CZECH TECHNICAL UNIVERSITY IN PRAGUE
Faculty of Electrical Engineering
Department of Radioelectronics

Master Thesis

Iterative Hierarchical Information Decoding Strategies for Wireless Physical Layer Network Coding

Supervisor: **Prof. Ing. Jan Sýkora, CSc.**
Student: Bc. Jan Smutný

May 2014

Proclamation

I declare, I have worked out this master thesis independently, and that I mentioned all the information sources according to “Methodical instruction about complying ethical principles during elaborating university theses”

Prague 12. 5. 2014

. . . .
Jan Smutný

Acknowledgments

I would like to thank my supervisor Prof. Ing. Jan Sýkorac CSc. and the rest of members of DiRaC research group for giving me valuable advices and suggestions.

Jan Smutný

Contents

Abstract

The main aim of this work was to get acquainted with Wireless Physical Network Coding and then utilize the iterative soft-information decoding based on Factor Graphs and the Sum-Product algorithm on the Hierarchical Decode and Forward and Joint Decode and Forward relaying strategies, cooping with the various scenarious of observations of the Hierarchical and Hierarchical Side information. The theoretical background about coding, Factor Graphs and WPNC is provided in the first part. In the second part we will give the designed for solving various scenarious of available HI and H-SI with comparasion based on simulation results.

Keywords
Factor Graphs, iterative soft-information decoding, Hierarchical Decode and Forward, Joint Decode and Forward

Abstract

Hlavním cílem této práce bylo seznámení se s bezdrátovým síťovým kódováním na fyzické vrstvě a poté využití iterativního dekódování s měkkou informací založeném na navrhu faktorového grafu na různé scénáře dostupných hierarchických a postranních hierarchických informací. Úvod do teorie kódování, faktorových grafů a bezdrátového síťového kódování na fyzické vrstvě je v první části. V části druhé jsou předloženy návrhy k řešení rozmanitých scénářů dostupných hierarchyckých a postraních hierarchických informací s jejich porovnáním na základě výsledků simulací.

Keywords
Faktorový graf, iterativní dekódování s měkkou informací, Hierarchical Decode and Forward, Joint Decode and Forward

List of Abbreviations

$AWGN$	Additive White Gaussian Noise
FG	Factor Graph
FFG	Forney Factor Graph
FN	Factor Node
GF	Gallois Field
HDF	Hierarchical Decode and Forward
HI	Hierarchical Information
HNC	Hierarchical Network Code
JDF	Joint Decode and Forward
NC	Network Coding
$LDPC$	Low Density Parity Check Code
PDF	Probability Density Function
SPA	Sum Product Algorithm
VN	Variable Node
$WPLNC$	Wireless Physical Network Layer Coding

List of Figures

List of Tables

Part I

Theoretical background

Chapter 1

Coding theory

1.1 Channel Capacity

1.1.1 Channel Coding Theorem

From channel coding theorem follows, that for any code rate $R < C$ exists code $(2^{nR}, n)$ with probability of error $P_e \to 0$ and analogically, any $(2^{nR}, n)$ code with $P_e \to 0$ must have $R \leq C$, where 2^{nR} denotes Typical set (further reading in).

1.1.2 Channel capacity

The channel capacity is defined as maximal mutual information over all input distribution.

$$C = \max_{p(x)} I(x; y) \tag{1.1}$$

For communication over wireless channel where the distribution of the noise is known as the AWGN, we can obtain capacity limit as

$$C = Blog(1 + SNR)[bit/s]. \tag{1.2}$$

This can be derived from original Shanno ergodic capacity which is considered as bits per channel usage (dimension)

$$C = lg\left(1 + \frac{{\sigma_x}^2}{{\sigma_w}^2}\right) \tag{1.3}$$

where ${\sigma_x}^2$ is variance of input distribution and ${\sigma_w}^2$ is variance of complex AWGN.

1.2 Elements of error-correction codes

1.2.1 Introduction to block codes

For purposes of this work, we restrict ourselves on linear error-correcting block codes. The main benefit of this code is that they are easily implementable in terms of factor graphs, which is the main benefit.

The reason why block codes are called block is, that the data are at first sorted into the blocks which are then processed according to a coding function. That means that the receiver has to wait for the whole block of data words for them to start decoding and even thought this can be consider as drawback the whole block of data words are relatively cheap to implement into the system because theoretically no memory is needed.

1.2.2 Mathematical definition

Information source produces data information message $\mathbf{d} = [\mathbf{d}_1,\ldots,\mathbf{d}_L]$ with N_d dimensional data words $\mathbf{d}_n = [d_{n,1},\ldots,d_{n,N_d}]$, where each data symbol $d_{n,k}$ is from alphabet $\mathcal{A}_d$ with a size M_d=$|\mathcal{A}_d|$. This data are input for encoder, which maps these data to codewords $\mathbf{c}_n = [c_{n,1},\ldots,c_{n,N_c}]$ with dimension N_c, where each symbol $c_{n,k}$ is from alphabet $\mathcal{A}_c$ with size M_c=$|\mathcal{A}_c|$. Codewords then form codeword sequence $\mathbf{c} = [\mathbf{c}_1,\ldots,\mathbf{c}_L]$. This mapping can be formally written as

$$\mathbf{d} \mapsto \mathbf{c}.$$

Now, let's consider code space $\mathcal{S}$, which is based on a field ($\mathcal{F}$,+,×), where $\mathcal{F}$ is set of elements (numbers), which is closed under + and × operators (further reading in **??**).

