrow (N_s - N_r)/N_s$

3 $n_r \leftarrow \mathcal{B}(bs, 1 - BER)$

4 **return** $(n_r \geq bs \cdot cr)$

Then, the average BER value for the complete PHY frame is calculated. We assume the bit errors to be independently distributed in the PHY frame, that is close to reality if PHY uses scrambler and/or interleaver. For example, in G.hn and IEEE 1901 the scrambling is applied on all PHY bits. In addition, in IEEE 1901 it is possible to use the interleaver for previously encoded PHY blocks with the turbo convolutional encoder. Assuming the bit error to be a Bernoulli event with the flipping probability $p = BER$, the number of error bits in the FEC block, n_r, is binomially distributed. Here, $\mathcal{B}(n, p)$ is the

function generating the binomially distributed random value after n trials and with error probability p. If the number of error bits exceeds the number of redundant bits then the erasure of the FEC block happens. Note that the FEC blocks in G.hn are mapped to LPDUs. Thus, the erasure of the FEC block yields the erasure of the LPDU. Originally in G.hn, the erasures are detected on the LLC with the CRC.

4.6 Summary

The available BB-PLC modems (Elcon UPA (Marvell chipset), Elcon G.hn (Marvell chipset), PPC IEEE1901 (Qualcomm chipset) and Devolo HomePlug AV (Qualcomm chipset)) are not ready to support the ANChOR functionality. First, they should extend the connection establishment procedure to enable collecting the tone maps from all senders. In this way, several receivers can demodulate the same PLC signal, which is a crucial requirement for cooperation. Alternatively, they can be modified to allow the multicast data transmission (omitting current implementation as the sequential unicast). Any of these modifications do not require changes to the PLC standards.

From the perspective of HARQ and routing protocol implementation, the PLC standards have to be extended. First, they should foresee a flag that allows the selection between the original and ANChOR setup. Second, the ANChOR functionality should be regulated in the same fashion as the standard setup, i.e. description of HARQ, routing protocol and the formats of messages.

On the MAC layer it makes sense to implement CSMA/CD instead of CSMA/CA [1]. With ANChOR, more nodes can be involved in communication than with the original RP. As a result, the channel access overhead may increase. We showed that CSMA/CD is not that much sensitive to the number of active modems as CSMA/CA. But it has a higher MAC efficiency [1]. The implementation of CSMA/CD requires the extension of the PLC standard with a description of the backoff algorithm. Also, it implies an extension to the PHY layer specifying in-band full-duplex functionality. The technical difficulty of CSMA/CD implementation relies on the difficulty involved in implementing of in-band full-duplex in PLC [1]. The implementation costs are explained in [77].

The PLC modems designed with the modified standard can be interoperable with the modems of the current state of the art if the new modems can default to the current settings. If the PLC modem v works with ANChOR setup and the PLC modem u works in the original setup, then the full interoperability is not possible. The modem v can receive and process all data from u but not the vice versa.

Chapter 5

Study of G.hn with ANChOR

The performance of ANChOR in BB-PLC depends not only on the criteria described in the ANChOR study with the SNS (chapter 3). First, SNS simplifies the calculation of the overhead created by the feedback messages. It is evaluated in terms of the number of feedback messages. In real systems, the feedback overhead consists of the additional header size, the channel access, and the PHY frame transmission timings. The SNS also does not consider the overhead of the PHY layer, which depends on the length of the PHY payload. With the G.hn simulator, we also study the method for the control of the sending data rate. In the study with the SNS, we assumed the sending data rate to be fixed for each vertex. In fact, the PER depends on it and each vertex has a variety of choices. In overall, we provide the full-stack simulation that allows evaluating the complete system performance in a realistic environment.

5.1 ARQ analysis

The ARQ mechanism takes care of a reliable data transmission. Without retransmissions, it is possible to send the data faster because no feedback messages are sent and no waiting timers are involved. Adding the retransmissions, we can repair the missing data but we also reduce the throughput. When losses of the transmitted data must be repaired, we also may be interested in optimizing the communication either for the throughput or for the latency. Aiming to decrease the latency, we may prefer to retransmit the data with smaller timeouts if no ACK message is received. This may decrease the throughput because the amount of unneccessary retransmissions increases. In the ARQ

mechanism of ANChOR there are several variables that can be used to improve either the throughput or the latency. We concentrate on this analysis in the following sections. Here, we study the additional effect produced by the ARQ. It influences the size of the PHY frames. This size has a significant impact on the throughput. The longer the PHY frame is, the more time is used for transmitting of the PHY payload in comparison to the PHY header duration, preamble, and other timings. The PHY payload size equals the MPDU size N_a. The MPDU consists of several LLC packets (packet train). In case of the original G.hn, these are the LPDUs, and, using G.hn with ANChOR, these are ANChOR packets. The transmission of the packets in packet trains has also the negative influence. Due the bursty packet arrivals the jitter may increase. Thus, in addition to the throughput and the latency we also analyze the jitter.

We use the ARQs as described in sections 4.5.1 and 4.5.2.

The average MPDU size N_a is limited by the layers above and below. From below, N_a is limited by the size of the PHY frame in G.hn. The lowest number of the OFDM symbols that carry the data bits is one. The maximum number of the OFDM symbols in one PHY frame corresponds to the PHY frame duration of 40 milliseconds (for the mains frequency of 50 Hz). As far as this statement does not follow from the G.hn standard directly, we provide the explanation. The channel access in G.hn is organized either with TDMA or with CSMA. With TDMA, in the defined time period only one node is allowed to transmit. With CSMA, a group of nodes can contend for the channel access. The duration of Contention-Free Transmission Slots (CFTS) and Contention-Based Transmission Slots (CBTS), operated by TDMA and CSMA correspondingly, is announced in the Multiple Access Plan (MAP) message. It is regularly broadcasted to all network members. This plan can cover at most 40 milliseconds [6]. Thus, a node can obtained the longest channel access period if the MAP describes only one transmission slot and only one node occupies it. Then, the maximum duration of the PHY frame equals 40 milliseconds.

In most cases, this limit is not hard enough comparing with several other reasons. The value N_a is limited from above by the LLC layer. The LLC cannot send more packets to the MAC than the maximum ARQ window size N_m. Depending on the application, different N_m may be optimal. With delay

tolerant, high volume data transfers (like file transfer), a greater N_m increases the throughput. For HTTP requests, a smaller N_m decreases the latency. G.hn defines a fixed $N_m = 376$ for data packets. With a perfect communication channel, the MPDU containing 376 LLC packets of 540 bytes can be transferred in 5.973 milliseconds (979 carriers with 12 bits per carrier, OFDM symbol duration 42.24 microseconds, PHY header duration of 51.2 microseconds and preamble of 51.2 microseconds).

N_m belongs to the setup parameters. There are also other factors that influence N_a in real time. As described in section 4.5.1, the head-of-the-line problem reduces the number of packets that can be sent in a packet train. The original ARQ and ANChOR HARQ have different sensitivities to this problem because ANChOR does not require to keep the packet order inside of each generation. This increases N_a. From another side, there are two factors that decrease N_a when using ANChOR. Consider the example in fig. 3.5 (arbitrary loss ratios). With CSPR, v_s selects one of m relays to forward the received message. The ORP allows all relays to contribute. As a result, due to the application of the filtering rule in BRR-E, each relay sends a shorter message than the original one. Moreover, the ACK window of v_s is limited by the end of the ACK windows of all m relays (see section 3.5.1). Thus, the bigger the cooperation group size N_c is, the smaller N_a is expected and the more the data rate drops. For certain m, adding new members to the cooperation group may not make sense.

We use a simple testbed with a pair of PLC modems connected with a power cable of fewer than 10 meters. The communication channel is good enough to enable the highest possible data rate between the modems. The PLC modems are connected to measurement computers over Ethernet ports. We create traffic trains (bursts) on the application layer using UDP on the transport layer. Within the packet train, the packets are pushed into the UDP socket as fast as the socket can allow (much faster than actual data transfer over the PLC network). No packet losses in the sender buffer occur. The duration between bursts is much greater than the burst duration. The size of packet on the application layer is $P_s = 1400$ bytes. The time for the burst transfer $T(N_a)$ is measured on the application layer as the difference of time between the reception of the first and the last packets. $T(N_a)$ is affected not

Table 5.1: Overhead parameters. Application layer. Approximated $T(N_a)$ based on the measurement results

	G.hn	HomePlug AV 200
O_c / μs	5740.63	6306.43
O_p / μs per LLC packet	117.51	189.438

only by the PHY frame duration but also by the ACK duration, the channel access time and other PHY timings. The measurement has shown that the obtained dependence can be linearly approximated [3] (see fig. 5.1):

$$T(N_a) = O_c + O_p \cdot N_a, \tag{5.1}$$

where the coefficients O_c and O_p describe PHY, DLL, network and transport layer overhead.

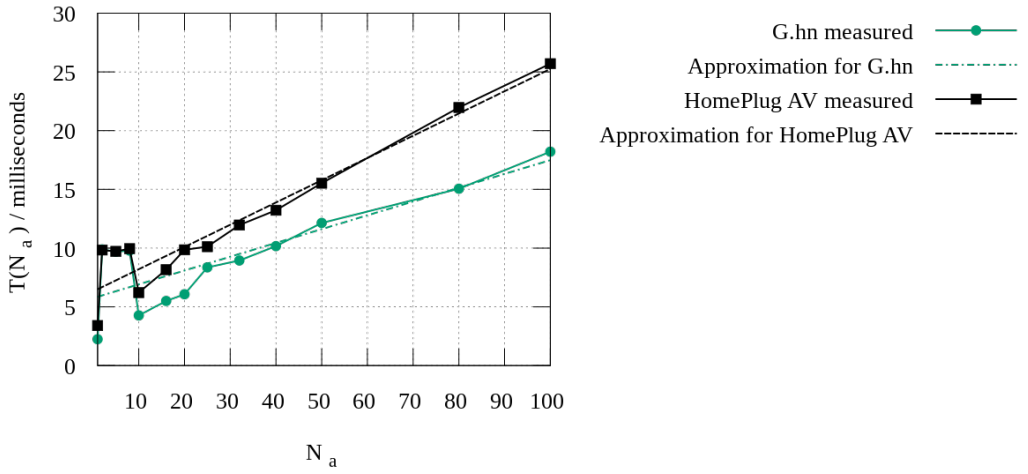


Figure 5.1: Measurement results. $P_s = 1400$ bytes; N_a - average traffic burst size; $T(N_a)$ - time to transfer the traffic burst

The constant overhead O_c consists of MAC timings, PHY preamble, PHY header and DLL service protocol overhead. The proportional overhead O_p depends on the coding redundancy, intersymbol gaps (OFDM), LLC headers and CRC. O_c and O_p are calculated in a linear approximation (see table 5.1).

The linear approximation of $T(N_a)$ for $N_a > 10$ on the application layer allows assuming also a linear form of time to send a packet train on the LLC. Surely, the values O_c and O_p on the LLC in G.hn are different from those

Table 5.2: G.hn PHY variables influencing O_c and O_p

Parameter	Most robust mode	Least robust mode
PHY header / OFDM symbols	1	1
LDPC coding rate	1/2	20/21
Repetition	7	0
Bits per subcarrier	1	12

obtained in the measurement on the application layer. We evalute O_c and O_p on the LLC in G.hn analytically.

In table 5.2, we list the protocol variables that can influence O_c and O_p. The values differ depending on the PHY layer setup. For the most robust setup, a bigger value O_p is typical due to high coding redundancy and low number of bits per OFDM symbol.

In addition to the PHY layer, the DLL layer overhead in terms of headers on LLC layers, MAC timings and the average time for getting the channel access is considered. The average channel access time is calculated for CSMA with average contention window size for two contending vertices [76]. In table 5.3 the resulting values are shown.

Table 5.3: Overhead parameters. LLC layer in G.hn. Analytical derivation based on [6, 4]

Variable	Most robust mode	Least robust mode
O_c / μs	465	465
O_p / μs per LLC packet	2980	16.29

The obtained parameters are very coarse since they do not include the overhead created by service protocols on the DLL like the channel estimation protocol. We also simplified CSMA removing the INUSE and PR (priority) slots [6]. Therefore, O_c in tables 5.3 and 5.1 differs a lot. The measured value of O_p in table 5.1 is in the range between the values in table 5.3 with the tendency to the least robust mode. This is true due to the good communication channel in the testbed.

The more robust a mode is, the less influence has O_c on $T(N_a)$. It means that N_a has less impact on the data rate. In contrary, with a good channel,

the influence of the constant overhead O_c is noticeable. If the MPDU contains 376 LPDUs of 540 bytes and the least robust mode is used then $T(N_a) = 6590$ microseconds. The constant overhead O_c is about 7% of $T(N_a)$ in this case.

Fig. 5.2 depicts the simulation results for the point-to-point scenario in the access powerline network. Through the BAT selection we achieve different values of PER on this link. As a result, the original ARQ produces retrans-

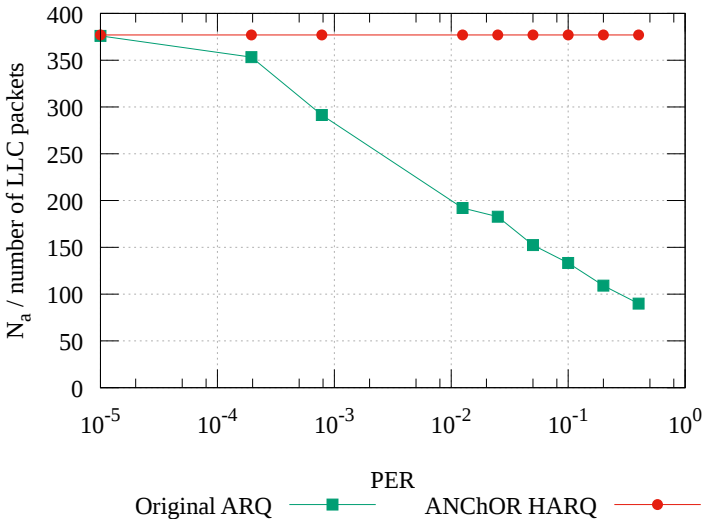


Figure 5.2: Simulation. Average MPDU size

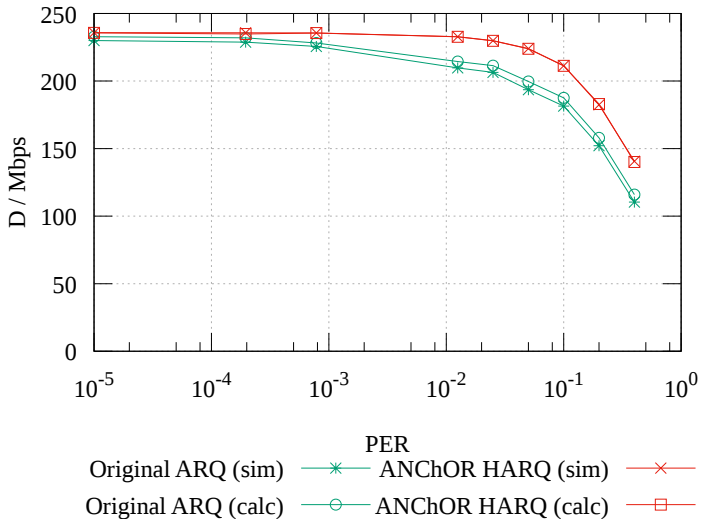


Figure 5.3: Simulation. Data rate on LLC after CRC check

missions that lead to the head-of-the-line problem. The ANChOR HARQ uses

the same $N_m = 376$ LLC packets and $G_s = 100$. We see that N_a for ANChOR does not change with the growing PER. It is explained with that the ordering of the coded symbols inside of the given generation does not occur. The bigger N_a leads to a higher data rate. In fig. 5.3, you can see both simulation and calculation results using eq. 5.2:

$$D = \frac{N_r \cdot P_s}{O_c + N_a \cdot O_p}, \tag{5.2}$$

where N_r is the average number of LLC packets in a MPDU that pass CRC check and P_s is the LLC packet size (536 bytes). N_a and N_r are obtained by means of simulation. O_c and O_p we take from table 5.3 for the least robust mode.

5.2 Medium and PHY requirements for "good" cooperation

With ORP, the receivers of the same PHY frame can cooperate in the sense of the mutual decision on the forwarding of the received information. Here, we investigate the requirements on the communication medium and the PLC modems that are necessary for the cooperation. The cooperation is "good" if it brings the positive gain of ORP over CSPR. The basic requirement is the presence of at least two vertices in the cooperation group(s). In the following, we analyze if this simple requirement can be fulfilled in realistic PLC scenarios.

Say vertex v wants to cooperate with vertices in $L(v) \subseteq V^+(O(v))$. The exclusion rule of BRR-E requires that $\forall u \in L(v) : \; p(u) > p(v)$. The ANChOR has also another rule that can reduce the size of $L(v)$, N_c. Each vertex $u \in V^+(O(v))$ should have sufficiently high priority and the loss ratio $\varepsilon_{(v,u)}$ should be sufficiently small so that adding u to $L(v)$ increases $p(v)$ on at least certain threshold value (section 3.7). From another side, $\varepsilon_{(v,u)}$ should be sufficiently big to avoid very small filtering coefficients. With very small $\varepsilon_{(v,u)}$, the cooperation is decreased as well. The value $\varepsilon_{(v,u)}$ can be influenced by v through the selection of the modulation and coding scheme. With CSPRs, in BB-PLC this value is below 0.01 [16]. With ORPs, higher values may be desired in order to increase the cooperation. The sending vertex v should be able to select certain $\varepsilon_{(v,u)}$. Here, we analyze how v can setup the transmission in order to achieve certain $\varepsilon_{(v,u)}$.