Definition 1.2.1. Code is linear if its $Nc - tuple$s from alphabet $\mathcal{A}$ belongs to a code space S. We mostly consider binary codes ($\mathcal{F}$=$\mathcal{F}_2$) from set 0,1 with modulo-2 arithmetic.
For linear code must hold:

- scalars $a_1, a_2 \in \mathcal{F}$; $\mathbf{c}_1, \mathbf{c}_1$ are valid codewords (1.4)
- $a_1\mathbf{c}_1 + a_2\mathbf{c}_2$ must be a valid codeword (1.5)
- linear code contains all-zero codeword $\mathbf{c}$=$\mathbf{0}$ (1.6)

Definition 1.2.2. Code is systematic if data word is a part of codeword. The remaining part of codeword fills a parity word.

Definition 1.2.3. Code rate is in a sence of number of bits per one codeword over channel symbol dimension:

$$R_{2d} = \frac{\lg M_d^{N_d}}{N_c} \text{ [bit/dimension]}. \tag{1.7}$$

In most cases $M_d = 2$:

$$R = \frac{N_d}{N_c}. \tag{1.8}$$

Definition 1.2.4. Generator matrix $\mathbf{G}$ ($N_c \times N_d$) for block code is matrix:

$$\mathbf{G} = \begin{bmatrix} \mathbf{I}_{N_d} \\ \mathbf{P} \end{bmatrix} \tag{1.9}$$

where parity matrix $\mathbf{P}$ is of size (N$_c$-N$_d$) × N$_d$ and columns of $\mathbf{G}$ form N_d dimensional basis of codeword sub-space.

Definition 1.2.5. Parity check matrix $\mathbf{H}$ with size ($N_c \times (N_c - N_d)$:

$$\mathbf{H} = \begin{bmatrix} -\mathbf{P}^T \\ \mathbf{I}_{N_c-N_d} \end{bmatrix}. \tag{1.10}$$

In a special case of binary code (M_c=2):

$$\mathbf{H} = \begin{bmatrix} \mathbf{P}^T \\ \mathbf{I}_{N_c-N_d} \end{bmatrix}. \tag{1.11}$$

Definition 1.2.6. Codeword $\mathbf{c}$ is a set:

$$\mathbf{c} = \{\mathbf{Gd} : \mathbf{d} \in \mathcal{F}^k\};\ \mathbf{Hc} = 0 \tag{1.12}$$

Due to the orthogonality:

$$\mathbf{H}^T\mathbf{G} = [-\mathbf{P} \ \ \mathbf{I}_{N_c-N_d}] \begin{bmatrix} \mathbf{I}_{N_d} \\ \mathbf{P} \end{bmatrix} = -\mathbf{P}\mathbf{I}_{N_d} + \mathbf{I}_{N_c-N_d}\mathbf{P} = \mathbf{0} \tag{1.13}$$

Definition 1.2.7. Hamming space codes are defined as binary codes:

$$M_c = M_d = 2, \text{ or non-binary} M_c = M_d > 2; N_c > N_d$$

Definition 1.2.8. Hamming distance, which is applied on Hamming space codes, is number of symbols in which two words differ

$$\rho_H(\mathbf{c}_1, \mathbf{c}_2) = \sum_k 1 - \delta[c_{1,k} - c_{2,k}]$$

Definition 1.2.9. A code is a **perfect t-error-correcting code** if the set of t-spheres centred on the codewords of the code fills the Hamming space without overlapping.

Definition 1.2.10. Hamming code (n,k) is a binary cyclic block code, with a number of parity symbols $m = n - k$ with minimal Hamming distance $\rho_{min} = 3$ (Hamming (7,4) block code).

1.3 LDPC codes

LDPC code is a linear block code with a "parse" parity-check $m \times n$ matrix $\mathbf{H}$. Matrix is sparse in sence of small number of nonzero elements. Gallanger proposed construction of such matrix in random way, which means random placing 1's and 0's, but respecting the condition, that number of 1's in each row must be equal to d_r and each column equal to d_c. Such code is then reffered to as *regular* (d_c, d_r) LDPC code of length n. Gallanger showed, that the minimum distance of regular LDPC code increases linearly with n if $d_v \geq 3$. This is the reason why regular LDPC codes are designed with d_v and d_c on the order of 3 or 4. Now we would like to derive the formula for code rate.Because of random construction of matrix, there is no guarentee that a matrix is full rank. If we eliminate the linearly dependent rows to find a $(n-k) \times n$ parity check matrix, we lose the regular property and this is not what we want. So we consider the designed rate of the code as:

$$R = 1 - \frac{m}{n} = 1 - \frac{d_c}{d_r}. \tag{1.14}$$

In order to have R<1 for regular code,

$$md_r = nd_c \text{ and } d_c < d_r \tag{1.15}$$

The most important advantage of LDPC codes is, that the parity check matrix $\mathbf{H}$ can be interpreted as the bipartite (Tanner) graph. The term bipartite graph will be explained in section 3.
The second cathegory of LDPC codes are irregular LDPC codes, which can't be defined in therm of the degree of d_r or d_c, because this number can by for each row or column different.

Chapter 2

Algebraic structures

This subsection gives an elementary mathematical background for our latter application purposes.

Definition 2.0.1. A binary operation on a nonempty set $\mathcal{S}$ is generally map $\mathcal{S} \times \mathcal{S} \to \mathcal{S}$.

Definition 2.0.2. A *monoid* $(M; \odot)$ is a set of elements $\mathcal{S}$ $(\mathcal{S} \odot \mathcal{S} \to \mathcal{S})$ on which an *associative* binary operator $\odot$ is defined as

- $\forall a, b, c \in \mathcal{S}; a \odot (b \odot c) = (a \odot b) \odot c.$

With respect to that operation, monoid also contain an identity element e, such that:

- $\forall a \in \mathcal{S}; \exists e \in \mathcal{S}; (a \odot e) = (e \odot a) = a.$

Definition 2.0.3. A *group* $(\mathcal{G}; \odot)$ is a monoid with an extra inverse element.

- $\forall a \exists a' : a \odot a' = a' \odot a = e; \forall a, a' \in \mathcal{S}.$

We talk about *commutative group* in case of:

- $a \odot b = b \odot a; \forall a, b \in \mathcal{G}$

The group is finite if size (or $|\mathcal{G}|$) of the group is finite. Group with $\odot$ operator is called *multiplicative group* and group with $\oplus$ operator is called *additive group*.

Definition 2.0.4. A *ring* is an algebraic structure $(\mathcal{R}, \oplus, \odot)$ consisting of a set and two operations, for which:

- $(\mathcal{R}, \oplus)$ forms a commutative group.
- $(\mathcal{R}, \odot)$ forms a monoid.
- $\forall a, b, c \in \mathcal{R}$, the operation $\odot$ distributes over $\oplus$:
  $a \odot (b \oplus c) = a \odot b \oplus a \odot c;$
  $(b \oplus c) \odot a = b \odot a \oplus c \odot a;$
- Additive identity $0 \in \mathcal{R}$: $\quad a + 0 = a;\ 0 + a = a$
- Multiplicative identity $1 \in \mathcal{R} : a \cdot 1 = a;\ 1 \cdot a = a$
- Additive inverse: $\forall a \in \mathcal{R}\ \exists - a \in \mathcal{R} : a + (-a) = (-a) + a = 0$
- It is impossible to generally define define something like multiplicative inverse, but in some special cases it is possible. If in given ring is commutative operator $\odot$, then is called *commutative ring*. If a ring is finite size, then is called $finite\ ring$.

For clarification, we present the simplest example of finite ring, the set of integers modulo q, i.e. $0, 1 \dots q-1, (q) = q$, in which the operations addition and multiplication are adition and multiplication modulo q.

Definition 2.0.5. A $finitefield$ ($Galoisfield$) of order (size) q, $GF(q) = \mathbb{F}_q$ is a ring containing multiplicative inverse for all elements except the element 0. The inverse is usually written as a^{-1}

So we can interpret subtraction as addition of the additive inverse and division as multiplication by the multiplicative inverse (except zero) respectively. Size q of finite fields is an integer power of a prime number $q = p^m$. For prime fields with $m = 1$, the elements can be written $0, 1, 2, \dots q-1$ and $\oplus/\odot$ are addition/multiplication mooodulo q (as for finite ring). For $extensionfields$, in which $m > 1$, the elements are polynomials with coefficients $0, 1, \dots p-1$ of order up to $m-1$. Then, the $\oplus$ is addition of the polynomials taking the coefficients modulo p, while $\odot$ is polynomial multiplication taken modulo some irreducible polynomial.

Because finite fileds are the most frequently exploited in coding and other ingeneering branch, we give here an short and clear overview of properties.

- operation $+$
    - $\mathcal{F}$ is closed under $+$
    - associative $(a+b)+c = a+(b+c)$
    - commutative $a+b = b+a$
    - zero element $a+0 = a$
    - negative element $a+(-a) = 0$
- operation $\times$
    - F is closed under $\times$
    - associative $a(bc) = (ab)c$
    - commutative $ab = ba$
    - identity element $1a = a$
    - inverse element $\forall a \neq 0,\ aa^{-1} = 1$
- $\times$ distributive over $+$
    - $a \times (b+c) = a \times b + a \times c$

Definition 2.0.6. A *bijection* (or $bijectivefunction$, or sometimes called *one−to−one correspondence*) is mapping between elements of two sets, where every element of A mapped into exactly one element of B ($f : X \mapsto Y$).Mathematically written, for bijection must hold:

$$\forall x \in A : f(x) \in B \tag{2.1}$$

$$\forall x, y \in A : x \neq y \Rightarrow f(x) \neq f(y) \tag{2.2}$$

$$\forall z \in B \exists x \in A : f(x) = z \tag{2.3}$$

An *injective* function $f : X \mapsto Y$ maps at most one element of A into B. Conditions (2.1) and (2.2) must hold but not necessarily condition (2.3).
A $surjective$ function maps at least one element of A into B. Condition (2.3) hold but not necessarily conditions (2.1) and (2.2).