The value $\varepsilon_{(v,u)}$ depends on the packet size P_s and the bit error probability b after FEC decoding. Applying the hard decoding, if the flipping of each bit in the packet of n bits is independent and equally probable then it can be also called the Bit Error Ratio (BER). Hence, the PER can be calculated as follows:

$$\varepsilon_{(v,u)} = 1 - (1 - b)^n. \tag{5.3}$$

For a given n, we can achieve a certain $\varepsilon_{(v,u)}$ only by choosing a certain b. The value b depends on FEC, modulation and SNR. G.hn [4] specifies the check functions for QC-LDPC FEC and constellations for modulated symbols in the range from BPSK to QAM4096. A realistic SNR can be obtained using the simulation model in [68]. Normally, b is expressed as a function of SNR for a certain modulation and coding rate. In fact, for multi-carrier data transfer, not only the average value of SNR but also its spectral shape plays a role. If the SNR is not flat in frequency domain then the bit error probability before decoding, b', is unequal for different subcarriers. I.e., the bit errors occur in bursts, which impacts the FEC effectiveness. The burstiness of the bit errors is typical in PLC [78]. It may also appear when the adaptive bit loading is applied (different modulations for different carriers). We assume that by using a scrambler we are free of such problem and the bit errors before decoding are uniformly distributed in the FEC codeword. In G.hn and IEEE 1901 the bits of the entire PHY frame can be scrambled. With this assumption, bursts of bit errors will not appear in the bit sequence after decoding as well, i.e. the assumption in eq. 5.3 holds. In this case, b can be averaged over all subcarriers as follows:

$$b = \sum_{i \in F} b_i / \sum_{i \in F} m_i, \tag{5.4}$$

where $F = \{0, 1, ...k\}$ is the set of subcarriers indices, b_i is BER after decoding for the i-th subcarrier if all bits of the given FEC codeword are modulated on this subcarrier and m_i is the modulation index used on the subcarrier i. In fig. 5.4, b_i for particular i is shown. In this calculation, we used BPSK modulation and QC-LDPC FEC with the check matrices from G.hn [4] (simulation with pyLDPC [79]). Given the spectral shape of the SNR, we can use the SNR for a particular subcarrier snr_i to find b_i in fig. 5.4. Analog mapping for other modulations and coding rates can be obtained. Substituting b_i values in eq. 5.4

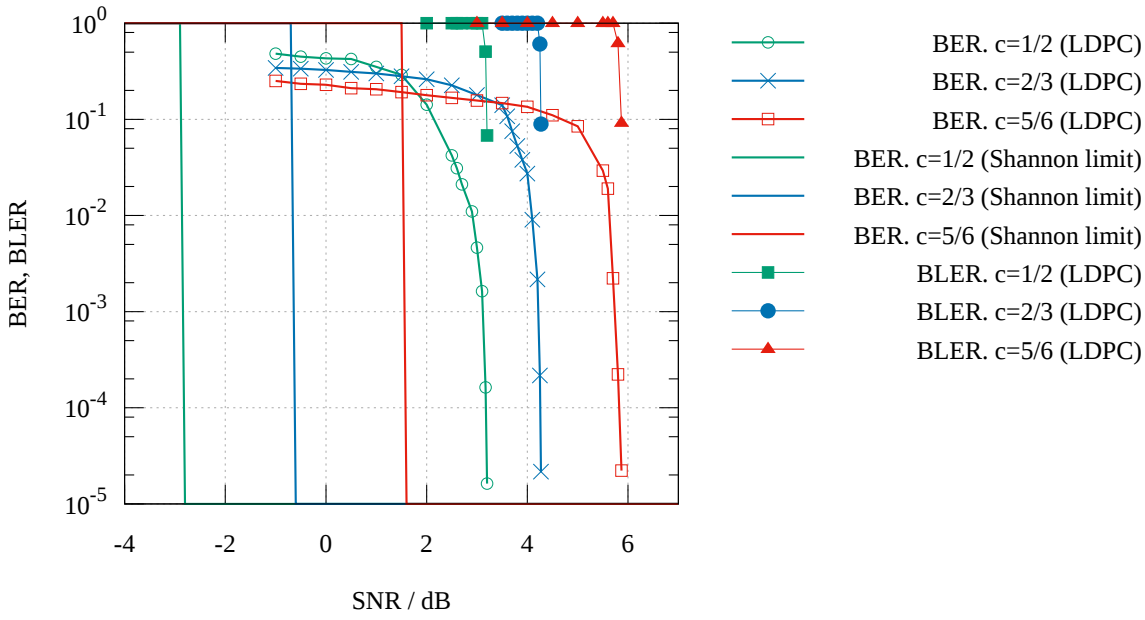


Figure 5.4: BER and BLER characteristics of QC-LDPC FEC in G.hn using BPSK modulation and FEC block size $M_F = 540$ bytes. Mother codes of check matrices in [4]

we find b. Using eq. 5.3, the corresponding PER value is obtained. It equals the BLock Error Rate (BLER) in fig. 5.4 when all subcarriers are modulated with BPSK. In our work, BLER is analog to PER because each FEC codeword is managed as a single data unit with a CRC block on the LLC. From fig. 5.4, we see that BLER is very steep, i.e. a minor change in SNR can produce a substantial change in BLER. This may result in certain cooperation limits.

It is obvious that the vertex $u \in V^+(O(v))$ is not able to cooperate with v if $\varepsilon_{(v,u)} = 1$. If $\varepsilon_{(v,u)} = 0$, using the BRR-E exclusion rule, it makes no sense for any vertices with lower priority than $p(u)$ to cooperate (section 3.3.2). So, it is desired to have $0 < \varepsilon_{(v,u)} < 1$.

The size of the cooperation group $N_c = 2$ is easy to achieve. Say, v has two neighbour vertices $u, w \in V^+(O(v))$ that experience SNRs with average values snr^u and snr^w correspondingly. Let $snr^w > snr^u$ and $p(u) > p(w)$. Then, v should calculate a BAT that guarantees $\varepsilon_{(v,u)} \in (0, 1)$. If $\Delta snr = snr^w - snr^u$ is high then $\varepsilon_{(v,w)} = 0$. Otherwise, $\varepsilon_{(v,w)} > 0$. In any way, in accordance to BRR, u and w can cooperate.

However, having more than two vertices in the cooperation group can be

problematic. For simplicity, let all subcarriers be modulated with the same number of bits per carrier. Then, the size of the cooperation group, k, can be achieved if $n \geq k-1$ vertices experience $\varepsilon_{(v,u)} \in (0,1)$, $u \in V^{+}(O(v))$. Analog to say that Δsnr for any pair of n vertices should be small enough. From fig. 5.4, Δsnr corresponding $\varepsilon_{(v,u)} \in (0,1)$ is far less than 1 dB. If the realistic difference in SNR for several cooperating candidates is below 1 dB then $k > 2$ is possible.

5.3 Access network scenario

Here, we obtain the realistic difference in SNR for the access power network in the form of a straight line. The PLC modems are installed equidistantly. The distance of 50 meters is realistic for the electricity metering network in Germany [80]. The background noise is set to the best case scenario defined in [5]. The head-end modem sends the PLC signal using a transmission PSD as defined in G.hn [4] (no power balancing). All other modems measure the observed SNR (fig. 5.5).

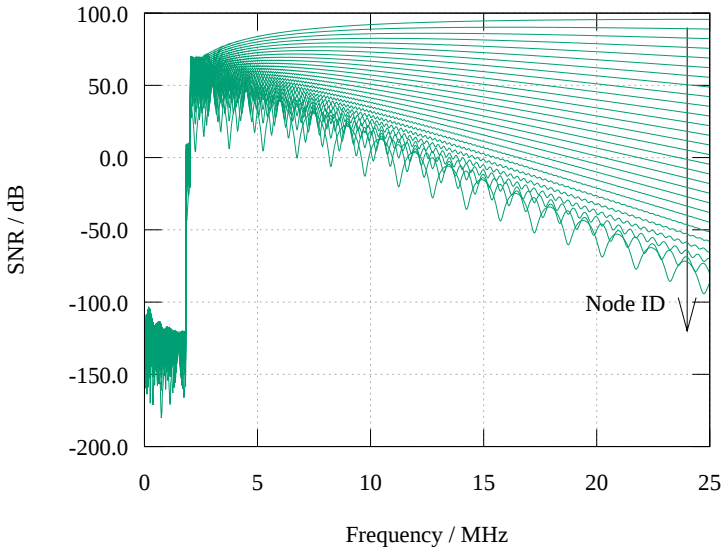


Figure 5.5: ns-3 simulation of SNR. Long line scenario (50 meters between nodes, 39 receiving nodes). G.hn TX PSD [4]. Worst case scenario of coloured noise from [5]

Small SNR values in lower frequency range happens due to the carrier masking in G.hn. Each curve corresponds to the SNR of the certain node.

The highest SNR experiences the direct neighbor of the sending node. We see that for almost the complete frequency range the difference in SNR between neighbor modems exceeds 1 dB. Note that increasing the distance between the modems or increasing the noise will just widen the SNR gap between neighbor modems making the cooperation even less possible. Thus, it is not feasible to have $N_c > 2$ in the access PLC network scenario. Nevertheless, even in this case, the gain of ORP over CSPR can reach up to 80% (see figs. 2.5a - 2.5d)

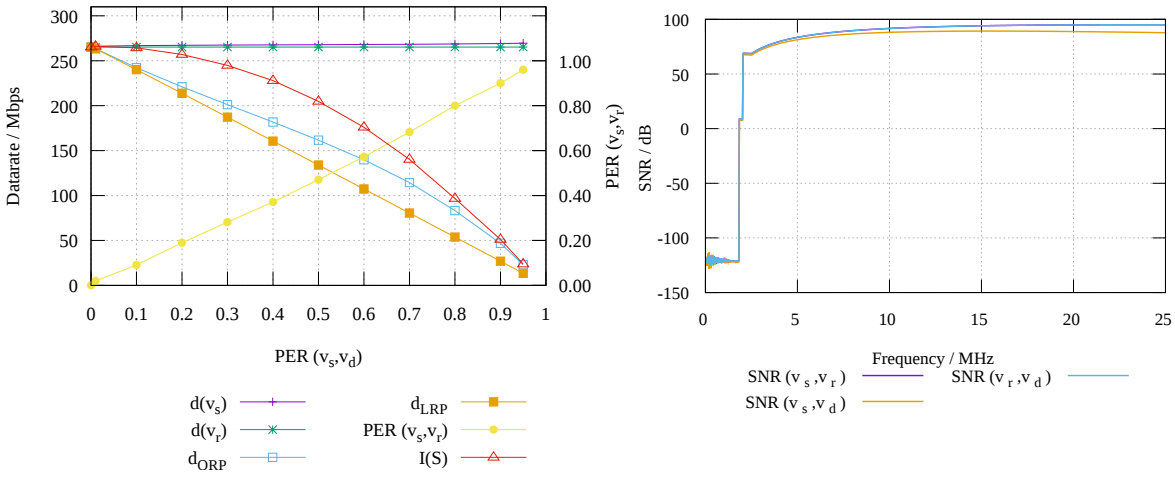
Now, we concentrate on the example with $N_c = 2$. Having $N_c > 1$ is a necessary but not a sufficient requirement for the effective cooperation. As figs. 2.5a - 2.5d demonstrate, it is important to achieve certain PER values to increase the gain of ORP over CSPR. The sending vertex can manipulate the bit loading table (BAT) to achieve different BER that results in the target PER value (section 4.3.1). But the question is if it makes sense to increase the PER that can result in the drop of the effective throughput. Surely, by increasing the PER, higher modulation indices can be used that increases the sending data rate. The effective throughput given as a product of the sending data rate and the success packet rate can have both negative and positive gradient depending on the protocol setup. We analyze this tradeoff on a simple example.

We investigate a fragment of the chain topology shown in fig. 1.1c that comprises the triangular scenario ($V = \{v_s, v_r, v_d\}$). We use the previous setup of the access network with 40 modems and analyze the performance of the first three modems from the begining of the chain. We do not shorten the line to leave these three modems only because then the reflection of the PLC signal from the other end of the line becomes significant that affects CTF and makes the results not so clear. In our analysis, we evaluate the upper bound on the achievable data rate with ORP D^O using eq. A.5 and CSPR D^C using eq. B.1. The values D^O and D^C do not consider the retransmission, protocol headers and other types of protocol overhead. Also, we observe the information flow $I(S)$ on the cut $S = \{v_s\}$. From the min-cut max-flow theorem, D^O is bounded by the flow on this cut. For a Packet Erasure Channel (PEC) without correlation of CTFs, $I(S) = d(v_s)\cdot(1-\varepsilon_{(v_s,v_r)}\cdot\varepsilon_{(v_s,v_d)})$, where $d(v_s)$ is the sending data rate of v_s. If we manipulate the BAT to achieve a certain $\varepsilon_{(v_s,v_d)}$ then $I(S)$ changes through the increase of $d(v_s)$ and decrease of $(1-\varepsilon_{(v_s,v_r)}\cdot\varepsilon_{(v_s,v_d)})$. We set the background noise to the worst scenario that is defined in [5] and

use the coding rate $c = 20/21$. Instead of QC-LDPC G.hn FEC we use the idealized FEC described in section 4.3.

Our simulation results were obtained for topologies with different distances between the neighbour modems. For each distance ds, we change the BAT on v_s to achieve a different $\varepsilon_{(v_s,v_d)}$ in the range $[10^{-5}, 1)$. The relay always calculates the BAT to reach $\varepsilon_{(v_r,v_d)} = 10^{-5}$.

We start the analysis for $ds = 50$ meters. For this distance, the SNR in the complete frequency band exceeds 70 dB (see fig. 5.6b) making the usage of the highest modulation index for all subcarriers possible (note that SNR (v_s, v_r) equals SNR (v_r, v_d)).



(a) Performance metrics

(b) SNRs for all edges

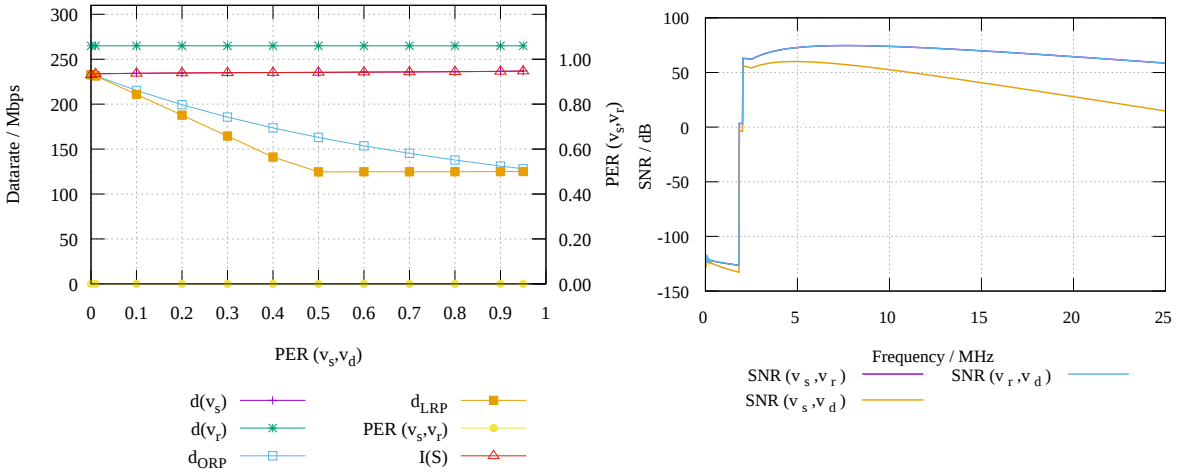
Figure 5.6: Access network. Evaluation for the first three vertices in the line with chain topology. 40 modems in total. $ds = 50$ meters. Idealized FEC. $c = 20/12$

As a result, when increasing $\varepsilon_{(v_s,v_d)}$ the source sending data rate $d(v_s)$ cannot increase (see fig. 5.6a). Therefore, $I(S)$ is constantly decreasing. It is impossible to achieve $D^O\big|_{\varepsilon_{(v_s,v_d)}\in(10^{-5},1)}$ greater than $D^C\big|_{\varepsilon_{(v_s,v_d)}=10^{-5}}$. In other words, it makes no sense to increase $\varepsilon_{(v_s,v_d)}$ creating cooperating opprotunities because D^C (about 270 Mbps in fig. 5.6a) with BAT on v_s optimized for almost zero packet loss probability is always greater than D^O for non-zero $\varepsilon_{(v_s,v_d)}$. Nevertheless, for given $\varepsilon_{(v_s,v_d)}$, D^O is greater or equal D^C, which agrees with the conclusion in appendix B. In the following we distinguish between the ORP

gain over CSPR as defined in eq. B.2 and the effective gain:

$$\alpha' = \left(D^O\big|_{\varepsilon_{(v_s,v_d)} \in (10^{-5},1)} - D^C\big|_{\varepsilon_{(v_s,v_d)}=10^{-5}} \right) / \, D^C\big|_{\varepsilon_{(v_s,v_d)}=10^{-5}} \, . \tag{5.5}$$

We increase ds in order to decrease $d(v_s)$ for $\varepsilon_{(v_s,v_d)} = 10^{-5}$. Then, for $\varepsilon_{(v_s,v_d)} > 10^{-5}$, the data rate $d(v_s)$ has a room to grow. In the range $ds \in [220; 250]$ meters, $\varepsilon_{(v_s,v_r)}$ becomes independent of $\varepsilon_{(v_s,v_d)}$. For $ds \geq 250$ meters, $\varepsilon_{(v_s,v_r)} = 0$ because of significant difference in SNRs on edges (v_s, v_d) and (v_s, v_r). As a result, $I(S) = d(v_s)$.



(a) Performance metrics

(b) SNRs for all edges

Figure 5.7: Access network. Evaluation for three vertices in the head-of-the-line. 40 modems in total. $ds = 300$ meters. Idealized FEC. $c = 20/12$

The data rate $d(v_s)$ with $\varepsilon_{(v_s,v_d)} = 10^{-5}$ decreases by 10% at $ds = 300$ meters. Nevertheless, in fig. 5.7a, we see that $d(v_s)$ just slightly changes with the growth of $\varepsilon_{(v_s,v_d)}$. It has the following reasons. First, FEC creates a steep dependence of BER from SNR. Consequently, when the tolerable BER increases, for a given SNR, the BAA cannot use much higher modulation indices. Therefore, the gradient of $d(v_s)$ should increase when selecting FEC with the lower coding rate.

As shown in fig. 5.8, for the case with no coding, Δsnr for subsequent modulation indices is about 3-5 dB. This plot was obtained using the simulator in [68]. It provides the capacity for Discrete input Continuous output Memoryless Channel (DCMC) for given SNR and modulation. First, we ob-

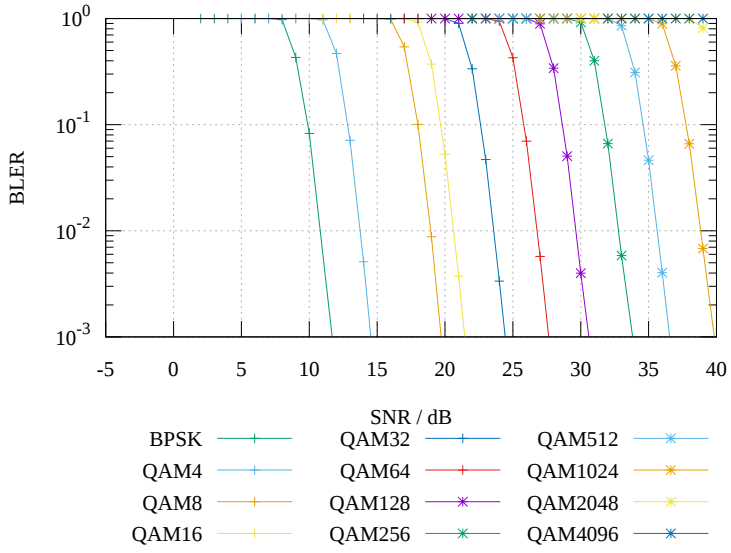


Figure 5.8: BLER as a function of SNR. $c = 1/1$. $M_F = 540$ bytes

tain the BER as the ratio of DCMC capacity to the number of bits loaded to the carrier. Then, using eq. 5.3 we calculate the BLER.

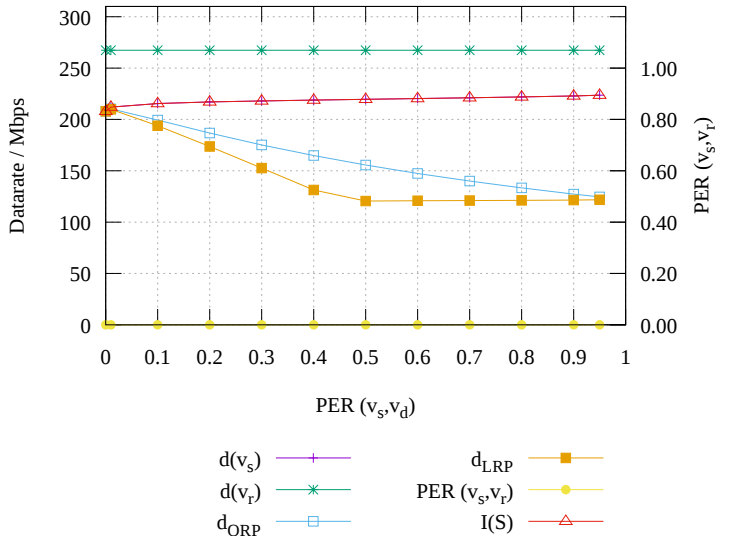


Figure 5.9: Performance metrics. No FEC. Packet size 540 byte. Adaptive BAA [2]

In fig. 5.9, we show the simulation result with the above setup but with no coding on PHY. We see that the gradient of $d(v_s)$ has increased. Nevertheless, $\alpha' \leq 0$ (see eq. 5.5).