Remark 2.0.1. For previous mentioned mapping functions from set A to set B arise these consequences:

- bijection: $|A| = |B|$

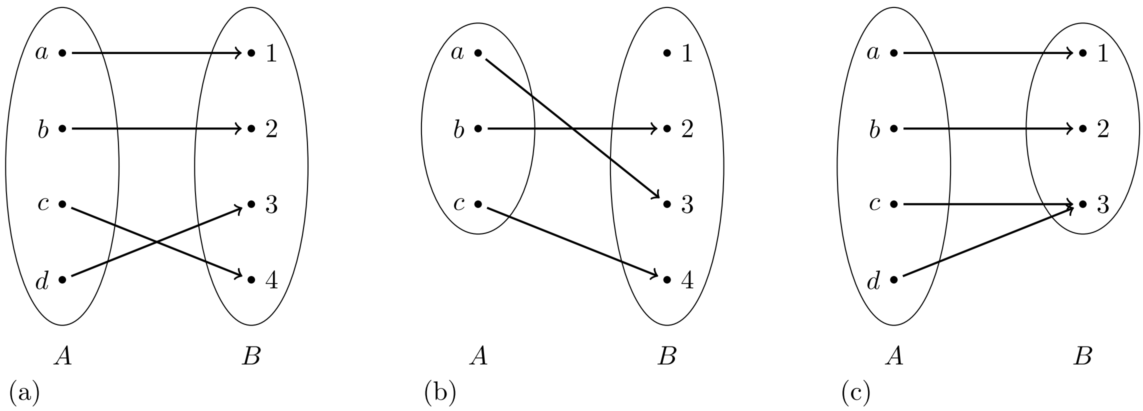


Figure 2.1: Illustration of (a) bijection; (b) injection bot not surjection; (c) surjection but not injection

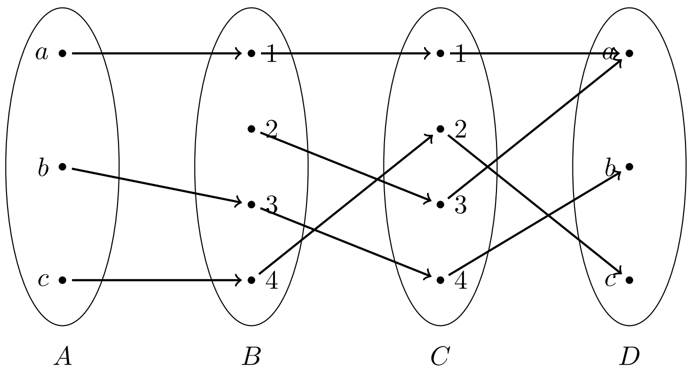


Figure 2.2: Concatenation of non-bijective functions forming a bijection

- injection: $|B| \geq |A|$
- surjection: $|A| \geq |B|$

Remark 2.0.2. If each function in composition of several functions is bijective, then the composition is bijective. But also there exists the concatenation of non-bijective functions that forms a bijection, as can be seen for example in 2.2. From this example is clear, that this rules must hold:

- Occurence of non-surjective function ($A \mapsto B$) must preceed a non-injective ($C \mapsto D$).
- The first function must be injective and the last surjective.
- First and last sets must have the same cardinality.
- All intervening sets must have cardinality at least as large as first(last).

Lemma 2.0.1. *Multiplication of an element* x *from a set* $\mathcal{S}$ *using* $\odot$ *by a coefficient* a *from a coefficient set* $\mathcal{S}_c$ *sucg that* $\mathcal{S}$ *is closed on the operation for* $\odot$ *which the associative law applies, constitues a bijection if and only if the coefficient has an inverse on* $\odot$ *within* $\mathcal{S}_c$.

Proof. Consider the function $f(x) \in \mathcal{S}$:

$$f(x) = a \odot x; x \in \mathcal{S}, a \in \mathcal{S}_c, 1 \in \mathcal{S}_c$$
$$f(x) = f(y) \Rightarrow a \odot x = a \odot y; y \in \mathcal{S}$$

If a has an inverse a^{-1} in $\mathcal{S}_c$:

$$a^{-1} \odot (a \odot y) \Rightarrow (a^{-1} \odot a) \odot x = (a^{-1} \odot a) \odot y \Rightarrow 1 \odot x = 1 \odot y \Rightarrow x = y.$$

So (2.1),(2.2),(2.3) is verified and $f(x)$ is the bijection from $\mathcal{S}$ to itself.

If a has an inverse a^{-1} in $\mathcal{S}_c$,then f has an inverse function f^{-1}:

$$f^{-1}(z) = a^{-1} \odot z = x; z \in \mathcal{S}$$

Again, we can assume bijection from $\mathcal{S}$ to itself. And vice versa, if a has no inverse (f cannot be bijection).

According to deffinition (2.0.2), $\mathcal{S}_c$ can be considered as monoid. A corollary to 2.0.1 is that for a multiplicative group $\mathcal{G}$ multiplication by a coefficient provided by any member of the group constitues a bijection. However it does not follow, that the set must be a group, since some elements of a monoid may have an inverse, and multiplication by these also constitues a bijection. So in fact at least one element, the identity, must have an inverse.

The proof of 2.0.2 can be applicated also on $\oplus$ operation with the all consequences.

Lemma 2.0.2. For a ring, addition of any element, and multiplication by any element which has an inverse (of which there is at least one),constitues a bijection

Lemma 2.0.3. For a Gallois field, addition of any element, and multiplication by any non-zero ,constitues a bijection.

Chapter 3

Factor graphs and the summary propagation algorithm

3.1 Introduction

The Factor graph is very universal mathematical tool which origin lies in coding theory, but offers many capabilities for solving artifical inteligence, signal processing and generally digital communications problems. The main task is to solve problems, for example a global function of many variables in computionally effective way, so that the "global" function is factored into product of simpler "local" functions depending on smaller subset of variables.

The computing algorithm, generally called $Marginalization - Combination$ or $Summary - Propagation$ algorithm, of the "local" functions, is interpreted as passing "messages" along the edges of graph. The sum-product algorithm is the main form of marginalization-combination (or summary propagation) algorithm utilized for purposes of this work.

3.2 Terms definiton

3.2.1 Forney Factor Graph

In this work will be considered only Forney factor graph style. The main difference between FFG and the other factor graph style is evident from figure 3.2.1. The both solves the same factorization problem of some function f.

$$f(a, b, c, d, e) = f_1(a, b, c)f_2(c, d)f_2(d, e) \tag{3.1}$$

Definition 3.2.1. The Forney Factor Graph is a **bipartite graph** consisting of nodes representing some factors (function), and edges, or "half edges" representing some variables. FFG is defined by the following rules:

- There is a (unique) node for every factor.
- There is a (unique) edge or half edge for every variable.
- The node representing some factor f is connected with the edge (or half edge) representing some variable a if and only if f is a function of a. As consequence, no variable node can be connected to more then two factors, but as we will see later, this restriction can be circumvented.

Definition 3.2.2. The Factor graph is cycle-free if the graph is without cycle. The graph without any cycle is a tree.

In another words, the path between particular factor and particular variable node is unique. However, from definition 3.2.1 is this property partiall consequence.

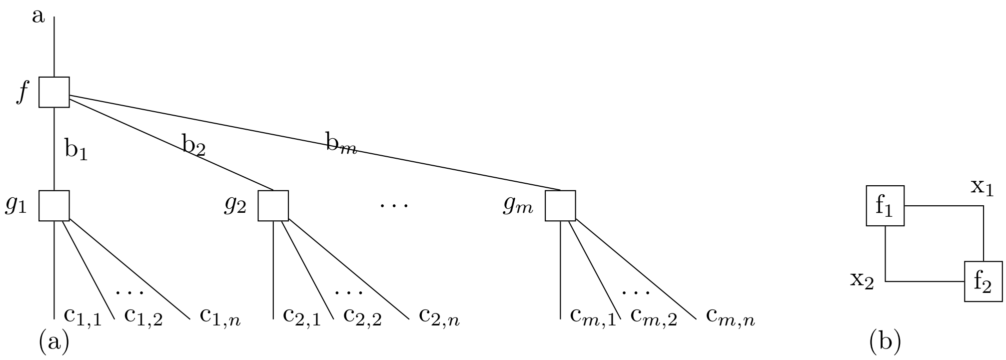


Figure 3.1: (a) Tree-like (Cycle-free) FFG (b) Cycled FFG

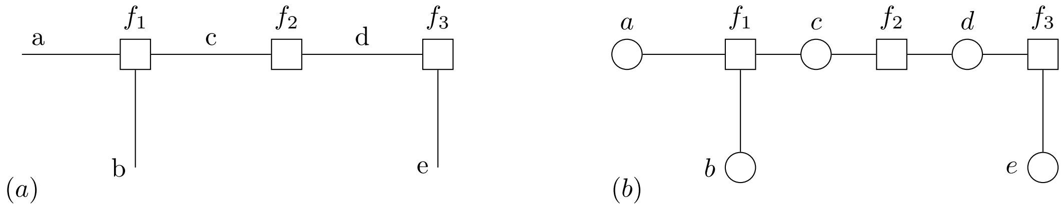


Figure 3.2: (a) Forney factor graph style (FFG). (b) Factor graph style

Definition 3.2.3. The Factor graph has cycles, if graph doesn't fulfil (3.2.1).
The difference is obvious from figure 3.2.1.

3.2.2 Global function

As can be seen in example 3.1, we have some function f representing the **global function**, which can be factorized into the local functions f_1, f_2 and f_3.

Generally, the global function f is in some domain (configuration space) Ω, where Ω as in example 3.1 can be set e.g. $\{0, 1\}^5$. Generally $f : \Omega \mapsto C$, where C is codomain of function g.

3.2.3 Marginalization-Combination Algorithm (MCA)

Generally, we have to define some abstract fundamental operations,that, depending on concrete application, proceed to appropriate operation. Then the global function from example 3.1 should be written in form

$$f(a, b, c, d, e) = f_1(a, b, c) \circ f_2(c, d) \circ f_3(d, e) \tag{3.2}$$

Definition 3.2.4. Factorization of local functions is written in the form (factors) $f(\centerdot) = f_1(\centerdot) \circ \ldots \circ f_n(\centerdot)$. The operation:

- $\circ$, or sometimes denoted as $\prod^*$ is a combination operator
- $\square$, or sometimes denoted as $\sum^*$ is a marginalization operator
- combination distributes over marginalization

$$a \circ (b_1 \square b_2) = a \circ b_1 \square a \circ b_2 \tag{3.3}$$

Provided the example (3.2), where $f(a, b, c, d, e)$ is a global function, and we are interested e.g. in marginal function $f(c)$:

$$f(c) = \sum_{a,b,d,e}^{*} f(a, b, c, d, e) \tag{3.4}$$

then employing the distribution property, the marginalization of global function splits into the combination of the marginalized local functions

$$f(c) = \sum^{*} f(a,b,c,d,e) = \left(\sum_{a,b}^{*} f_1(a,b,c)\right) \circ \left(\sum_{d}^{*} f_2(c,d)\right) \circ \left(\sum_{e}^{*} f_3(d,e)\right) \tag{3.5}$$

The property of factorization is mostly used in Markov chain model.