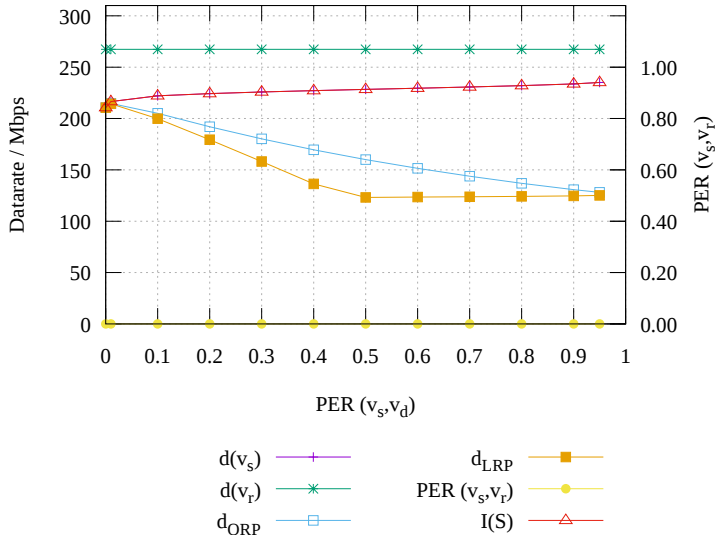


Figure 5.10: Performance metrics. No FEC. Packet size 32 bytes. Adaptive BAA [2]

We can further increase the gradient reducing the packet size (each packet is controlled with CRC). In this way, the dependence of PER from BER becomes less steep.

We can see in fig. 5.10 that the gradient of $d(v_s)$ has further increased. Still, $\alpha' \leq 0$.

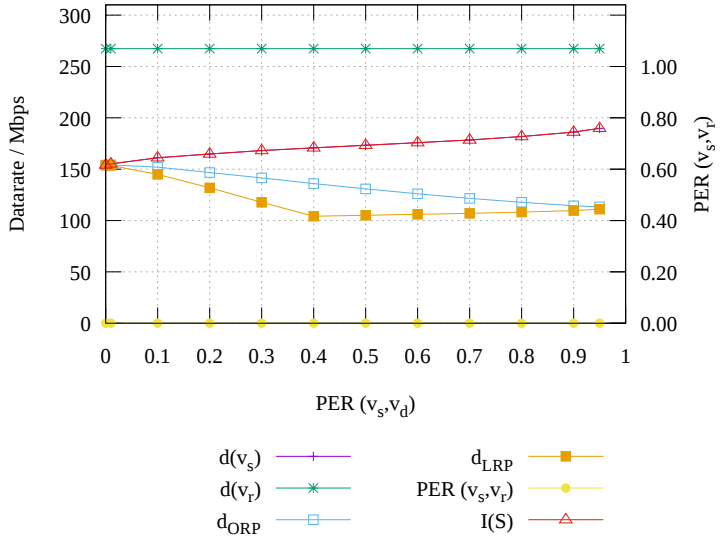


Figure 5.11: Performance metrics. No FEC. Packet size 32 bytes. Single modulation index BAA

Then, we replace the adaptive BAA [2] with the more simple one. It works similar to the previous BAA but instead of reducing the modulation on a particular subcarrier we notch the subcarrier completely. In this way, all active subcarriers are modulated in the same way. Such a BAA allows increasing the number of bits per symbol faster than BAA from [2] for the same increase of the tolerable BER.

Now, the gradient is further increased (see fig. 5.11) and α' becomes positive. Still, the gain is not considerable. Moreover, the resulting protocol setup is unrealistic. Using no FEC the number of bits per OFDM symbol decreases significantly. Also, the packet size is so small that the protocol overhead of LLC layer comprises about 100%. Eventually, adaptive bit loading allows a more accurate adjusting of the sending data rate and therefore yields much better performance than it is achievable with the single modulation index BAA. With all these arguments, we conclude that an ORP in BB-PLC with the current setup of the simulation model is not applicable for the access power network.

So far we are aware of only one study of ORP with NC on the DLL in PLC. In [38], the authors advocate the successful implementation in NB-PLC. They obtained the positive result due to the difference of protocol stacks of BB- and NB-PLC.

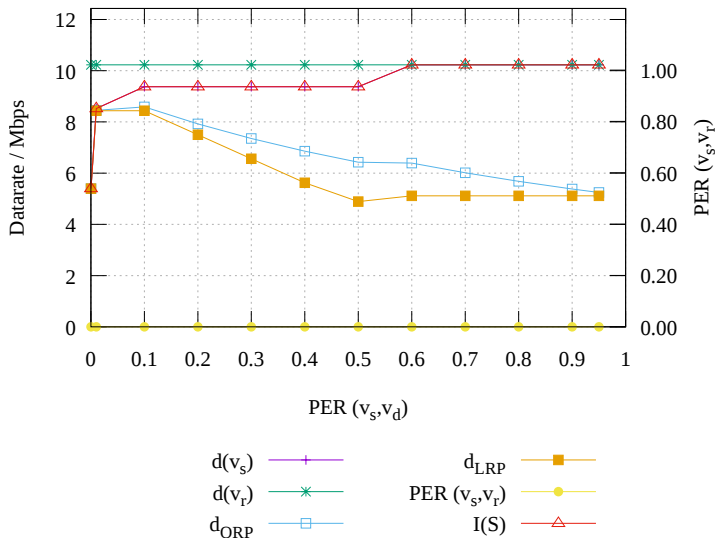


Figure 5.12: Performance metrics. No FEC. Packet size 32 bytes. Single modulation index BAA. 36 subcarriers used

In NB-PLC, the packet size on DLL is naturally smaller because of the lower data rates. In [38], the authors use a packet size of 25 bytes (compared with 540 bytes in G.hn). They do not mention about the application of FEC on the PHY in the FSK modem (ST7538Q). It means that our unrealistic setup of BB-PLC protocol is realistic for NB-PLC. Another significant difference of NB- to BB-PLC lies in the much smaller number of subcarriers. Modern NB-PLC standards like ITU-T G.9903 use OFDM. The number of subcarriers in G.9903 equals 36 only. And, all carriers should use the same modulation index (unless notched). In the same time, G.9960 in the 25 MHz spectrum can use up to 979 subcarriers (counting only the ones not notched by default). As a result, the number of bits per symbol in G.9903 can change more significantly with the increase of the tolerable BER.

In fig. 5.12, we demonstrate the effect of reducing the number of subcarriers by simple notching of G.hn subcarriers leaving only 36 carriers in the beginning of G.hn spectrum. Here, $\alpha' \approx 0.5$. The application of ORP in NB-PLC we leave for further research.

The above conclusions relate not only to the access power network but also to the in-home power network. Surely, the difference of SNR at several receivers in the latter scenario can be smaller. It means that the size of the cooperation group can be greater than two. Still, the problem with the gradient of $d(v_s)$ remains.

Influence of impulsive noise

The impulsive noise component can play the vital role in the application of ORPs in PLC. Due to the time variation of the channel capacity, the sender node should select more robust modulation and coding schemes to ensure the error-free communication. As a result the function of $d(v_s)$ from the PER can be less steep. We model only the background noise in the access power network scenario due to the absence of the measurement results of the other noise types in such an environment in the BB-PLC frequency range.

5.4 In-home scenario

In the following simulations, we pertain no simplifications from the previous chapter. We use 25 MHz frequency band (979 subcarriers), an idealized MDS

coder with a coding rate $c = 20/21$ and the BAA from [2].

We generate the topology of the in-home network using the algorithm in [69]. Then, we place the PLC modems randomly in the generated household. Before the communication starts, each vertex measures the SNR to each other vertex. With the measured SNR and target PER=10^{-5} it calculates BATs, one for each neighbour vertex. Afterward, using Dijkstra's algorithm the best CSPR route is found. Each vertex selects such a BAT that corresponds the next hop vertex. The selected BAT is used for modulation of all PHY frames sent by this vertex.

5.4.1 Modeling packet erasures

In the in-home scenario, besides the background noise we also use the simulation model of the impulsive noise from [71]. For in-home PLC, the following types of noise are typical [81]: impulsive noise synchronous with the mains frequency (Type 1), periodic asynchronous impulsive noise (Type 2), sporadic asynchronous impulsive noise (Type 3), and colored noise (Type 4) composed of the background and narrowband noise. We adopted the numbering of noise types for the simplicity of referencing.

In [71], the noise types 2 and 3 are not modeled. Though their impact can be significant. As the measurement results in [42] show, these noise components have the wide bandwidth (11-13 MHz) and the power between -35 dBm and -14 dBm (compare with -55 dBm of transmission PSD mask in G.hn). The noise of type 2 has an impulse duration up to 1.5 microseconds. Thus, the single impulse can damage at most one G.hn OFDM symbol (42.24 microseconds [4]). But, the impulses occur very often (12.6-217.2 kHz). The G.hn OFDM symbols are sent with frequency about 23.6 kHz. It means that each OFDM symbol can be impared with several impulses of this noise type. The noise of type 3 has the bursts of a longer duration (up to 20 microseconds) and can corrupt 1-2 OFDM symbols. The frequency of this bursts is not advocated by [42].

Implementing additional noise types 2 and 3 should add more time dependence of PER. In this case, the CSPR should use a more robust PHY setup to guarantee PER = 10^{-5}. An ORP does not require that low PER. Consequently, the communication with the less robust PHY setup should be possible. For example, the BAT can be calculated for the average instead of

the lowest SNR (over mains period). In such a scenario, α' should be positive. Besides that, the noise of type 3 adds randomness to the packet losses, that increases the PEC channel capacity reducing the PER correlation.

We considered the noise of type 1 as the noise with the highest power. It occurs in trains of impulses with an overall duration of up to 500 microseconds [42], which is the duration of about 12 OFDM symbols in G.hn. Up to 10 noise bursts per the mains period may occur. With the mains frequency of 50 Hz, up to 25% of OFDM symbols are impaired with this noise type. Using the measurement results from [82], this value is even greater. The authors provide a statistical model of the noise, whereby the noise train duration is modeled with the gamma distribution. Using the parameters of this distribution from [82], a mean value is 7,57 milliseconds, which is approx. 38% of the mains period duration.

Although we implement the strongest component of the impulsive noise, we cannot obtain the realistic distribution of the packet erasures on the LLC layer. One reason lies in the idealistic BAT calculation. Before communication starts, using the global knowledge of the noise PSD in the future that is available in the simulator, we evaluate BATs for the lowest SNR on each link. The lowest SNR corresponds to the burst of the impulsive noise. As a result, in the periods with no impulsive noise, the communication is error-free (PER = 0). When the noise burst occurs, we receive packets with a PER that we targeted during the BAT calculation. In the error-free period, no gain of ORP is possible. The system behavior in the period with non-zero PER is similar to the case when we have only the colored noise. Analog scenario was studied for PLC in the access power network (see section 5.3). We run into the same problem that the slight increase in the sending data rate causes a jump in the PER. And, α' remains negative for any target PER value. Therefore, we decided to switch off the impulsive noise model from [71] and implement the simplified method for the packet erasing.

In reality, the amplitude of the noise signal in the burst is unequal along the burst. Also, it may differ for several bursts. Depending on the quantization frequency, it can also make sense to consider the position of the noise bursts alongside the OFDM symbol. Implementing the scrambler, the interleaver, and the FEC method on the PHY layer is also important for obtaining of the

realistic distribution of the packet erasure probability on the LLC layer P_e. The model considering all these effects and implementing all impulsive noise types may require a significant computational effort. Hence, for our research, it makes sense to use the hierarchical modeling. With this approach, the channel output and the PHY layer are described with their statistical models basing either on the measurement or on the simulation results. To the author's best knowledge, there is no P_e available for the BB-PLC solutions nowadays. Therefore, we describe the packet erasure as the Bernoulli process. The possible range of the erasure probabilities we motivate as follows.

If the burst of the impulsive noise occurs, the SNR value decreases significantly. Say, the background noise has power of -120 dBm/Hz [5] and the PLC signal transmission power of -55 dBm/Hz [4]. Let the signal attenuation be very small so that the reception power of the PLC signal equals -55 dBm/Hz as well. Then, SNR equals 65 dB. If a noise burst occurs with a power between -35 dBm/Hz and -14 dBm/Hz [42] then the SNR value decreases to -35...-25 dB. If the sender optimizes the BAT for a SNR of 65 dB then it will not be able to decode any FEC block transmitted in the period of the noise burst.

This assumption allows us to induce the packet erasures. We do not model the impulsive noise itself. Instead, the simulator erases the FEC blocks randomly on the PHY with a given PER. Upon PHY frame reception, we firstly check if the Shannon capacity of the channel (considering only colored noise) within the duration of the PHY frame is greater or equal the uncoded PHY frame size. If this check fails, the complete PHY frame is discarded. This decision is based on the network topology and the background noise power. If the frame is not discarded then, with the certain Bernoulli probability, we discard each of the received FEC blocks emulating the impulsive noise. This probability equals PER. We also assume that the positions of the erased FEC blocks do not conside with the actual bursts of the noise due to the application of the scrambler, which distributes the error bits alonside the complete PHY frame.

We select a wide range of possible PER values. Basing on [42] and [82], up to 25-38% of OFDM symbols can be corrupted with the synchronous impulsive noise. Although the results in both papers are based on extensive measurements, the obtained statistics differ a lot. In fact, due to the versatil-

ity of electrical appliances, the quality of PLC channel can be much different. Therefore, it is not possible to make an overall conclusion on the realistic range of PER for each household basing on the measurements in one of them. We use the range $[0.05; 0.55]$ as a tradeoff between available measurement results.

5.4.2 Linear Dependence Ratio (LDR)

In the previous chapter, we showed that ANChOR provides effective rules for the ORP operation that are based on the prediction of the feedback information. Remember that in ORP many nodes overhear transmissions receiving partially the same data. The common ORP problem lies in the forwarding of the duplicate messages. If NC is applied, the duplicates do not increase the rank of the receiver coding matrix because they are the linear combinations of already present information. The LDR p_l is successively reduced with ANChOR. In the SNS simulations, we showed that the p_l in particular scenarios can be just a few percent. In fig. 5.13, we show the p_l for the in-home scenario with three PLC modems using ANChOR as a part of the G.hn protocol. The simulation model is implemented in ns-3. In overall, p_l is very high. Each

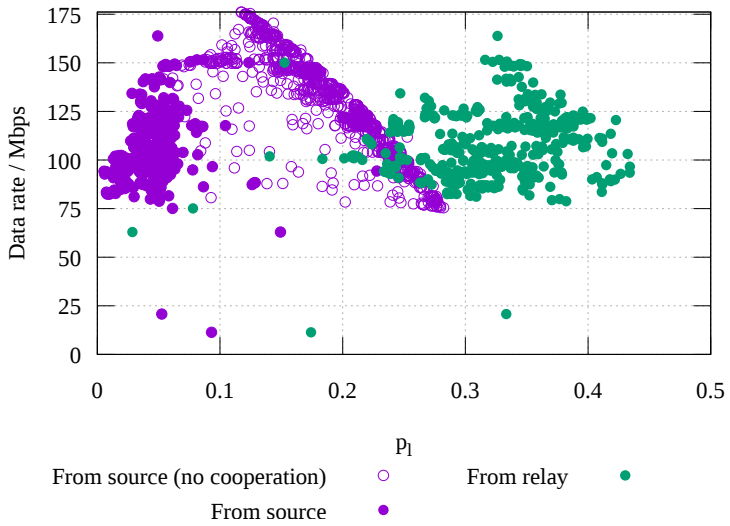


Figure 5.13: ANChOR in G.hn. In-home scenario. Triangular topology

point in this plot is obtained with a simulation run of 5 seconds duration as described above. For each simulation, the topology and loss ratios are individually generated. In many cases, the source does not cooperate with the relay. Remember that the exclusion rule filters the relay from the cooperation

group if its priority is less than the priority of the source. More simple in this scenario, the relay does not cooperate when $\varepsilon_{(v_s,v_d)} < \varepsilon_{(v_r,v_d)}$. In this case, p_l of codewords received by v_d from v_s lies between 10 and 30%. When the source does cooperate, it sends less data and p_l increases. But for codewords received by v_d from v_r, $p_l \in [20; 40]\%$.

The reason for such a bad performance in comparison with the SNS results can be explained as follows. In G.hn, there can be several codewords sent in a single PHY frame (the transmission of packet trains). It is not typical for the simulation in the SNS. In NB-PLC, the data is also not grouped in packet trains. In BB-PLC, the grouping allows significant reduction of PHY and MAC layer overhead (see section 5.1).

With the following simulation results, we demonstrate the reason for the increase of p_l in G.hn. For this purpose, we slightly modify the function that defines the transmission plan (TX plan).

Algorithm 10. TX plan calculation by vertex v

Form filtered plan FiP
for each generation g in reception window RW
  for each sender u
    $FiP[g]$ += $RW[g][v] \cdot p_f(v, u)$ (see filtering coefficient in eq. 3.12)
  if $FiP[g] > G_s$ **then** $FiP[g] \leftarrow G_s$
Form forwarding plan FoP
for each generation g in RW
  $FoP[g] \leftarrow FiP[g] + RP[g] - SP[g] \cdot c(v)$
  if $FoP[g] < 0$ **then** $FoP[g] \leftarrow 0$
Form initial TX plan TPi
for each generation g in FoP
  $TPi[g] \leftarrow FoP[g]/c(v)$
Form TX plan TP
for each generation g in TPi
  if $TPi[g] > a$ **then** $TP[g] \leftarrow TPi[g] \cdot b$ **otherwise** $TP[g] \leftarrow TPi[g]$

Here, RP is the retransmission plan, SP is the plan of already sent data and $c(v)$ is the coding rate used by v. Originally, ANChOR uses $TP = TPi$, which means that a vertex sends all information it has when it gains the channel

access. For this setup, the results in fig. 5.13 were obtained. Here, we add the parameters a and b that allow spreading the transmission of each generation in several PHY frames. If $a = 0$ and $b = 1$ then $TP = TPi$. We denote the parameters a and b as spreading limit and spreading factor correspondingly. First, we analyze the influence of the spreading factor on the performance in the Pt2Pt scenario. We place two PLC modems in the generated topology as described above and set the packet loss ratio between them $\varepsilon = 0.6$ in all simulations. We measure the throughput above UDP layer on v_d. Additionally, we measure p_l and N_a.

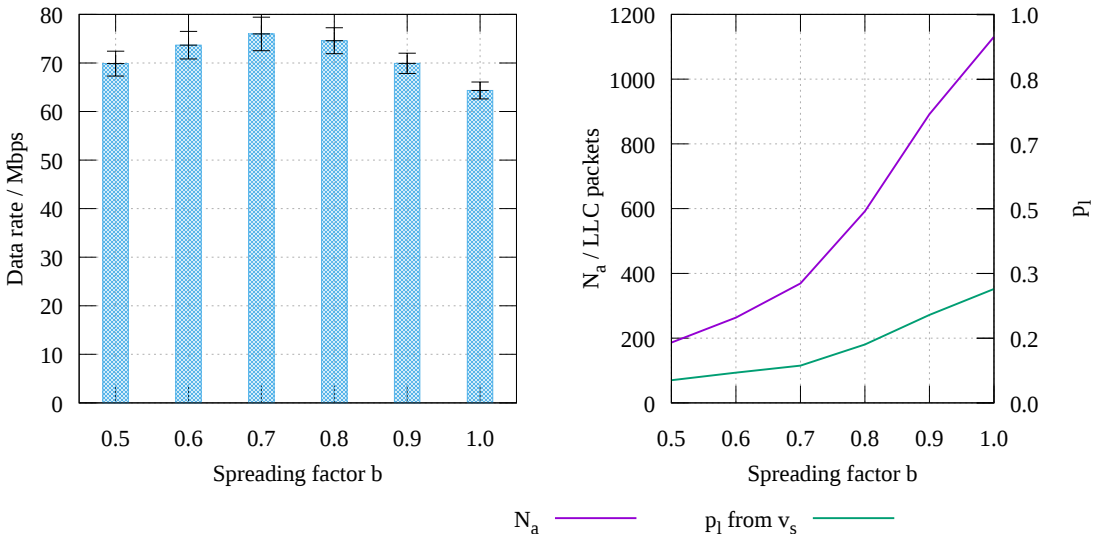


Figure 5.14: ANChOR in G.hn. In-home scenario. Pt2Pt connection: $\varepsilon_{(v_s,v_d)} = 0.6$. $a = 0$

As shown in fig. 5.14, decreasing b from the default value 1.0 to 0.5, p_l drops, which increases the data rate. But simultenously, N_a decreases that reasons more protocol overhead and reduces the data rate.

We repeat an analog experiment for the triangular topology. As the simulation results in fig. 5.15 show, p_l of codewords received by v_d from v_s lies below several percent for any value of b. p_l of codewords received by v_d from v_r drops from 27 to 3%. When we do not send the data in trains, like in SNS, we can say that $b \approx 1/G_s$ that is much less than the smallest b in fig. 5.15 (0.5). Therefore, it is obvious that the reason of substantial p_l when transmitting packets in trains lies in the granularity of the data transmission for each par-

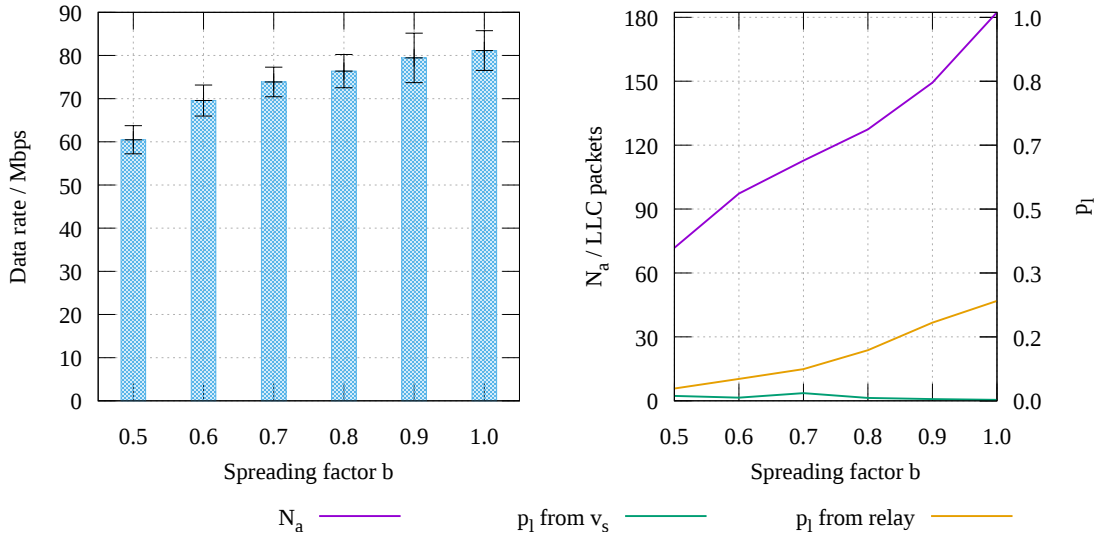


Figure 5.15: ANChOR in G.hn. In-home scenario. Triangular topology: $\varepsilon_{(v_s,v_r)} = 0.1$, $\varepsilon_{(v_s,v_d)} = 0.6$, $\varepsilon_{(v_r,v_d)} = 0.3$. $a = 0$

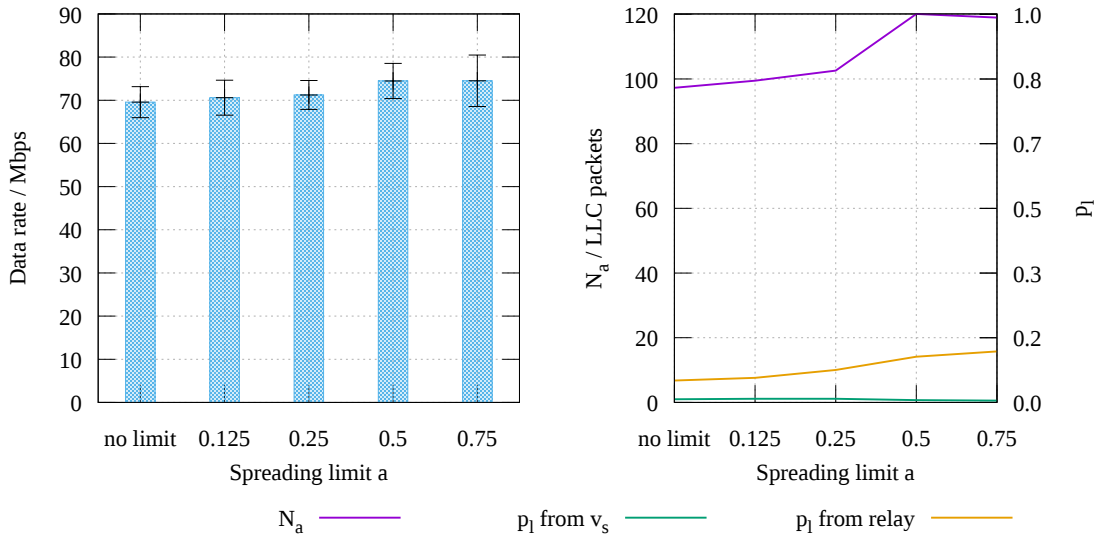
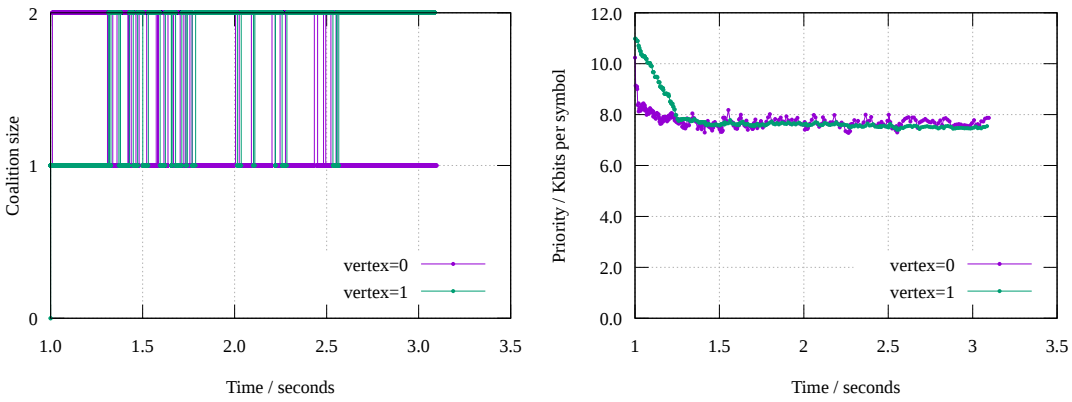


Figure 5.16: ANChOR in G.hn. In-home scenario. Triangular topology: $\varepsilon_{(v_s,v_r)} = 0.1$, $\varepsilon_{(v_s,v_d)} = 0.6$, $\varepsilon_{(v_r,v_d)} = 0.3$. $b = 0.6$

ticular generation. Thus, we can reduce p_l to the value below 1% if we reduce the spreading factor b. Nevertheless, the effective data rate above UDP does not increase in this case because N_a decreases. We can overcome this problem by setting a greater ACK window size. But it impairs the latency. We can also issue a spreading limit, i.e. a minimum number of packets in the initial TX plan before the spreading factor should be applied.

In the previous simulation, the spreading limit has been set to $a = 0$, which means "no limit".

Now, we set the spreading factor to be constant $b = 0.6$ and change a (see fig. 5.16). There is a slight increase in data rate for $a \geq 0.5$. But the highest data rate for $a > 0$ and $b < 1$ (about 75 Mbps) is less than the highest data rate with the default ANChOR settings $a = 0,\ b = 1$ (80 Mbps). Therefore, we use the default setup in the following simulations.



(a) Variation of coalition size

(b) Variation of priorities

Figure 5.17: Triangular topology: $\varepsilon_{(v_s,v_r)} = \varepsilon_{(v_s,v_d)} = \varepsilon_{(v_r,v_d)} = 0.3$. $v_s = v_0$, $v_r = v_1$. $\Delta p = 0$ (eq. 5.6), $b = 0.01$ (eq. 3.31)

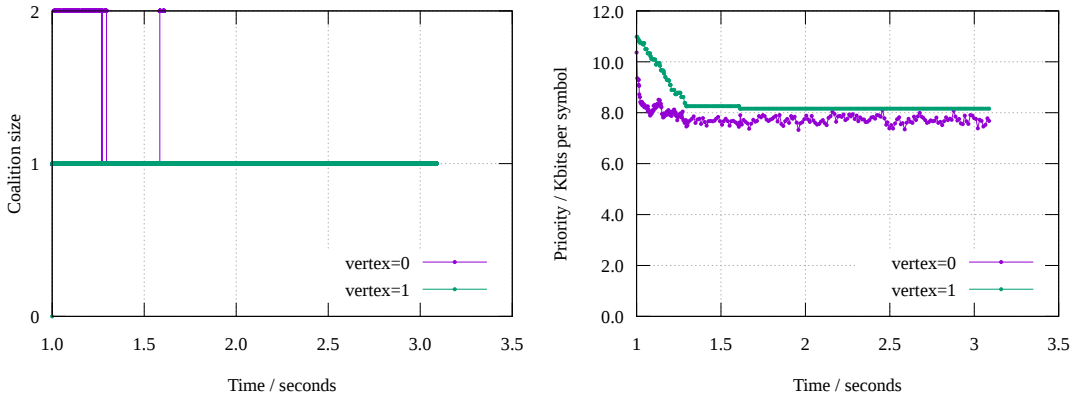
5.4.3 Worst case scenario

First, we design a scenario that may cause the instability of ANChOR and reduce the number of cooperation opportunities. For this purpose, each node uses the same loss ratio for each in-coming link. In this case, all vertices that can communicate with the destination directly, get the same priority. Since the priority is calculated based on the statistics of packet losses, it differs slightly with time. ANChOR forms its cooperation groups (and consequently

the filtering coefficients and coding rates) basing on the nodes' priorities. The comparison of priorities can be affected by the statistical errors. As a result, the nodes may constantly modify the cooperation groups and routing decisions. We avoid this effect by inducing a hysteresis for the priority comparison operators:

$$p(v) < p(w) = \begin{cases} 1 \text{ if } p(w) - p(v) > \Delta p \\ 0 \text{ otherwise,} \end{cases} \tag{5.6}$$

where $p(v)$ and $p(w)$ are priorities of nodes v and w correspondingly. The value Δp depends on channel stability and the size of the reception maps (averaging window of the filters for the loss ratio calculation).



(a) Variation of coalition size

(b) Variation of priorities

Figure 5.18: Triangular topology: $\varepsilon_{(v_s,v_r)} = \varepsilon_{(v_s,v_d)} = \varepsilon_{(v_r,v_d)} = 0.3$. $v_s = v_0$, $v_r = v_1$. $\Delta p = 1000$ (eq. 5.6), $b = 0.05$ (eq. 3.31)

In figs. 5.17a and 5.18a you can see the dependence of the cooperation group size over time for two cases with $\Delta p = 0$ and $\Delta p = 1000$, which is approx. 10% of the maximum priority. Remember that the priority metric has the same unit as the data rate. And, it is equal to the upper bound on the achievable data rate between the given node and the destination. In G.hn, it is more convenient to be expressed in bits per OFDM symbol. With 979 subcarriers and the maximum of 12 bits per subcarrier, the maximum priority equals 11748 bits per OFDM symbol. Additionally, in simulation in fig. 5.18a, we increase the threshold for adding the nodes to the cooperation groupd using eq. 3.31. The time $t \in [0; 1)$ in figs. 5.17a and 5.18a is not shown because it

does not refer to the performance on the LLC layer, which we study here. At $t = 0$ we send a single packet on the application layer that triggers the network discovery protocol of the IP layer. We do not send any other data till $t = 1$. Thus, the period $t \in [0; 1)$ refers to the warm-up of the IP layer. Here, we investigate the warm-up of the DLL layer only.

In fig. 5.17b, the priorities of the nodes are approximately the same because both relay and source connect to the destination with the link of the same quality. In addition, none of them has the permanent cooperation group. In fig. 5.17a they interchangeably add each other to its group. In figs. 5.18a and 5.18b, this effect is removed. Here, the source adds the relay to its coalition only in the transient phase on the beginning before the loss ratios are accurately estimated. Approximately at 1.4 seconds, the priority of the relay reduces to such value that the source skips the relay in the formation of its cooperation group (eq. 3.31). In contrary to fig. 5.17a, in fig. 5.18a there are only a few switches due to the usage of the hysteresis for the priority comparison.

Using $\Delta p = 1000$ and $b = 0.05$ we run the following simulations.

5.4.4 Analysis of in-home topologies

We generate 900 different topologies using the principle described in [69] and the simulation model from [71]. It requires the size of the apartment as a parameter. We use 60 m^2. Then, we place between 2 and 6 modems that corresponds to $m \in [0; 4]$ slaves. For each possible pair of v_s and v_d we run the simulation for a period of 5 seconds simulation time. For each pair of v_s and v_d, from the data collected in this period, we cut three seconds as the warm-up duration. Although, in many cases, this period is much shorter (e.g. see fig. 5.18b). For the data collected in the remaining two seconds, we calculated the data rate D, average latency L, and average jitter J on the application layer. The dash sign for L and J we omit for simplicity. We notice that for the in-home topology, all vertices select the highest PHY data rate (about 260 Mbps). In the remaining two seconds after the warm-up period, they can transfer about 89000 packets of 762 bytes size on the application layer. Alongside with the calculation of the mean values, we evaluate also the confidence intervals. Although two seconds is a short period, nevertheless the big number of packets can be transferred in it. As a result, the 95% confidence interval size is below 5% of the measured mean value, which is a good level of

reliability (compare figs. 5.14, 5.15, 5.16).

Selection of generation size

The size of the coding block in ANChOR is defined through the generation size G_s. Here, we demonstrate the influence of G_s on the overall system performance.

In the original G.hn, $N_m = 376$ for data packets. In G.hn with ANChOR, it is defined as the product of G_s and the number of generations n. For the constant ARQ buffer size, $\{G_s \cdot n\}$, we run the simulation for different pairs of G_s and n. We use the following pairs for the comparison: $G_s = \{8, 16, 32, 64\}$ and $n = \{40, 20, 12, 5\}$. The pair $\{G_s, n\} = \{32, 12\}$ gives slightly different product $\{G_s \cdot n\}$. We select this pair basing on multiple simulations for $G_s \in [8; 64]$, $n \in [5; 40]$, and $m \in [0; 4]$. Some of these simulation results are described in the sections below. This pair gives the best tradeoff between L, D, and J. Therefore, we add it for comparison in all figures. The result of L, D, or J corresponding to $\{G_s, n, m\} = \{32, 12, 1\}$ is always marked with the green bar.

In fig. 5.19, we see that G.hn with ANChOR using the combination $\{32 \cdot 12\}$ allows achieving a slightly lower L and much better D than the original G.hn and any other combinations $\{G_s \cdot n\}$ (triangular topology with PER=rand(0.05,0.55)).

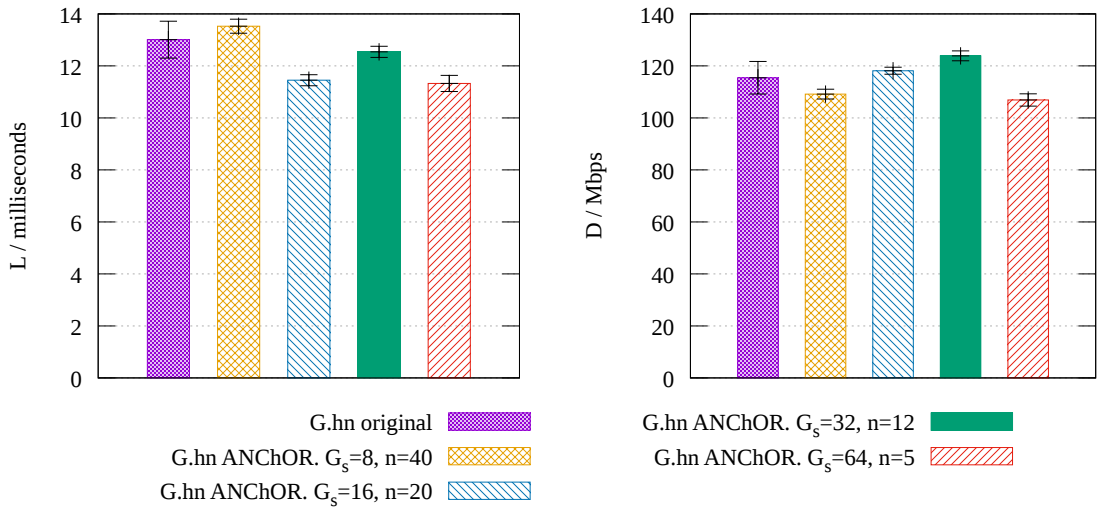


Figure 5.19: In-home scenario. Varying G_s, $m = 1$

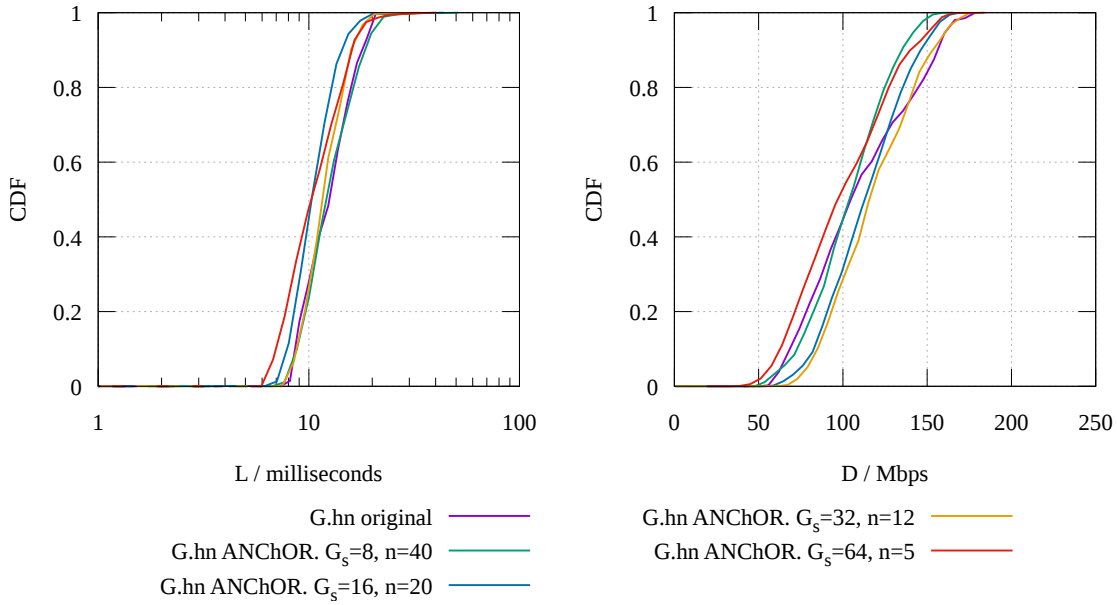


Figure 5.20: In-home scenario. Varying G_s, $m = 1$. CDFs

The higher D is explained by the greater average MPDU size N_a. Although the buffer sizes of both G.hn versions are approximately the same, the maximum number of LLC packets in the MPDU differs. The ANChOR can add also the redundant packets, which amount is not limited by $\{G_s \cdot n\}$. Also, it grows due to the reduction of the head-of-the-line problem with coding [3]. But the difference in N_a is not the only reason for the performance discrepancy. For example, adding the ANChOR header (with $G_s = 32$ approx. 25 bytes header for the packet with the payload of 515 bytes) increases L and reduces D. Also, channel access time with the ANChOR increases (here one more node can participate in communication), which has the same effect. The greater G_s is, the longer the ANChOR header is. As a result, D with $G_s = 64$ drops significantly (see fig. 5.19). For the small generations, $G_s = \{8, 16\}$, D drops as well. This can be explained with the change in the ANChOR accuracy. The ANChOR can make a better prediction on the amount of the useful redundant symbols and the filtering coefficients if G_s is bigger (see fig. 3.6). The value of $G_s = 32$ gives a compromise between these problems. D for the pair $\{32 \cdot 12\}$ is about 7% greater than D with the original G.hn setup.