Definition 3.2.5. Assume a Markov chain $x \to y \to z$, where x, y, z are random variables, then:

$$p_{x,y,z} = p(z|x,y)p(x,y) = p(z|y)p(x,y) = p(z|y)p(y|x)p(x) \tag{3.6}$$

where we utilized the Markov chain property:

$$p(z|x,y) = p(z|y), \text{ where } p(\boldsymbol{.}) \text{ is PDF.} \tag{3.7}$$

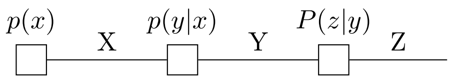


Figure 3.3: An FFG of Markov chain

3.2.4 Message passing on the FFG

As follows from previous sections, the factor graph represents structure of some system. Till now, we have mentioned two building blocks of factor graph - the variable node and the check node. If we consider tree-like graph, as depicted in figure 3.2.1(a), we start computing the marginal function from leaves and succesively continue to the top where is the variable of our interest. Geting back to the equation 3.2.3 and the relating figure 3.2.1(a), we add the folloving notations (figure 3.2.4):

$$\mu_{f_3 \to d}(d) = \sum_{e}^{*} f_3(d,e) \tag{3.8}$$

$$\mu_{f_2 \to c}(c) \sum_{d}^{*} f_2(d,e) \tag{3.9}$$

$$\mu_{f_1 \to c}(c) \sum_{d}^{*} f_2(a,b) \tag{3.10}$$

As can be seen, we have introduced the notation for results of individual local marginalizations $\mu_{f_{\boldsymbol{.}} \to \boldsymbol{.}}(\boldsymbol{.})$ called messages.

Passing messages from node to node represents sequential evaluation of all local marginalizations, and as can be seen from figure 3.2.4, it is executed in both directions, where form and interpretation depend on a particular application.

Finally, result of local marginalization $f(c)$ (where $f(\boldsymbol{.})$ can represent for example, as was mentioned in definition 3.2.3, the PDF)

$$f(c) = \mu_{f_2 \to c}(c) \circ \mu_{f_1 \to c}(c) \tag{3.11}$$

where the message $\mu_{f_1 \to c}(c)$ is sometimes denoted as forward message, and message $\mu_{f_2 \to c}(c)$ can be denoted as backward message.

Now we skip to the next chapter, where the Sum-Product algorithm is described. There we will define next examples of message passing rules associated with given nodes, essential for purposes of the SPA.

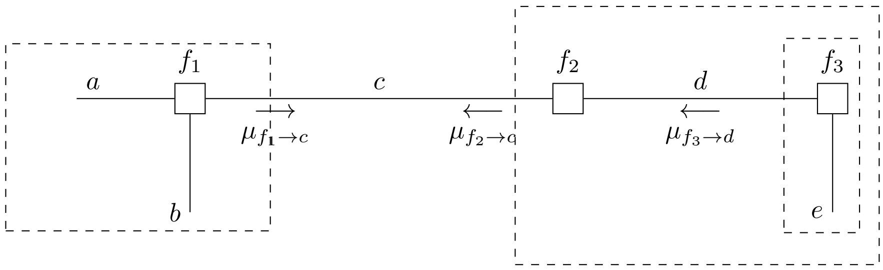


Figure 3.4: FFG message passing

3.2.5 Sum-Product Algorithm (SPA)

Now let's define Sum-Product Algorithm, which is particular form of Marginalization-Combination Algorithm. The result of this task is to determine an appropriate form of functions defined in 3.2.3.

Before defining concrete functions, it is necessary to say, that the main application of SPA is to solve the evaluation of marginalized Bayessian MAP objective function for given variable nodes, which was already hinted in definition 3.2.3.

For simplicity and clarity we present overview of MCA operations with corresponding SPA operation in table 3.1.

It is no reason to having doubts about the validity of SPA instead of MCA operations, because it is easy to prove the distributivity and commutativity of the summing and multiplying operators, which is the fundamental property for the message passing purposes.

MCA operation	**SPA operation**
marginalization ($\sum^*$ or $\square$)	summation ($+$)
composition ($\prod^*$ or $\circ$)	multiplying ($\times$)
message (μ)	probability densities (p)
factor node (f)	conditional probabilities (p)
marginal	marginalized Bayesian MAP density (belief)

Table 3.1: SPA operations table

Definition 3.2.6. Probabilistic (soft-information) messages on Cycle-Free factor graph consist of:

- **Forward messages**
  - a priori PDF $p(x)$
- **Backward messages**
  - likelihoods $p(y = y^{(0)}|x)$
- **Factor**
  - conditional PDF $f(x|y_1, \ldots, y_n) = p(x|y_1, \ldots, y_n)$
- **Source**
  - a priori PDF $p(x)$
- **Observation**
  - Dirac delta PDF $\delta(y - y^{(0)})$
- **Belief**
  - MAP decision objective function (valid only for Cycle-Free FG)
  - Product of all incoming messages at VN

$$B(d) = \prod_i \mu_i(d) = p(x = x^{(0)}, d)$$

- ∘ forward & backward messages multiplication

$$B(d) = p(x = x^{(0)}, d) = p(d)p(x = x^{(0)}|d)$$

As we mentioned in previous definition e.g. *Factor*, where we considered PDF, then we talk about:

- **continues-valued variables**, where PDF is intertwine with $\int(\centerdot)dy_i$ operator.

Or

- **discrete-valued variables**, where we consider probability mass function and the appropriate operator $\sum(\centerdot)$

Note, that the variable names are here only illustrative, and have no exact meaning.

3.3 Sum-Product Algorithm update rules

In the following subsections the update rules are considered only for discrete-valued variables (for continues variables it is analogous procedure but instead of summation operator is used the integration one).

3.3.1 Factor Node

Lets consider the situation depicted in figure 3.3.1. To be fully explicit, let the function f be provisionally open.