In fig. 5.20, we show the CDF of the same results that are shown in fig. 5.19. Here we can see that the gain in D is greater for lower data rates (up to 40%). The achievable data rate is lower when the channel is much impaired with

noise. It means that the ANChOR significantly helps when the help is needed at most, i.e. when the channel quality is bad.

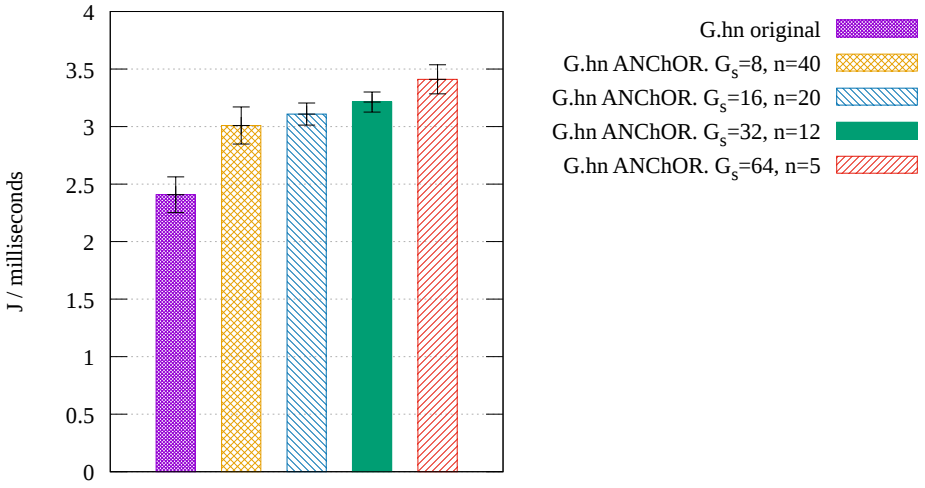


Figure 5.21: In-home scenario. $G_s = 32$, $n = 12$

L in most cases is less for G.hn with ANChOR (see fig. 5.19). It can be explained with that adding adding the redundant packets it is possible to deliver the complete message at the smaller number of attempts.

In fig. 5.21, we depict J for the same simulations. Due to block coding, for bigger generations, the application layer receives packets in bigger bursts. As a result, J grows. For the smaller G_s, it can also grow due to the intensifying of the head-of-the-line problem.

Selection of ARQ window size

In fig. 5.20, the size of the ARQ buffer n was constant. By varying this size we can balance between L and D. Certain applications can have a clear priority to either L or D. For example, offline video streaming should be optimized for D, while web searching requires a small L.

Setting $G_s = 32$, we vary $n = \{5, 12, 20, 40\}$. From the simulation results in fig. 5.22, we see that it is possible to adjust L and D in a wide range. With $n = 40$, the gain of ANChOR in D is 22% and with $n = 5$ the gain in L is 38%. If both data rate and L should be as good as possible, the highest gain of both metrics gives $n = 12$.

Again, observing the CDFs of D and L the gain of ANChOR is greater for

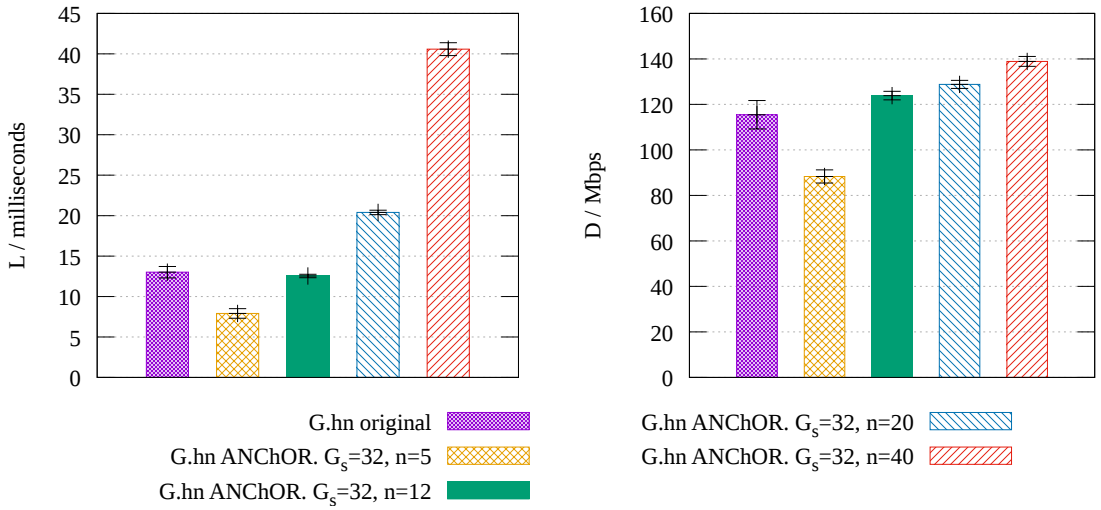


Figure 5.22: In-home scenario. Varying ARQ buffer size, $m = 1$

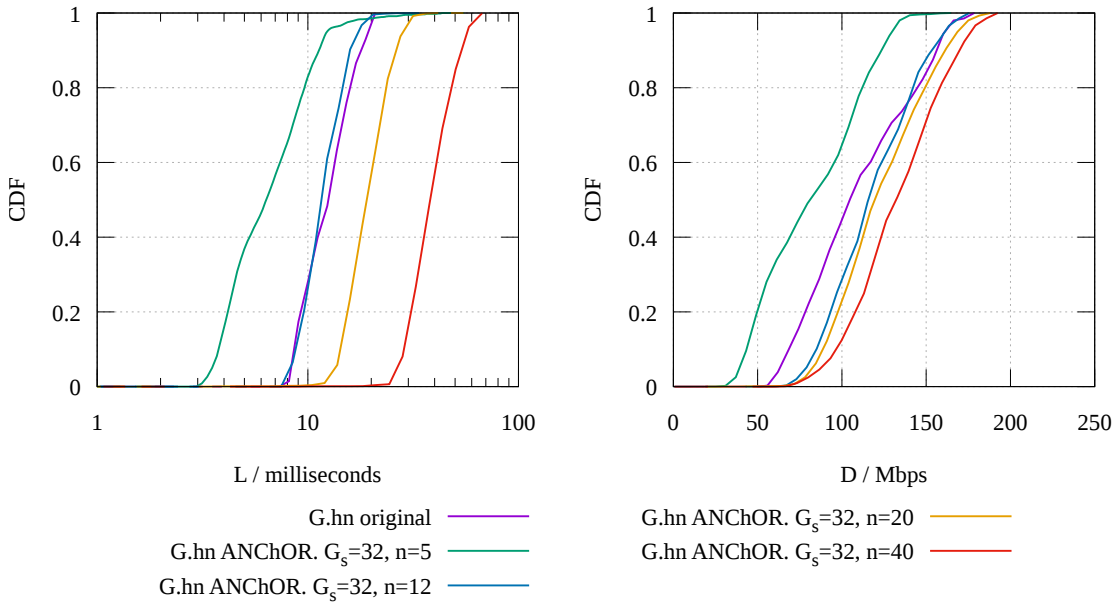


Figure 5.23: In-home scenario. Varying ARQ buffer size, $m = 1$. CDFs

the worse channel conditions (fig. 5.23). So, the gain in D is up to 40% and the gain in L is up to 60%.

Exploring the cooperation with several relays

Here, we investigate the influence of the cooperation on the performance. As the theoretical study of ORP in chapter 2 shows, cooperation brings only the positive gain of ORPs over CSPRs. So far, we analyzed a number of pitfalls that can demolish this conclusion in practice. With the same setup as before, and using $\{G_s \cdot n\} = \{32 \cdot 12\}$, we vary the number of relays $n_r \in [1; 4]$. As shown in fig. 5.24.b, the size of the cooperation group N_c of v_s never goes beyond three. For $n_r = 1$ in 70% of cases, $N_c = 1$, i.e. the relay is not used in the communication. This result agrees with BRR-E. In accordance with it, the relay should fulfill certain requirements to be added to the cooperation group. For example, in the scenario with one relay (G.hn ANChOR with $m = 1$ in fig. 5.24.b), the link (v_r, v_d) should be better than the link (v_s, v_d). With the random selection of PER on these links, the source should use the relay in 50% of cases. From fig. 5.24.b, we see that it occurs only in 30% of the cases. The reduction by 20% is reasoned by the additional rules like priority comparison (eq. 5.6) and the increase of the threshold priority for entering the cooperation group (eq. 3.31). With a greater number of the relays, there are more cases with the cooperation group size equal two. We conclude that in the networks with the bigger number of relays the maximum N_c is the same ($N_c = 2$) but the probability $P[N_c > 1]$ is higher.

A big N_c is necessary but not sufficient condition for the effective cooperation. In the case of a good direct link between the source and the destination, the source calculates a significant filtering coefficient and the relay(s) send very few data. In fig. 5.24.a, we show the proportion of traffic that the destination receives from the relay(s) η. In most cases, $\eta < 5\%$. For a greater number of relays m, it is more probable that the relays send more data because the source has more choice to select a "good" node for cooperation.

In fig. 5.25, we show the average values of L and D for different m. Here, we also compare the original G.hn and the G.hn with ANChOR. For the original G.hn, we show the results with $m = 1$ only.

From fig. 5.25, we see that for any m, D with ANChOR is greater. L is

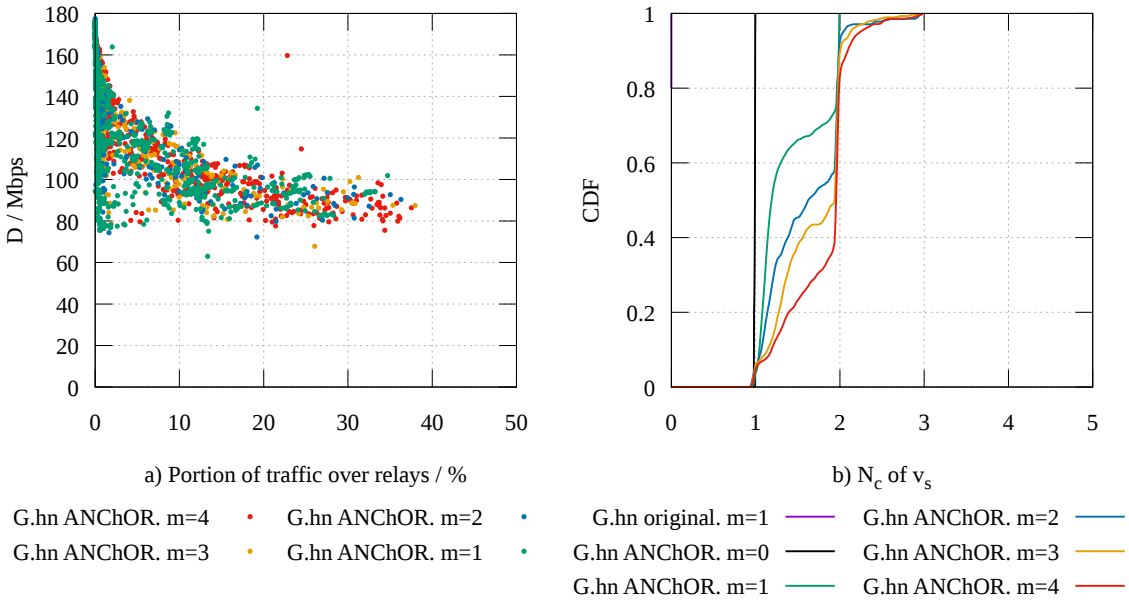


Figure 5.24: Usage of relays

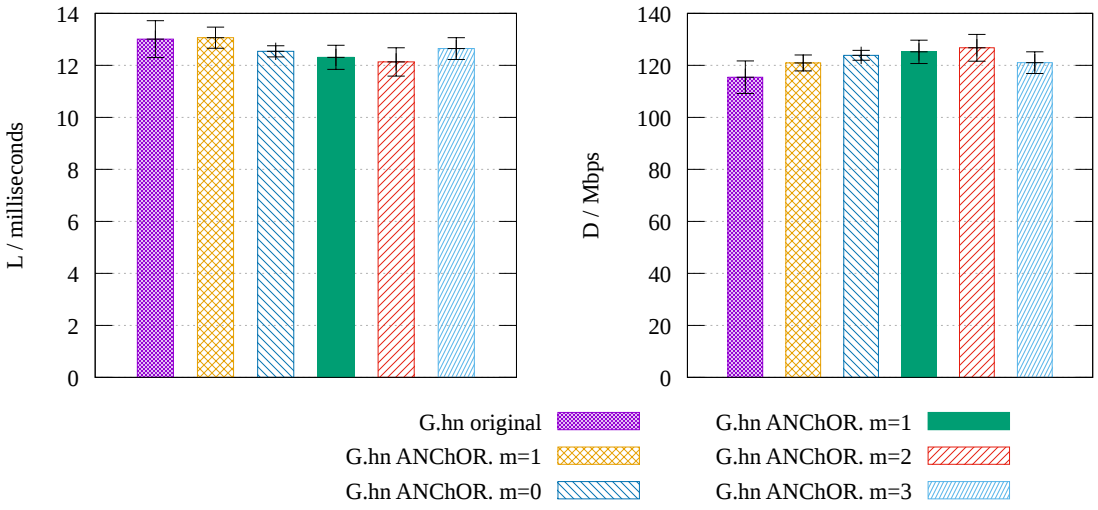


Figure 5.25: In-home scenario. $G_s = 32$, $n = 12$

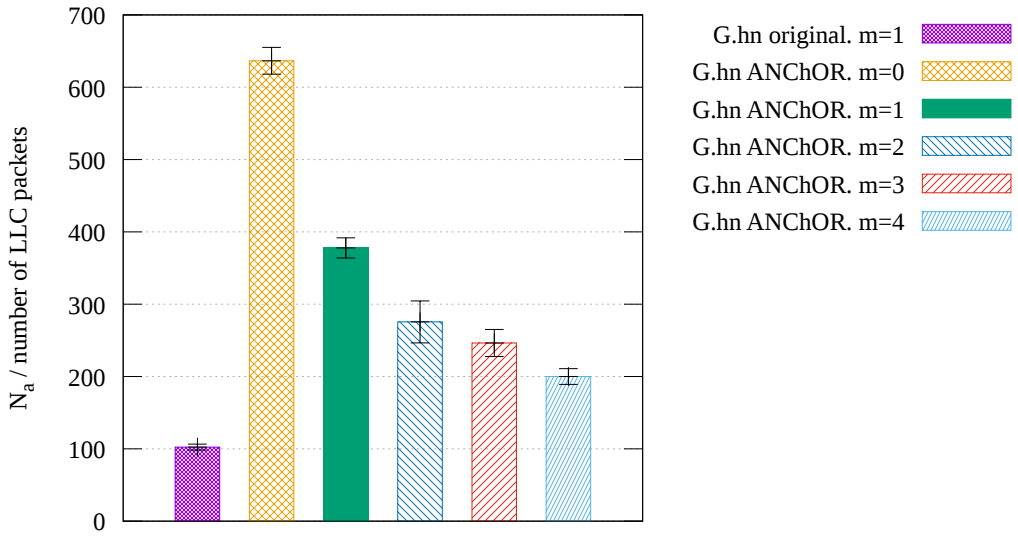


Figure 5.26: Influence of m on N_a

slightly lower (for $m \geq 1$). In the scenario with no relays, D is higher for the G.hn with ANChOR because of the greater N_a (see fig. 5.26). As we mentioned before, the coding on the LLC allows reducing the head-of-the-line problem. In addition, the ANChOR adds redundant packets. Both factors yield much N_a. Nevertheless, the difference in D is not big. The reason lies in the substantial p_l (as shown with analog setup in fig. 5.13 (points for v_s with no cooperation)).

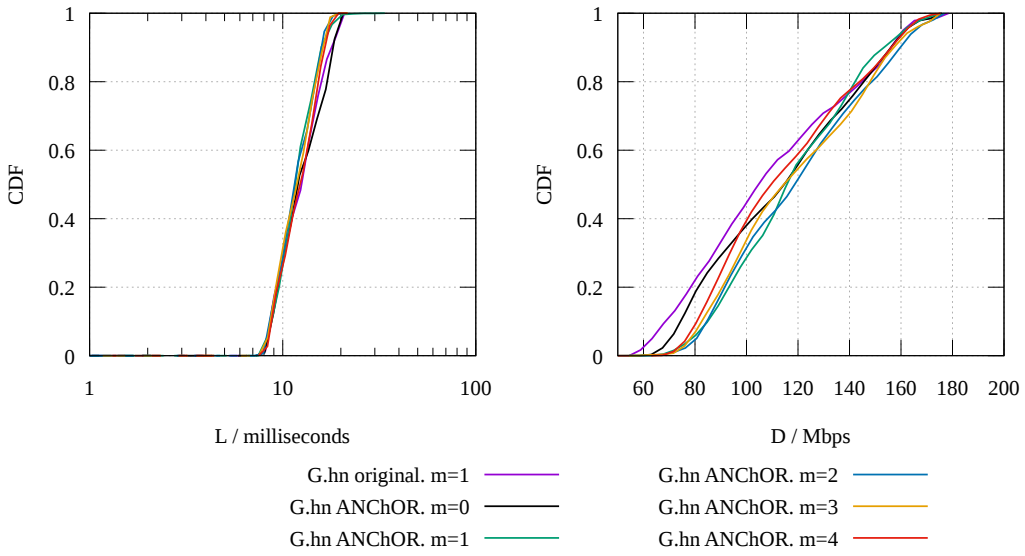


Figure 5.27: In-home scenario. $G_s = 32$, $n = 12$

From fig. 5.26 we observe that N_a decreases when m is growing. It is

explained with the application of the filtering rule of BRR-E. The greater m results in the greater N_c. And, this yields the smaller filtering coefficients for each member of the cooperation group. As far as the amount of the relayed information is proportional to the filtering coefficient, N_a decreases. Nevertheless, D increases. It is possible only when the relay can successively cooperate. Although N_a gets smaller, with greater m, the source has more choices to select the good candidate for the cooperation.

Fig. 5.27 depicts the CDFs of the simulated raw data that was used for fig. 5.25. The gain in D for different m is higher with the bad channel conditions. This result agrees with the theoretical study of ORP potential gain over CSPR in chapter 2.

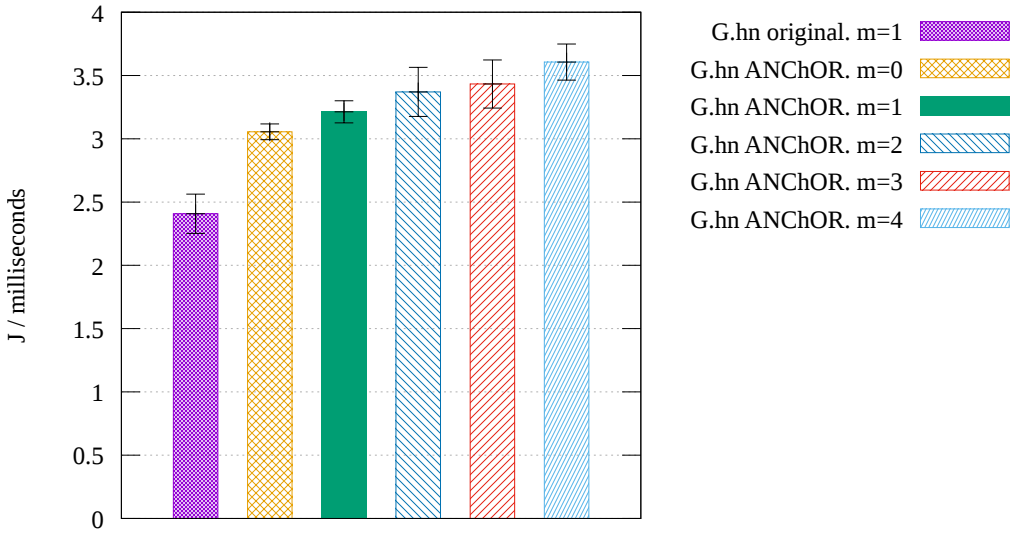


Figure 5.28: In-home scenario. $G_s = 32$, $n = 12$

J with the G.hn with ANChOR (fig. 5.28) is approx. two times greater than J with the original G.hn. In these simulations, we used the on-the-fly NC to decrease J. Nevertheless, the increase of J for the block code is inevitable. J grows for bigger G_s. Anyway, the observed J is small for both the original and the extended G.hn version.

Chapter 6

Conclusions

We developed the Advanced Network Coding based Opportunistic Routing protocol (ANChOR) that can be used as a part of any protocol operating in a broadcast channel (both wireless and PLC). The communication devices can run ANChOR if they support either promiscuous (overhearing) or multicast mode (not as sequential unicast) that enables cooperation. At the moment, available PLC implementations do not fulfill this requirement. We proposed a set of protocol extensions required for to support ANChOR. Some of them require changes to the PLC standards.

The performance of ANChOR is analyzed analytically and by means of simulations with a Simple Network Simulator (SNS) and the detailed simulator ns-3.

We compare the simulated throughput with the channel capacity on the interface between LLC and convergence layers that we model as Packet Erasure Channel (PEC) non-degraded broadcast TDM channel. It shows room for possible improvements. Also, we evaluate the theoretical limit of the achievable gain of ANChOR over Common Single-path Routing Protocols (CSRPs). The highest gain is achieved when the paths for data propagation have approximately the same quality and Packet Erasure Ratio (PER) values are high. Both the channel capacity and the gain can be evaluated with the SNS.

We designed SNS for testing of basic ANChOR functions. It is a discrete-event simulator using PEC non-degraded broadcast Time Division Multiplexing (TDM) channel. The packet losses are modeled by a Bernoulli process. Despite its simplicity, it allows modeling the protocols that do not group sev-

eral Protocol Data Units (PDUs) on the Logical Link Control layer (LLC) into a single PDU on the Multiple Access Control layer (MAC). In PLC, these are NB-PLC protocols like G3, PRIME, IEEE1901.2, etc.

With the help of SNS we investigate such ANChOR parts as extended Basic Routing Rules (BRR-E), Automatic ReQuest replay (ARQ), network initialization, etc. In our research, we started with the design of BRR, which is the predecessor of BRR-E. With an analytical and a simulative approach, we proved that the achievable data rate with BRR reaches the upper bound on the data rate in the considered channel if certain assumptions fulfill. Among them are the channel stationarity and the zero cost of the overhead. These assumptions are not realistic. Therefore, we extended BRR to overcome them resulting in BRR-E.

In BRR-E, the channel stationarity is a helpful but not necessary requirement. The adaptation to the changing channel conditions is realized with the decentralized self-learning mechanism of routing metrics. ANChOR constantly monitors the channel conditions and updates the routing decisions basing on the last observations. The more varying the channel is the more last observations have to be encountered. In order to further enhance the stability of routing decisions, we implement a hysteresis for the comparison operator of the basic routing metric, the vertex priority. Any routing protocol uses a special metric describing the link quality. ANChOR uses the priority as such a metric. It evaluates the effective data rate between the given vertex and the destination. We propose an algorithm for the priority evaluation that considers the data propagation simultaneously on several routes and the correlation of packet losses on all outgoing edges of the given node. It is decentralized and requires the information from the neighbor vertices only. These can be periodically piggybacked to the data packets. We also showed that the priority value equals the upper bound on achievable data rate between the given vertex and the destination. Thus, this is the optimal link quality metric for ORP that targets the throughput maximization. The data rate equal to the source vertex priority can be achieved if all network vertices follow a certain Time Division Multiple Access Plan (TDMAP). In PLC, the channel access is often realized with TDM. Therefore, we added this property to the investigated channel. As a result, a certain source vertex priority corresponds to the certain TDMAP.

It can be obtained by solving the optimization task constructed with the help of min-cut max-flow theorem and using linear programming. We also provide an algorithm allowing to obtain the same solution without the necessity to solve the optimization task. This algorithm can be used for obtaining the theoretically optimal scheduling plan for TDMA. It allows achieving the data rate equal to the vertex priority if the source communicates an infinitely long message and each vertex knows, which part of the received message should be relayed. In reality, such assumptions are not feasible. Therefore, the optimal scheduling is not possible. Instead, we develop a scheduling mechanism based on CSMA that allows achieving the near optimum performance. With this mechanism, each vertex decides on its own, how much of the received information should be forwarded. And, it contends for the channel access using CSMA/CA or CSMA/CD whenever it has any outstanding data. The selection of data amount for relaying is realized by each vertex autonomously with the ANChOR filtering function. It does not require an extensive feedback like ExOR or CCACK. The excessive feedback is considered as one of the main problems of current ORPs. With the help of progressive header and feedback reduction mechanisms, we managed to approach the near optimal performance without unrealistic assumptions. The filtering function is the core part of this mechanism. Our implementation of this function became possible thanking the usage of NC. The key point lies in that each vertex does not have to know *which packets* have to be relayed. When coding data on LLC, it is sufficient to guess *the amount of packets* that have to be forwarded for the particular generation. Analog filtering function can be implemented also without coding on LLC. But the prediction error is much smaller if the coding on LLC is applied. When the generation size grows to infinity, the prediction error approaches zero.

In parallel to the design of BRR-E, we analyze the suitable ARQ mechanisms. ANChOR guarantees the full data recovery on the destination. For this purpose, it performs retransmissions. The developed ARQ operates similarly to the Selective-Repeat ARQ. In difference to the traditional implementation, it handles not the single packets but the generations as the ACK item. It also belongs to the class of the incremental redundancy Hybrid ARQs (HARQs). Combined with the Selective-Repeat ARQ it allows transmitting data of sev-

eral generations before the oldest one is completely received by the destination. Changing the number of active generations, it is possible to adapt the protocol performance to the traffic requirements. The greater the number of active generations is allowed, the higher is the achievable data rate. But the latency grows as well. For the transfer of a HDTV stream, the number of generations should be high, while for web searching it should be small. Fortunately, the LLC allows creating several data flows, each one with its own ARQ setup. Thus, for example, the HDTV and web searching traffic can be fully separated and handled on LLC independently. In addition, Quality of Service (QoS) can be further considered in the MAC scheduler. This topic is not studied in the thesis.

In protocols with CSPRs, ARQ operates for Pt2Pt connections. With ORP, the sender adjusts its transmission window depending on the end of the ACK window on several vertices. Each member of the sender's cooperation group can block the moving of the ARQ window at the sender. Thus, the congestion control for ARQ in protocol with ORP is more demanding. In TCP, although having only one sink vertex there is a number of mechanisms for data recovery in case of a lossy channel like fast retransmit and various congestion control algorithms. Thanks to the coding on LLC, ANChOR does not try to repair all the packet losses. Instead, it uses the losses as an advantage that provides the channel diversity. Each ANChOR vertex does not target to deliver all information it has to each member of its cooperation group. Instead, it only takes care that the group has sufficient information to decode the sent message. If any cooperation group member blocks the sender specifying a relatively old Generation IDentifier (GID) as its ACK window end, the sender applies overshooting of the oldest generation(s). It sends extra redundant data till the ACK window can move forward. The vertices in the cooperation group use the cumulative feedbacks that aggregate the ACK info of all their own cooperation groups. In this way, the single feedback can acknowledge the information that is distributed in the group of nodes.

The contents of the feedback message in ANChOR is different to a set of Serial Sequence Numbers (SSNs). We showed that using SSNs is ineffective when using NC in multi-hop meshed networks. We also analyzed other known approaches and proposed the min-max estimator of useful information.

Comparing these approaches with SNS, we found that our solution gives the best tradeoff between the overhead and the accuracy of the coder information description.

Another problem of ARQ with ORP lies in the selection of the RR forwarder. This question does appear in ARQs with CSPR. We provided the set of conditions when the vertex is allowed forwarding RRM. One of them lies the specification of the RRM forwarder by the originator of RRM. The best strategy for the selection of the RRM forwarder depends on topology. The overall best results we achieved with the selection by the vertex priority.

Setting up ANChOR for usage with the given communication devices, it is necessary to tune such coding parameters as the field, CS and generation size. We used a 2^4 field size as a good compromise between Linear Dependency Ratio (LDR) and header size. Coded symbol (SC) size is platform dependent. In our implementation, it equals the size of FEC block on PHY (540 bytes). It was convenient since the decoding failure of one FEC block causes the loss of exactly one CS. Though, bigger CS is recommended. Then, the proportion of the coding header to the payload size decreases. But, in the same time, the probability of CS loss increases. Thus, a proper compromise has to be found. The last coding parameter, the generation size, also creates the tradeoff between Linear Dependence Ratio (LDR) and header size. For the considered scenarios with Gigabit Home Network (G.hn), a value of 32 packets was selected. The best generation size value depends on various parameters including the network topology because it also influences the prediction accuracy of the ANChOR filtering function. Thus, we select the simulative approach for obtaining the optimal generation size.

The protocol simulation with G.hn DLL and PHY layers was performed using the ns-3 discrete-event simulator. We used a simplified model of packet loss process because of the realistic modeling in BB-PLC requires a great effort. We showed that it is insufficient to model the channel transfer function and the colored noise only. One should model also the impulsive noise types. We implemented the synchronous impulsive noise as the strongest impulsive noise component. Nevertheless, the abstract information rate based model of PHY appears to be insufficient. The complete PHY frontend modeling is required. Otherwise, it is impossible to obtain a realistic packet loss pattern (random-

ness and burstiness). Alternatively, it can be obtained by real measurements.

With the ns-3 model, we first analyzed the scenarios where the application of ANChOR makes sense. In the analytical study, it was shown that the gain of ORPs over CSPRs is always positive. This conclusion holds if no protocol overhead is considered. The simulation results prove that it is possible to obtain a positive gain in most scenarios for a certain set of coding parameters and ANChOR setup. It is even possible to achieve the gain in both the data rate (up to 40%) and in the latency (up to 60%). The highest gain has been observed in the most harsh environments. Such gains become possible thanks to the cooperation and the reduction of the head-of-the-line problem. By generating multiple random topologies we found that the cooperation is more probable if the number of vertices increases. Still, the maximum size of the cooperation group even in the in-home scenario is limited by two (in most cases). ANChOR considers all vertices as potential cooperation candidates. But the number of the eventually selected candidates depends on the contribution of each of them. It is evaluated as the increase in the sender priority when adding more nodes to the cooperation group. With no limitation on the cooperation group size, some nodes send more redundant than useful information. Nevertheless, it makes sense to install more communication device. Then, it is easier to select "the better" cooperation candidates.

Appendix A

Proof of the neccessity of the exclusion rule

Notation

The random variables are denoted by capital letters and their realizations by the respective lower case letters. The cardinality of a vector Z is denoted by $|Z|$. The notation for the random vector $(X_1, ..., X_n)$ is shortened with X^n. Lower case letters $(x_1, ..., x_n)$ are used to show a realization of vector $(X_1, ..., X_n)$ and shortened by x^n.

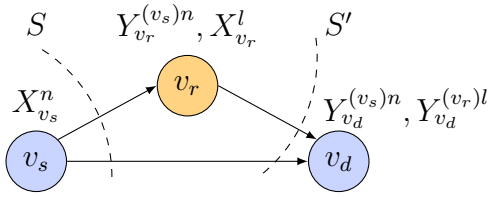


Figure A.1: 1-hop relay network with a direct path

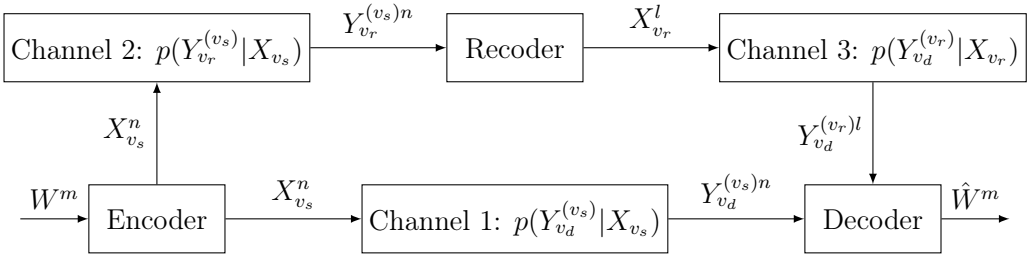


Figure A.2: Point-to-Point transmission model

Model

Let us consider the one-hop relay network with a direct path as shown in fig. A.1. The vertex v_s contains a discrete memoryless source. Let v_s convey the message block S^m with size m to the destination v_d (fig. A.2). Each message element $S_i,\ \forall i \in [0, m)$, is a scalar in the finite field $GF(q)$ with size q. S^m is encoded to $X_{v_s}^n$ before being sent to the channel. The relay vertex v_r overhears the transmission. It receives $Y_{v_r}^{(v_s)n}$ and relays only the side information, i.e. the information that is not received by v_d from v_s directly. The vertex v_r encodes $Y_{v_r}^{(v_s)n}$ to $X_{v_r}^l$. The destination observes on its channel output with v_s the code sequence $Y_{v_d}^{(v_s)n}$ and on the channel output with v_r the code sequence $Y_{v_d}^{(v_r)l}$ (see fig. A.2). We denote the estimation of the decoded message as $\hat{S}^m$. The relation between the channel in- and outputs is defined by CTFs as shown in fig. A.2. We use the channel model as defined in section 2.2.

In some cases, the relay cooperation does not increase the effective data rate between v_s and v_d. This statement is proved with the following theorem.

Theorem 1. The highest achievable rate R so that for any $\mu > 0$ with sufficiently large length of code sequence n: $P[\hat{S}^m \neq S^m] < \mu$, cannot be increased with a cooperation of the relays if $I(X_{v_r}; Y_{v_d}^{(v_r)}) \leq I(X_{v_s}; Y_{v_d}^{(v_s)})$.

Proof. For notation simplicity let $X_{v_s} = X^{(1)}$, $X_{v_r} = X^{(2)}$, $Y_{v_r}^{(v_s)} = Y^{(1)}$, $Y_{v_d}^{(v_s)} = Y^{(2)}$ and $Y_{v_d}^{(v_r)} = Y^{(3)}$ (fig. A.1). We evaluate the network capacity between v_s and v_d as a max-flow of a min-cut [54, Theorem 1]:

$$R \leq I(C) = \sup_{\alpha, 0 \leq \alpha \leq 1} \min\left(I(C_1), I(C_2)\right), \tag{A.1}$$

where the variable α is defined as follows. Let v_d needs time T to receive $Y^{(2)} \cup Y^{(3)}$ and decode $\hat{S}^m$. Also let v_s and v_r be in "sending" state for periods $t_s, t_r \leq T$. Then: $\alpha = t_s/(t_s + t_r)$. Correspondingly, the node v_r stays in the "sending" state for $(1 - \alpha)$ fraction of T.

We expand $I(C)$ as follows:

$$\begin{aligned} R \leq I(C) = \sup_{\alpha, 0 \leq \alpha \leq 1} \min\big[&\alpha \cdot I(X^{(1)}; Y^{(1)}, Y^{(2)}), \\ &\alpha \cdot I(X^{(1)}; Y^{(2)}) + (1 - \alpha) \cdot I(X^{(2)}; Y^{(3)})\big]. \end{aligned} \tag{A.2}$$

We observe that both functions $I(C_1, \alpha) = \alpha \cdot I(X^{(1)}; Y^{(1)}, Y^{(2)})$ and $I(C_2, \alpha) = \alpha \cdot I(X^{(1)}; Y^{(2)}) + (1-\alpha) \cdot I(X^{(2)}; Y^{(3)})$ conditioned by $I(X^{(2)}; Y^{(3)}) < I(X^{(1)}; Y^{(2)})$

are increasing on all period $\alpha \in [0,1]$. So, both $I(C_1,\alpha)$ and $I(C_2,\alpha)$ are maximized when $\alpha = 1$. In fact it means that the relay will no transmit at all, which proves the theorem. □

Notice that the equation $I(X_{v_r}; Y_{v_d}^{(v_r)}) \leq I(X_{v_s}; Y_{v_d}^{(v_s)})$ is analog to $p(v_r) \leq p(v_s)$. This motivates the exclusion rule of BRR.

Corrolary 1. The upper bound on the achievable data rate between v_s and v_d is given by the maximum flow on the minimum cut $I(C)$:

$$I(C) = \begin{cases} I(X^{(1)}; Y^{(2)}) & \text{if } I(X^{(2)}; Y^{(3)}) \leq I(X^{(1)}; Y^{(2)}) \\ \frac{I(X^{(1)};Y^{(1)},Y^{(2)}) \cdot I(X^{(2)};Y^{(3)})}{I(X^{(1)};Y^{(1)},Y^{(2)}) - I(X^{(1)};Y^{(2)}) + I(X^{(2)};Y^{(3)})} & \text{otherwise} \end{cases} \tag{A.3}$$

Proof. The equation in the case $I(X^{(2)}; Y^{(3)}) \leq I(X^{(1)}; Y^{(2)})$ follows from the theorem 1. If $I(X^{(2)}; Y^{(3)}) > I(X^{(1)}; Y^{(2)})$ then the maximum of $I(C_1)$ corresponds to $I(C_1) = I(C_2)$. Using eq. A.2.

$$I(C_1)|_{I(C_1)=I(C_2)} = \frac{I(X^{(1)}; Y^{(1)}, Y^{(2)}) \cdot I(X^{(2)}; Y^{(3)})}{I(X^{(1)}; Y^{(1)}, Y^{(2)}) - I(X^{(1)}; Y^{(2)}) + I(X^{(2)}; Y^{(3)})}, \tag{A.4}$$

which confirms the statement in the corrolary. □

Corrolary 2. If packet losses on edges in fig. A.1 can be described with i.i.d. Bernoulli process, then the upper bound on the achievable data rate can be calculated as follows (using corrolary 1):

$$R \leq I(C) = \begin{cases} d(v_0) \cdot (1 - \varepsilon_3) & \text{if } d(v_1) \cdot (1 - \varepsilon_2) < d(v_0) \cdot (1 - \varepsilon_3) \\ \frac{d(v_0) \cdot d(v_1) \cdot (1-\varepsilon_2) \cdot (1-\varepsilon_1 \cdot \varepsilon_3)}{d(v_1) \cdot (1-\varepsilon_2) + d(v_0) \cdot \varepsilon_3 \cdot (1-\varepsilon_1)} & \text{otherwise,} \end{cases} \tag{A.5}$$

where $\varepsilon_1,\ \varepsilon_2,\ \varepsilon_3$ are the expectation values of the packet erasure ratios on edges (v_s, v_r), (v_r, v_d) and (v_s, v_d) correspondingly, and $d(v)$ is the sending data rate of v.