In case of MAC, we have the following update rule:

$$\mu_{f\to x}(x) = \sum_{y_1,\dots,y_n}^{*} \left(f(x, y_1, \dots, y_n) \prod_{i}^{ast} \mu_{y_i\to f}(y_i) \right). \tag{3.12}$$

The SPA update rule:

$$\mu_{f\to x}(x) = \sum_{y_1,\dots,y_n} \left(f(x|y_1, \dots, y_n) \prod_{i} \mu_{y_i\to f}(y_i) \right). \tag{3.13}$$

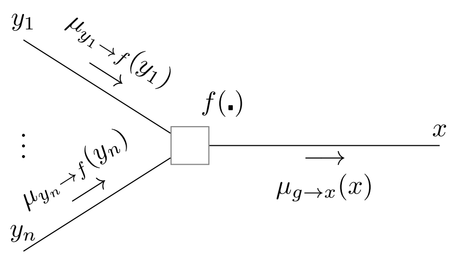


Figure 3.5: Sum-Product Rule

3.3.2 Variable/Equality Node

In factor graph type described in [1], we talk about Variable Node and the situation is expressed in figure 3.3.2 with the general (according to MCA) update equation

$$\mu_{x\to(\centerdot\centerdot)}(x) = \prod_{i}^{*} \mu_{(\centerdot_i)\to x}(x) \tag{3.14}$$

and update rule corresponding to SPA update

$$\mu_{x \to (\bullet\bullet)}(x) = \prod_i \mu_{(\bullet_i) \to x}(x) \tag{3.15}$$

Totally anoteher situation is for FFG, where we have to utilize the so called **Equality node** (or sometimes called **Replication variable node**). The corresponding FFG of this factor node (from this point anymore variable node) is depicted in figure 3.3.3 with related equation:

$$\mu_{x \to (\bullet\bullet)}(x) = \sum_{(\bullet_1)\dots(\bullet_n)} \delta\big((x) - (x_1)\big) \dots \delta\big((x) - (x_n)\big) \prod_i \mu_{(\bullet_i) \to x}(x_i). \tag{3.16}$$

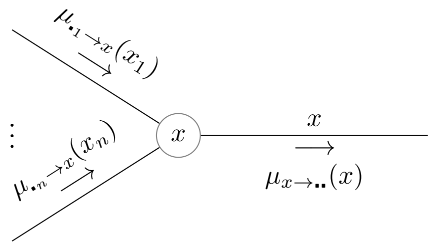


Figure 3.6: Variable node update

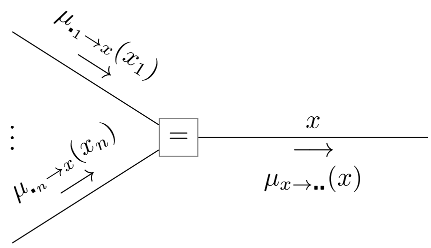


Figure 3.7: Variable node update

3.3.3 Source/Observation Factor Node

The source or observation is fixed, thus no marginalization or other operation can be directly made. Then this is the single edge generating message directly.

$$\mu_{f \to x}(x) = f(x) \tag{3.17}$$

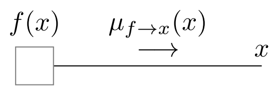


Figure 3.8: Variable node update

3.3.4 Memoryless channel model

Let's have vector $\vec{y} = (y_1, \dots, y_n)$ representing the channel output symbol sequence and block of $\vec{x} = (x_1, \dots, x_n)$ of channel input symbols. Then channel model $p(y|x)$ describing that y is received when x is transmitted is depicted in figure 3.3.4:

$$p(y|x) = \prod_{i=1}^{n} p(y_i|x_i) \tag{3.18}$$

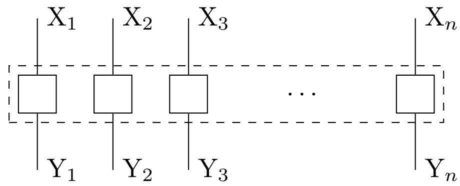


Figure 3.9: Memoryless channel

3.3.5 State-space channel model

The state-space representation of channel with internal states depicted in figure 3.3.5 is given by equation:

$$p(y, s|x) = p(s_0) \prod_{i=1}^{n} p(y_i, s_i|x_i, s_{i-1}). \tag{3.19}$$

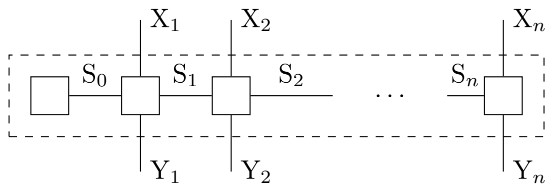


Figure 3.10: State-space channel model

3.3.6 Messages for binary arithmetics on GF(2)

For discrete messages, such as binary messages is for our purposes (probilistic modeling) no other option then considering probability mass function (PMF). Forward as well as backward messages have some probability for state one and complementary probability for state zero. Thus keeping probability only for one state is always sufficient.

- Forward message $p(d) = \{p_d(0), p_d(1)\}$ (3.20)
- Backward message $p(x^0|d) = \{p_d(x^{(0)}|0), p_d(x^{(0)}|1)\}$ (3.21)

Further, for simple notation, we consider $\mu(0)$ (or $\mu(1)$) for both backward and forward recursion.