Appendix B

Gain of ORPs to CSRPs

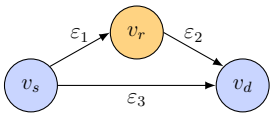


Figure B.1: 1-hop relay network with a direct path and i.i.d Bernoulli loss process on edges

Consider the network topology in fig. A.1. With CSRP, the upper bound achievable data rate can be calculated as (assuming i.i.d Bernoulli loss process and unit sending data rate for both v_s and v_r):

$$R_{CSRP} = \max\left(d(v_0)\cdot(1-\varepsilon_3), \frac{d(v_0)\cdot d(v_1)\cdot(1-\varepsilon_1)\cdot(1-\varepsilon_2)}{d(v_0)\cdot(1-\varepsilon_1)+d(v_1)\cdot(1-\varepsilon_2)}\right). \quad \text{(B.1)}$$

The upper bound on achievable data rate with ORP we provided in eq. A.5. The ORP gain against CSRP we express as follows:

$$\alpha = (R_{ORP} - R_{CSRP})/R_{CSRP}. \quad \text{(B.2)}$$

It has three cases. First, both ORP and CSRP use the direct path only. Thus, the gain is zero. Second, ORP uses both paths and CSRP is still using the direct path. Third, ORP uses both paths and CSRP selects the path with the relay.

$$\alpha = \begin{cases} 0 & \text{if } d(v_1)\cdot(1-\varepsilon_2) < d(v_0)\cdot(1-\varepsilon_3) \\ \frac{\varepsilon_3\cdot(1-\varepsilon_1)\cdot[d(v_1)\cdot(1-\varepsilon_2)-d(v_0)\cdot(1-\varepsilon_3)]}{(1-\varepsilon_3)\cdot[d(v_1)\cdot(1-\varepsilon_2)+d(v_0)\cdot\varepsilon_3\cdot(1-\varepsilon_1)]} & \text{if } X \\ \frac{(1-\varepsilon_3)\cdot[d(v_1)\cdot\varepsilon_1\cdot(1-\varepsilon_2)+d(v_0)\cdot(1-\varepsilon_1)]}{(1-\varepsilon_1)\cdot[d(v_1)\cdot(1-\varepsilon_2)+d(v_0)\cdot\varepsilon_3\cdot(1-\varepsilon_1)]} & \text{otherwise,} \end{cases} \quad \text{(B.3)}$$

where $X = \left[\frac{d(v_0)\cdot d(v_1)\cdot(1-\varepsilon_1)\cdot(1-\varepsilon_2)}{d(v_0)\cdot(1-\varepsilon_1)+d(v_1)\cdot(1-\varepsilon_2)} \leq d(v_0)\cdot(1-\varepsilon_3) \leq d(v_1)\cdot(1-\varepsilon_2)\right]$.

In each case, $\alpha \geq 0$, i.e. ORP outperforms CSRP for any combinations of loss ratios unless all ratios equal zero or $d(v_1)\cdot(1-\varepsilon_2) < d(v_0)\cdot(1-\varepsilon_3)$.

Appendix C

Broadcasting rule

Consider the network topology in fig. B.1. Let the source v_s aim to deliver the symbols of one generation with size G_s to the destination v_d. In accordance to the redundancy rule in BRR-E, the source should send on average $n_s = G_s/(1-\varepsilon_1 \cdot \varepsilon_3)$ symbols. If $G_s \to \infty$, after sending n_s symbols, the expectation value of total DOF on the relay v_r and v_d equals G_s, $\vartheta(\{v_r, v_d\}) = G_s$. The filtering coefficient of the relay v_r in BRR-E $p_f = \varepsilon_3$. The relay gets on average $m_r = n_s \cdot (1-\varepsilon_1)$ coded symbols and sends $n_r = m_r \cdot p_f/(1-\varepsilon_2) = G_s \cdot (1-\varepsilon_1) \cdot \varepsilon_3/(1-\varepsilon_1 \cdot \varepsilon_3)/(1-\varepsilon_2)$ recoded symbols. If $G_s \to \infty$, n_r issues the upper bound on the number of recoded symbols to be sent by v_r. But BRR-E does not give any rule on that when v_r should send the recoded symbols. It can broadcast anytime when $\rho(v)$. We show that there is no need in such rule for the optimal performance with the following lemma.

Lemma 1. The total number n_t of sent symbols by both v_s and v_r before $\rho(v_d) = G_s$ does not change if v_r starts sending the recoded symbols at any time in interval $[t_1, t_2]$ where t_1 corresponds to the first time when $\rho(v_r) > 0$ and t_2 to the moment when $\vartheta(\{v_r, v_d\}) = G_s$.

Proof. Let us divide n_t in three parts. First, when the symbols sent by v_s before v_r starts helping (r). Second, the symbols sent by both v_s and v_r when both v_s and v_r are active $(2 \cdot s)$. Third, the symbols sent by v_r when v_s stopped sending (m). Let also both v_r and v_s stop sending at latest when $\vartheta(\{v_r, v_d\}) = G_s$. Then, the following equation is valid:

$$G_s = k \cdot (1-\epsilon_3) + j \cdot (1-\epsilon_2),$$

where $k = r + s$ and $j = s + m$ ($n_t = k + j$). Note that $k = n_s$ for any possible combinations of r and s. Then, the value j allows reaching $\vartheta(\{v_r, v_d\}) = G_s$ with the following condition:

$$j \geq \frac{G_s \cdot \epsilon_3 \cdot (1 - \epsilon_1)}{(1 - \epsilon_1 \cdot \epsilon_3) \cdot (1 - \epsilon_2)}.$$

Using minimal value of j, we see that n_t remains constant for all possible combinations of s and m. So, the lemma is proved. □

Since n_t does not change, one needs the same number of channel uses to deliver the complete generation to v_d for different helping start time of v_r.

Appendix D

Proof of optimality of BRR for triangular topology

Theorem 2. BRR-E creates a complete set of routing rules to reach the optimal performance for the topology in fig. B.1.

Proof. Let the source v_s aim to deliver the symbols of one generation with size G_s to the destination v_d. In accordance to appendix C, v_s sends on average $n_s = G_s/(1-\varepsilon_1 \cdot \varepsilon_3)$ and v_r $n_r = G_s \cdot (1-\varepsilon_1) \cdot \varepsilon_3/(1-\varepsilon_1 \cdot \varepsilon_3)/(1-\varepsilon_2)$ symbols (using the redundancy and the filtering rules of BRR-E). Thus, both v_s and v_r need in total on average $n_s + n_r$ channel uses to deliver G_s original symbols to v_d. So, the effective data rate between v_s and v_d can be calculated as follows ($\varepsilon_2 \leq \varepsilon_3$):

$$\begin{aligned} d = \frac{G_s}{n_s + n_r} &= \frac{G_s}{\frac{G_s}{1-\varepsilon_1 \cdot \varepsilon_3} + \frac{G_s \cdot (1-\varepsilon_1) \cdot \varepsilon_3}{(1-\varepsilon_1 \cdot \varepsilon_3) \cdot (1-\varepsilon_2)}} \\ &= \frac{(1-\varepsilon_2) \cdot (1-\varepsilon_1 \cdot \varepsilon_3)}{1-\varepsilon_2+\varepsilon_3 \cdot (1-\varepsilon_1)}. \end{aligned} \tag{D.1}$$

If $\varepsilon_2 > \varepsilon_3$ then $n_s = G_s/(1-\varepsilon_3)$ and $n_r = 0$ that gives $d = 1-\varepsilon_3$ (using the exclusion rule of BRR-E). Notice that this result is analog to the upper bound on achievable data rate given by eq. B.3, which proves the theorem. □

Appendix E

Reducing the retransmission probability

Let v have the cooperation group $L(v)$ and the packet losses on edges (v,u) $\forall u \in L(v)$ are i.i.d. Bernoulli distributed with expectation values or loss ratios $\varepsilon_{(v,u)}$. Let also denote the binomial distribution cumulative function as follows:

$$\mathcal{B}(s,e,n,p) = \sum_{k=s}^{e} \left(\begin{array}{c} n \\ k \end{array} \right) \cdot (1-p)^k \cdot p^{n-k}.$$

Then, the probability that, after v broadcasts $n \geq \rho(v)$ coded symbols, the total DOF of the cooperation group $\vartheta(L(v))$ ($K = \vartheta(L(v))$) is less then the unique DOF on v can be evaluated as follows:

$$P[K < \rho(v)] = \mathcal{B}(0, \rho(v)-1, n, p), \tag{E.1}$$

where $p = \prod_{u \in L(v)} \varepsilon_{(v,u)}$. $P[K < \rho(v)]$ can be also called the retransmission probability. It can be easily observed that for $n = \rho(v)/(1-p)$ the probability $P[K < \rho(v)] \approx 0.5$, which is independent of p. If $L(v)$ receives less symbols than v plans to deliver, it may cause the retransmissions. Adding the redundancy equal to the expectation amount of lost symbols, the retransmissions can be required in 50% cases. But intuitively, for small p, adding just a few additional redundant symbols can substantially reduce $P[K < \rho(v)]$. We can easily prove it evaluating $P[K < \rho(v)]$ with eq. E.1 for $n = \rho(v)/p + \Delta$, where $\Delta > 0$. On praxis, one can aim certain $P[K < \rho(v)]$ and the corresponding n has to be found. It is possible to find n for certain $P[K < \rho(v)] = P$ applying eq. E.1 iteratively with $n > \rho(v)/p$ till $P[K < \rho(v)] \leq P$. We calculate n

in a more efficient way using the following idea. Note that for $n > 20$ and p locating not near 0 and 1, the normal distribution approximates the binomial distribution (Moivre-Laplace theorem). We aim to send so many symbols n that $\rho(v) = m - a \cdot \sigma$, where $m = n \cdot (1-p)$ and $\sigma = \sqrt{n \cdot p \cdot (1-p)}$ are the expectation value and the standard deviation of the number of the received symbols by vertices in $L(v)$. Then, $P[K < \rho(v)]$ for $a = 1, 2, 3$ can be calculated as $\{\Phi(1), \Phi(2), \Phi(3)\} = \{0.8413, 0.9773, 0.9987\}$, where $\Phi(x)$ is the probability of the normal distribution with zero mean and standard deviation equal one. Thus, we can select the aim retransmission probability from the set $\{0.5, 0.8413, 0.9773, 0.9987\}$ and obtain the corresponding n. Remember that for $P[K < \rho(v)] = 0.5$: $n = \rho(v)/(1-p)$. For other values of retransmission probability we derive the equation.

The aim value n we derive from:

$$\rho(v) = n \cdot (1-p) - a \cdot \sqrt{n \cdot p \cdot (1-p)}. \tag{E.2}$$

Let $n = b^2$. Then, we solve $b^2 \cdot (1-p) - b \cdot a \cdot \sqrt{p \cdot (1-p)} - \rho(v) = 0$:

$$n = (1/2 \cdot [\sqrt{d + 4 \cdot \rho(v)/(1-p)} - \sqrt{d}])^2, \tag{E.3}$$

where $d = a^2 \cdot p/(1-p)$.

The coding rate $c(a)$ required to achive certain retransmission probability equals:

$$c(a) = \rho(v)/n = \frac{\rho(v)}{(1/2 \cdot [\sqrt{d + 4 \cdot \rho(v)/(1-p)} - \sqrt{d}])^2}. \tag{E.4}$$

Note that for $a = 0$, $c(a) = 1 - p$.

Appendix F

Calculation of Expected Average number of transmissions (EAX) for topologies with bi-directional links

We check the suitability of the algorithm for EAX calculation in [64] for the topology in fig. 1.1b. Here, the link between v_1 and v_2 is bi-directional. Using notations from [64], we denote $EAX(v_i, v_d)$ as D_i and the cooperation group $L(v_i)$ as F_i. The vertex set S will contain the vertices, for which D_i and F_i are already calculated ($Q = \overline{S}$). Now, we apply the algorithm in [64] for EAX calculation.

Initialization:

$$D_0 = D_1 = D_2 = \infty,\ D_3 = 0$$

$$F_0 = F_1 = F_2 = F_3 = \emptyset,\ S = \emptyset,\ Q = V$$

The minimum $D_i,\ v_i \in Q$, is D_3:

$$j = 3$$

$$S = \{v_3\},\ Q = \{v_0, v_1, v_2\}$$

Vertex v_3 has following incoming edges $\{(v_1, v_3), (v_2, v_3)\}$. For the soruce vertex of the first edge, let us calculate D_1 and F_1:

$$
\begin{aligned}
&i = 1 \\
&J = F_1 \cup \{v_3\} = \{v_3\} \\
&D_1 > D_3 \rightarrow true \\
&Then,\ D_1 = d_{1J} + D_J = 1/(1 - \varepsilon_{13}) + 0 \\
&F_1 = J = \{v_3\}
\end{aligned}
$$

For the source vertex of the second edge, let us calculate D_2 and F_2:

$$
\begin{aligned}
&i = 2 \\
&J = F_2 \cup \{v_3\} \\
&D_2 > D_3 \rightarrow true \\
&Then,\ D_2 = d_{2J} + D_J = 1/(1 - \varepsilon_{23}) + 0 \\
&F_2 = J = \{v_3\}
\end{aligned}
$$

Now, the minimum $D_i,\ v_i \in Q$, depends on relation between ε_{13} and ε_{23}. Assume, $\varepsilon_{13} > \varepsilon_{23}$ then the minimum is D_2:

$$
\begin{aligned}
&j = 2 \\
&S = \{v_2, v_3\},\ Q = \{v_0, v_1\}
\end{aligned}
$$

Vertex v_2 has the following incoming edges $\{(v_0, v_2), (v_1, v_2)\}$. We do not shown the calculation of D_0 and F_0 because it does not affect our conclusion. We

calculate D_1 and F_1:

$$
\begin{aligned}
& i = 1 \\
& J = F_1 \cup \{v_2\} = \{v_2, v_3\} \\
& D_1 > D_2 \rightarrow true \\
& Then,\ D_1 = d_{1J} + D_J \\
& d_{1J} = 1/(1 - \varepsilon_{13} \cdot \varepsilon_{12}) \\
& D_J = w_2 \cdot D_2 + w_3 \cdot D_3 \\
& w_2 = \varepsilon_{13} \cdot (1 - \varepsilon_{12})/(1 - \varepsilon_{13} \cdot \varepsilon_{12}) \\
& D_2 = 1/(1 - \varepsilon_{23}) \\
& w_3 = (1 - \varepsilon_{13})/(1 - \varepsilon_{13} \cdot \varepsilon_{12}) \\
& D_3 = 0 \\
& D_J = \varepsilon_{13} \cdot (1 - \varepsilon_{12})/(1 - \varepsilon_{13} \cdot \varepsilon_{12})/(1 - \varepsilon_{23}) \\
& D_1 = [(1 - \varepsilon_{23}) + \varepsilon_{13} \cdot (1 - \varepsilon_{12})]/(1 - \varepsilon_{13} \cdot \varepsilon_{12})/(1 - \varepsilon_{23}) \\
& F_1 = J = \{v_2, v_3\}
\end{aligned}
$$

Now, the minimum $D_i,\ v_i \in Q$, is D_1:

$$
\begin{aligned}
& j = 1 \\
& S = \{v_1, v_2, v_3\},\ Q = \{v_0\}
\end{aligned}
$$

$$
\begin{aligned}
& i = 2 \\
& J = F_2 \cup \{v_1\} = \{v_1, v_3\} \\
& D_2 > D_1 \rightarrow true
\end{aligned}
$$

If D_2 can be greater than D_1 then the vertex v_1 includes v_2 in its cooperation group F_1. But the vertex v_2 have already included v_1 in its cooperation group F_2. Therefore, if $D_2 > D_1$ is possible, the algorithm in [64] creates uncertainty, i.e. v_1 relays data from v_2 and v_2 relays data from v_1 creating an endless cycle.

We check if $D_2 > D_1$ is possible:

$$
\begin{aligned}
& D_2 > D_1 \\
& 1/(1-\varepsilon_{23}) > [(1-\varepsilon_{23}) + \varepsilon_{13} \cdot (1-\varepsilon_{12})]/(1-\varepsilon_{13} \cdot \varepsilon_{12})/(1-\varepsilon_{23}) \\
& 1 > [(1-\varepsilon_{23}) + \varepsilon_{13} \cdot (1-\varepsilon_{12})]/(1-\varepsilon_{13} \cdot \varepsilon_{12}) \\
& 1-\varepsilon_{13} \cdot \varepsilon_{12} > 1-\varepsilon_{23}+\varepsilon_{13}-\varepsilon_{13} \cdot \varepsilon_{12}) \\
& \varepsilon_{23} > \varepsilon_{13}
\end{aligned}
$$

So, $D_2 > D_1$ is possible only if $\varepsilon_{23} > \varepsilon_{13}$. But we have previously assumed that $\varepsilon_{23} < \varepsilon_{13}$. Thus, $D_2 > D_1$ is not possible. Therefore, the algorithm in [64] should not create uncertainty for topologies with bi-directional edges.

Appendix G

Feedback overhead of full coding matrices

We estimate the amount of protocol overhead that can be created with a RR message if it contains the complete set of coding vectors. The example is based on the G.hn protocol [6].

Let N be the number of generations in the LLC buffer, K the average number of codewords in each MAC PDU with size P_s bits per codeword, G_s the generation size and $GF(q)$ the Galois finite field with size q. Then, the size of all coding vectors in the buffer is $S = N \cdot G_s^2 \cdot q$ bits. The piggybacked feedback creates the overhead of θ:

$$\theta = S/(K \cdot P_s).$$

The relation between K and $N \cdot G_s$ depends on the sizes of the cooperation groups, communication link reliability and ARQ. With per-hop ACK, $K/(N \cdot G_s)$ is greater than with end-to-end ACK because the data can be faster acknowledged. In G.hn the per-hop ACK is used. Thus, $(N \cdot G_s) \leq 2 \cdot K$. Assuming that the sent codewords are evenly distributed between the vertices of each cooperation group and the size of the cooperation group is n, the size of each MAC PDU equals $K = K_{max}/n$.

Let $K_{max} = 376$ [6], $P_s = 540 \cdot 8 = 4320$ [6], $G_s = 20$, $q = 8$ and $n = 2$. Then, $K = 376/2 = 188$, $N = 2 \cdot 188/20 = 18.8$, $S = 18.8 \cdot 20^2 \cdot 8 = 1203200$. Thus:

$$\theta = 60160/(188 \cdot 4320) = 7.4\%.$$

Appendix H

Block diagram of G.hn physical layer in ns-3 model

Fig. H.1 shows the block diagram of the PHY layer. The shaded blocks are not implemented. Thus, the message on the input of such blocks appears on the output of them immediately and without changes.

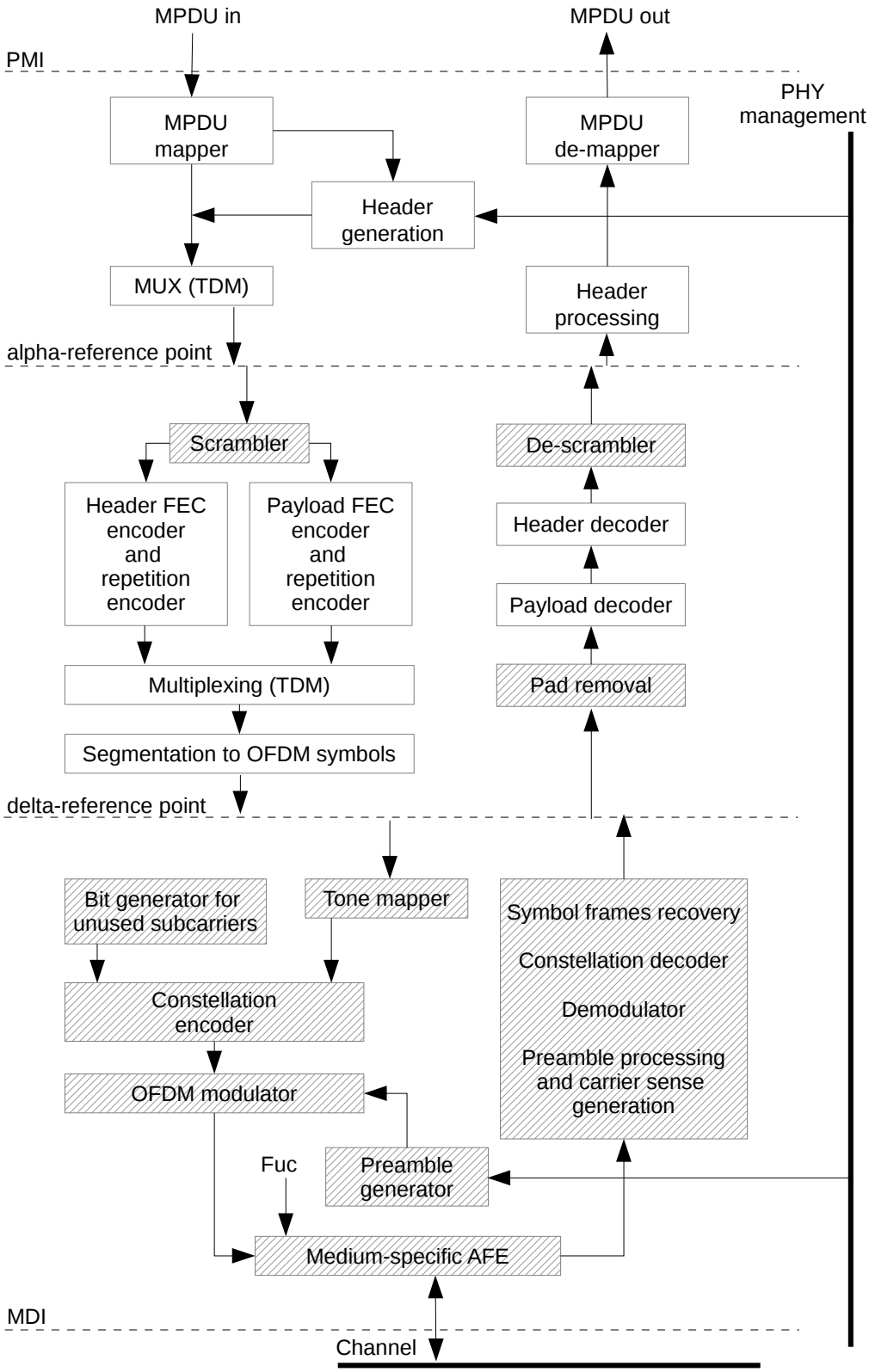


Figure H.1: PHY structure (compare with [4])
Shaded blocks are transparent in the simulator

Appendix I

PER to BER mapping

For a certain value PER ($p^{(m)}$), we aim to determine the best PHY coding rate (c) and the corresponing BER ($p^{(b)}$). Assuming the bit loss process to be Bernoulli distributed, the number of lost bits in a FEC block is described with the binomial distribution. Let k be the length of the FEC block before encoding and n be the length of the encoded block (both k and n are in bits). Then, PER as a function of BER can be expressed as follows (considering ideal MDS code):

$$p^{(m)} = 1 - \mathcal{B}(n-k, n, p^{(b)}) = 1 - \sum_{s=0}^{n-k} C_n^s \cdot (p^{(b)})^s \cdot (1-p^{(b)})^{n-s}. \tag{I.1}$$

Let c_i be the i-th PHY coding coefficient: $c_i \in \mathcal{C} = \{1/4,\ 1/2,\ 2/3,\ 5/6,\ 16/21,\ 16/18,\ 20/21\}$ (G.hn [4]). For a given value $p^{(m)}$, we need to calculate the corresponding tuple of $c \in \mathcal{C}$ and $p^{(b)}$ that maximizes the rate of FEC blocks without bit errors. This data rate can be calculated as $R = \max_{c_i}(B(p_i^{(b)}) \cdot c_i \cdot (1 - p_i^{(m)}))$, where $B(p_i^{(b)})$ is the amount of bits per OFDM symbol calculated with BAA for PER value $p_i^{(m)}$. Thus, the optimal PHY coding rate and the corresponding BER depend on BAA implementation. Since we do not study the gain from the mutual coding on PHY and LLC layers, we set the PHY coding rate to be constant $c_i = 20/21$. In this case, the values n and k in eq. I.1 are constant (for a FEC block size of 540 bytes in G.hn $n = 540 \cdot 8 \cdot 21/20$, $k = 540 \cdot 8$). Thus, $p^{(m)}$ is the function of $p^{(b)}$ only. Since this function is hard to invert, we calculate the PER to BER mapping numerically. First, we calculate the value $p'^{(m)}$ for different BER ($p^{(b)}$) with a very small step. Then,

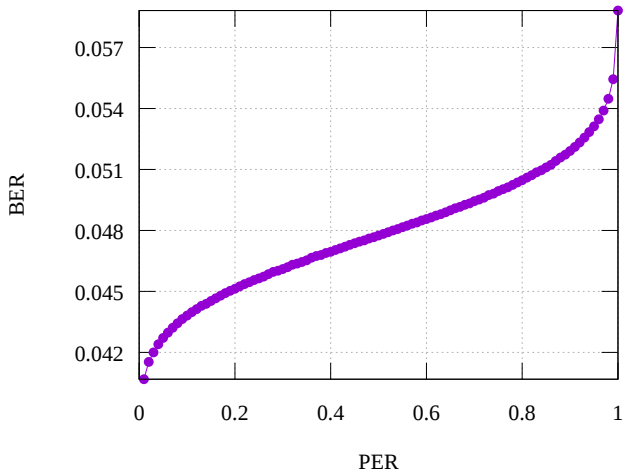


Figure I.1: PER to BER mapping

we filter the tuples $(p'^{(m)}, p^{(b)})$ to leave only those entries, which satisfy:

$$p^{(m)} - 0.0005 < p'^{(m)} < p^{(m)} + 0.005,\ p^{(m)} = \{0.01, 0.02, ..., 1.0\}.$$

The small BER step allows finding the entries with $|p'^{(m)} - p^{(m)}| < 0.01$ for each value $p^{(m)}$. The corresponding table of tuples $(p'^{(m)}, p^{(m)}, c, p^{(b)})$ is saved for usage by ANChOR. The table values are shown in fig. I.1. Thus, specifying certain $p^{(m)}$ we look up in the table for the corresponding c and $p^{(b)}$.

Bibliography

[1] I. Tsokalo, G. Prasad, S. Mudriievskyi, and R. Lehnert. CSMA/CD in PLC: Test with full G.hn and IP/UDP protocol stack. *ISPLC 2017, Madrid, Spain*, 3 - 5 Apr. 2017.

[2] Roee Bar. In-vehicle powerline communication using software-defined radio. *Master thesis. University of British Columbia*, August 2016.

[3] I. Tsokalo, R. Lehnert, and F. H.P. Fitzek. Intraflow network coding on the data link layer for broadband PLC. *ISPLC 2016, Bottrop, Germany*, 20 - 23 March 2016.

[4] ITU-T. G.9960. system architecture and physical layer specification. 06/2010.

[5] Dirk Benyoucef. A new statistical model of the noise power density spectrum for powerline communication. *ISPLC 2003, Kyoto, Japan*, 26-28 March 2003.

[6] ITU-T. G.9961. data link layer specification. 06/2010.

[7] Homeplug. *[Online] www.homeplug.org/*, Last visited: 04.2017.

[8] PLC4SG project. *[Online] www.pt-it.pt-dlr.de/de/2709.php*, Last visited: 02.2017.

[9] DLC+VIT4IP project. *[Online] www.dlc-vit4ip.org/wb/*, Last visited: 02.2017.

[10] Ernst and Young GmbH. Kosten-Nutzen-Analyse für einen flächendeckenden Einsatz intelligenter Zähler. 2013.

[11] I. Tsokalo, R. Bernstein, and R. Lehnert. Investigation of powerline communication modems for energy management systems. *WSPLC2012, Rome, Italy*, September 2012.

[12] I. Tsokalo, R. Lehnert, and E. Mayer. Evaluation of broadband PLC for electricity metering with the field trial. *WSPLC2015, Klagenfurt, Austria*, September 2015.

[13] Iberdrola. *[Online] www.iberdroladistribucion.es/about-us/innovation/projects-in-europe*, Last visited: 04.2017.

[14] PRIME. *[Online] www.prime-alliance.org/?page_id=249*, Last visited: 04.2017.

[15] OPERA. *[Online] www.ipcf.org/company/opera/open-plc-european-research- alliance.html*, Last visited: 04.2017.

[16] IEEE. Standard for broadband over power line networks: Medium access control and physical layer specifications. 12/2010.

[17] HomePlug. Homeplug green phy specification. 09/2012.

[18] Z. Zhong, J. Wang, G-H. Lu, and S. Nelakuditi. On selection of candidates for opportunistic any-path forwarding. *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 10, no. 4 October 2006.

[19] C. Perkins, E. Belding-Royer, , and S. Das. Ad hoc on-demand distance vector (AODV) routing. *Internet Requests for Comments, RFC 3561*, July 2003.

[20] B.A.T.M.A.N. routing protcol. *[Online] www.open-mesh.org/projects/batman-adv/*, Last visited: 02.2017.

[21] I. Tsokalo, B. Matthiesen, and F. H. P. Fitzek. Approaching the limits in routing in power line communication exploiting network coding. *ISPLC 2017, Madrid, Spain*, 3-5 April 2017.

[22] Sanjit Biswas and Robert Morris. Exor: Opportunistic multi-hop routing for wireless networks. *ACM SIGCOMM Computer Communication Review, New York, USA*, 35(4):133–144, October 2005.

[23] Joon-Sang Park, Mario Gerla, Desmond S. Lun, Yunjung Yi, and M. Medard. Codecast: A network-coding-based ad hoc multicast protocol. *IEEE Wireless Communications*, Volume: 13, Issue: 5 October 2006.

[24] M. Naghshvar and T. Javidi. Opportunistic routing with congestion diversity in wireless multi-hop networks. *INFOCOM 2010, San Diego, USA*, 15-19 March 2010.

[25] A. Khreishah, I. Khalil, and Jie Wu. Universal network coding-based opportunistic routing for unicast. *IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 6*, June 2015.

[26] K. Zeng, Z. Yang, and W. Lou. Opportunistic routing in multi-radio multi-channel multi-hop wireless networks. *IEEE Trans.Wireless Commun., vol. 9, no. 11*, November 2010.

[27] M. Lu, P. Steenkiste, and T. Chen. Design, implementation and evaluation of an efficient opportunistic retransmission protocol. *IEEE/ACM Annual Int. Conf.MobiCom, Beijing, China*, 20-25 September 2009.

[28] C. Gkantsidis, Wenjun Hu, P. Key, B. Radunovic, P. Rodriguez, and S. Gheorghiu. Multipath code casting for wireless mesh networks, new york, USA. *ACM CoNEXT*, 10-13 December 2007.

[29] Che-Jung Hsu, Huey-Ing Liu, and Winston Seah. Economy: A duplicate free opportunistic routing. *Proceedings of the 6th International Conference on Mobile Technology, Application and Systems, Nice, France*, 10-13 September 2009.

[30] D. Koutsonikolas, C. Wang, and Y. Hu. Ccack: Efficient network coding based opportunistic routing through cumulative coded acknowledgments. *IEEE Conf. INFOCOM, San Diego, USA*, 15-19 March 2010.

[31] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading structure for randomness in wireless opportunistic routing. *ACM SIGCOMM, Kyoto, Japan*, October 2007.

[32] E. Rozner, J. Seshadri, Y. Mehta, , and L. Qiu. Soar: Simple opportunistic adaptive routing protocol for wireless mesh networks. *IEEE Trans. Mobile Comput., vol. 8, no. 12*, December 2009.

[33] P. Pahlevani, D. E. Lucani, and F. H.P. Fitzek. Adaptive relay activation in the network coding protocols. *European Wireless 2015, Budapest, Hungary*, 20-22 May 2015.

[34] S. Pandi, F. H.P. Fitzek, J. Pihl, M. V. Pedersen, and D. Lucani. Sending policies in dynamic wireless mesh using network coding. *European Wireless 2015, Budapest, Hungary*, 20-22 May 2015.

[35] D. E. Lucani, F. H.P. Fitzek, M. Medard, and M. Stojanovic. Network coding for data dissemination: It is not what you know, but what your neighbors don't know. *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, Seoul, Korea*, 23-27 June 2009.

[36] J. Cloud and M. Medard. Network coding over satcom: Lessons learned. *Cornell University Library*, 19 June 2015.

[37] M. Karzand, D. J. Leith, J. Cloud, and M. Medard. FEC for lower in-order delivery delay in packet networks. *Cornell University Library*, 2 September 2016.

[38] J. Bilbao, P. M. Crespo, I. Armendariz, and M. Medard. Network coding in the link layer for reliable narrowband powerline communications. *IEEE journal on selected areas in communications*, Volume: 34, Issue: 7, 2016.

[39] F. R. Kschischang, D. Katabi, D. Katti, H. Rahul, Chen Fend, Baochun Li, and F. H.P. Fitzek et al. *Book: Network Coding. Fundamentals and Applications*. Elsevier, 2012.

[40] G3-PLC Alliance. Performance test suite for G3-PLC device certification. 11/2014.

[41] IEEE. Standard for low-frequency (less than 500 khz) narrowband power line communications for smart grid applications. 10/2013.

[42] J. A. Cortes, L. Diez, F. J. Canete, and J. J. Sanchez-Martinez. Analysis of the indoor broadband power-line noise scenario. *IEEE Trans. on Electromagnetic Compatibility*, vol. 54, no. 4 Novermber 2010.

[43] L. Lampe and A.J. Han Vinck. On cooperative coding for narrowband plc networks. *AEÜ - International Journal of Electronics and Communications*, vol. 65, no. 8 August 2011.

[44] P. Pahlevani, D. E. Lucani, M. V. Pedersen, and F. H.P. Fitzek. Playncool: Opportunistic network coding for local optimization of routing in wireless mesh networks. *GLOBECOM 2013 Workshop, Antlanta, USA*, 9-13 December 2013.

[45] H. E. Saffar L. Lampe, Md. J. Rahman. Characteristics of power line networks: Diversity and interference alignment. *ISPLC 2017, Madrid, Spain*, 3-5 April 2017.

[46] M. Tomoskozi, F. H.P. Fitzek, D. E. Lucani, M. Pedersen, and P. Seeling. On the delay characteristics for point-to-point links using random linear network coding with on-the-fly coding capabilities. *European wireless 2014, Barcelona, Spain*, 14-16 May 2014.

[47] Heide, Janus; Pedersen, M. Videbaek, and F. H.P. Fitzek. Decoding algorithms for random linear network codes. *Lecture Notes in Computer Science, Aalborg University, Denmark*, 2011.

[48] Manual of the KODO library. *[Online] kodo-docs.steinwurf.com/en/latest/*, Last visited: 02.2017.

[49] M.A. Tunc, E. Perrins, and L. Lampe. Optimal LPTV-aware bit loading in broadband PLC. *IEEE Transactions on Communications*, 07 November 2013.

[50] Thomas M. Cover. Capacity theorems for the relay channel. *IEEE Transactions on Information Theory*, IT-18(1), January 1972.

[51] Edward c. Van Der Meulen. A survey of multi-way channels in information theory: 1961-1976. *IEEE Transactions on Information Theory*, IT-23(1), January 1977.

[52] Thomas M. Cover and Abbas A. El Gamal. Capacity theorems for the relay channel. *IEEE Transactions on Information Theory*, IT-25(5), September 1979.

[53] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Publisher: John Wiley & Sons, Edition: 2, 2006.

[54] Khojastepour, Mohammad Ali, Sabharwal, Ashutosh, Aazhang, and Behnaam A. Bounds on achievable rates for general multi-terminal networks with practical constraints. *Lecture Notes in Computer Science*, April 2003.

[55] R. Khalili and K. Salamatian. On the capacity of erasure relay channel: Multi-relay case. *IEEE ISOC ITW2005 on Coding and Complexity, Rotorua, New Zealand*, 29 Aug.-1 Sept. 2005.

[56] Chih-Chun Wang. On the capacity of 1-to-k broadcast packet erasure channels with channel output feedback. *IEEE Transactions on Information Theory*, vol. 58, no. 2 February 2012.

[57] T. Clausen and P. Jaquet. Optimized link state routing protocol (olsr). *Internet Requests for Comments, RFC 3626*, October 2003.

[58] D. Johnson, Y. Hu, and D. Maltz. The dynamic source routing protocol (dsr) for mobile ad hoc networks for ipv4. *Internet Requests for Comments, RFC 4728*, February 2007.

[59] R. C. Shah, S. Wieth, and A. Wolisz. When does opportunistic routing make sense? *Pervasive Computing and Communications Workshops, Kauai Island, USA*, 8-12 March 2005.

[60] I. Tsokalo and R. Lehnert. Potential of routing with network coding in PLC networks. *10th WSPLC 2016, Paris, France*, 10-11 October 2016.

[61] Coin-CLP. *[Online] https://github.com/coin-or/Clp*, Last visited: 02.2017.

[62] Nessrine Chakchouk. A survey on opportunistic routing inwireless communication networks. *IEEE Communication Surveys & Tutorials*, vol. 17, no. 4 2015.

[63] A. Trivino-Cabrera and S. Canadas-Hurtado. Survey on opportunistic routing in multihop wireless networks. *International Journal of Communication Networks and Information Security*, 3(2), August 2011.

[64] R. Laufer, H. Dubois-Ferriere, and L. Kleinrock. Multirate anypath routing in wireless mesh networks. *IEEE Conf. INFOCOM, Rio de Janeiro, Brazil*, 19-25 April 2009.

[65] SimpleNetSim. *[Online] https://github.com/tsokalo/SimpleNetSim*, Last visited: 02.2017.

[66] ns-3 simulator. manual. *[Online] www.nsnam.org*, Last visited: 02.2017.

[67] Physical and Data Link layer models of G.hn. ns-3 module. *[Online] https://github.com/tsokalo/ghn-plc/tree/coded-llc-flow*, Last visited: 02.2017.

[68] F. Aalamifar, A. Schloegl, D. Harris, and L. Lampe. Modelling power line communication using network simulator-3. *Globecom 2013, Atlanta, USA*, 09 Dec - 13 Dec 2013.

[69] A. M. Tonello and F. Versolatto. Bottom-up statistical PLC channel modeling-part i: Random topology model and efficient transfer function computation. *IEEE Transactions on Power Delivery, vol. 26, no. 2*, April 2011.

[70] In-home topology generator. ns-3 module. *[Online] https://github.com/tsokalo/ topology-generator*, Last visited: 02.2017.

[71] I. Tsokalo and R. Lehnert. Modeling approach of broadband in-home PLC in network simulator 3. *ISPLC 2015, Austin, USA*, 29 March - 1 April 2015.

[72] Model of electrical device with random parameters. ns-3 module. *[Online] https://github.com/tsokalo/electrical-device-model*, Last visited: 02.2017.

[73] P.J. Pinero, J.A. Cortes, J. Malgosa, F.J. Canete, P. Manzanares, and L. Diez. Analysis and improvement of multicast communications in homeplug av-based in-home networks. *Computer Networks journal*, Vol. 62 January 2014.

[74] Nexans, "Power cables 1-30 kV". *[Online] www.nexans.de/ eservice/ Germany-de_DE/ fileLibrary/ Download_540171919/ Germany/files/ Starkstrom_DuGB_12okt12_klein.pdf*, Last visited: 02.2017.

[75] L. Di Bert, P. Caldera, D. Schwingshackl, and A. M. Tonello. On noise modeling for power line communications. *ISPLC 2011, Udine, Italy*, 3-6 April 2011.

[76] S. Mudriievskyi, R. Radeke, and R. Lehnert. CSMA/CA: Improvements of the contention window adaptation. *ISPLC 2013, Johannesburg, South Africa*, 24 - 27 Mar. 2013.

[77] G. Prasad, L. Lampe, and S. Shekhar. In-band full duplex broadband power line communications. *IEEE Trans. Commun.*, Vol. 64, no. 9 September 2016.

[78] J. Bilbao, A. Calvo, I. Armendariz, and P. Crespo. Fast characterization method and error sequence analysis for narrowband indoor powerline channel. *ISPLC 2013, Johannesburg, South Africa*, 24-27 March 2013.

[79] pyLDPC module. *[Online] pypi.python.org/pypi/pyldpc/0.7.3*, Last visited: 02.2017.

[80] I. Tsokalo, R. Lehnert, and E. Mayer. Evaluation of broadband PLC for electricity metering with the field trial. *WSPLC 2015, Klagenfurt, Austria*, 21-22 September 2015.

[81] M. Zimmermann and K. Dostert. Analysis and modeling of impulsive noise in broadband powerline communications. *IEEE Trans. on Electromagnetic Compatibility*, vol. 44, no. 1 February 2002.

[82] D. Chariag, D. Guezgouz, J.-C. Le Bunetel, and Y. Raingeaud. Modeling and simulation of temporal variation of channel and noise in indoor powerline network. *IEEE Trans. on Power Delivery*, vol. 27, no. 4 October 2012.