- probability difference (PD) $\Delta(d) = \mu_d(0) - \mu_d(1)$ (3.22)
- Likelihood ratio (LR) $L(d) = \dfrac{\mu_d(0)}{\mu_d(1)}$ (3.23)
- Log-likelihood ratio (LLR) $LLR(d) = log\dfrac{\mu_d(0)}{\mu_d(1)}$ (3.24)

(3.25)

As the masseges representing probabilities pass on the edges and are modified according to update rules on given factor nodes, it is necessary to norm them on the output from an each node. It is implied with the following equation, where we don't strictly say what the node (from which the message goes out) represents and thus it can be used generally:

$$\mu_{norm}(0) = \frac{\mu(0)}{\mu(1) + \mu(0)} \tag{3.26}$$

$$\mu_{norm}(0) = \frac{\mu 1)}{\mu(1) + \mu(0)}, \tag{3.27}$$

where μ_{norm} represents the updated normalized message. Then we can write:

$$\mu_{norm}(0) = 1 - \mu_{norm}(1) \text{ and vice versa.} \tag{3.28}$$

3.4 FFG of Codes

Block codes were described in chapter 1, but to be fully explicit, let's again consider some error correcting block code C fromed from vector space F^n where we restrict ourselves on binary modulo-2 arithmetic $F = F_2$ (shortcut F).

The code is written:

$$C = uG : u \in F^k \tag{3.29}$$

and must hold

$$C = x \in f^n : Hx^T = 0, \tag{3.30}$$

where u are input symbols and x are coded symbols.

Then according to 3.4 we define the indicator (or characteristic) function

$$I_C = F^n \to 0, 1 : x \mapsto \begin{cases} 1, & \text{if } x \in C \\ 0 & \text{else} \end{cases} \tag{3.31}$$

Now, if we put channel model together with some code (X represent codewords) as depicted in figure 3.4 then the joint a posteriori probability of coded symbols

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} \tag{3.32}$$

and after neglecting some scaling factors (which would be anyway after normalizing reduced) and for fixed observation y:

$$p(x|y) \propto p(y|x)I_C(x) \tag{3.33}$$

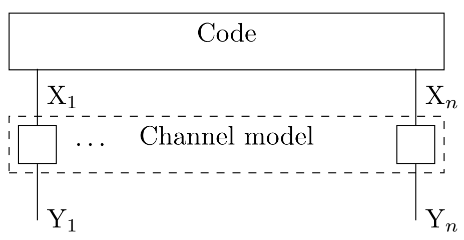


Figure 3.11: FFG of code with memoryless channel

The indicator function is clarified in the subsection 3.4.3.

3.4.1 FFG of linear Block codes

Let's have Hamming(7,4) block code with following parity check equation:

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Then the indicator function I_C defined in previous section is:

$$\begin{aligned} I_C(x_1, \ldots, x_n) &= \delta(x_1 \oplus x_2 \oplus x_4 \oplus x_5) \\ &\quad \cdot \delta(x_1 \oplus x_3 \oplus x_4 \oplus x_6) \\ &\quad \cdot \delta(x_2 \oplus x_3 \oplus x_4 \oplus x_7) \end{aligned} \tag{3.34}$$

where δ represents Kronecker delta function.

The resulting factor graph of this code is in figure 3.4.1. The new factor node called **parity check node** will be fully described in the following subsection, but the function of this node is evident from name, and the equation 3.4.1. Each parity check node corresponds to one row in matrix 3.4.1 and each euality node to each column (where is more then one 1 element) of this parity check matrix. We can apply this this method for every linear block code and then it is called Tanner graph. As can be seen from figure 3.4.1, this graph is bipartite graph.

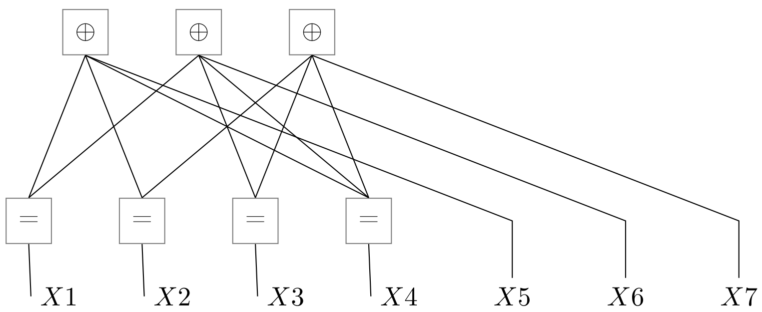


Figure 3.12: FFG Hamming(7,4)

3.4.2 Graph of LDPC codes

LDPC codes were described in 1.3 and now, let's look on this graph. As can be seen from figure 3.4.2, we have similar structure as in previous Hamming block code, which is no surprise, but additionally, we can consider random connections denoted as "Interleaver" between check and equality nodes, which comes from the large sparse matrix with big cycles. In other words, the individual code symbols are considered as independent. From figure is evident, that it corresponds to regular LDPC (4,3) code - each equality node has three connecting branches to parity check nodes with degree four.

3.4.3 Parity check factor node

Taking figure 3.4.3 as example, then parity check node fulfils the equation:

$$\mu(c) = \sum_{a,b} \delta(c - (a \oplus b))q(a)q(b) \tag{3.35}$$

If we convey it into probability domain:

$$\mu(c) = \begin{Bmatrix} \mu_c(0) \\ \mu_c(1) \end{Bmatrix} = \begin{Bmatrix} \mu_a(0)\mu_b(0) + \mu_a(1)\mu_b(1) \\ \mu_a(0)\mu_b(1) + \mu_a(1)\mu_b(0) \end{Bmatrix} \tag{3.36}$$

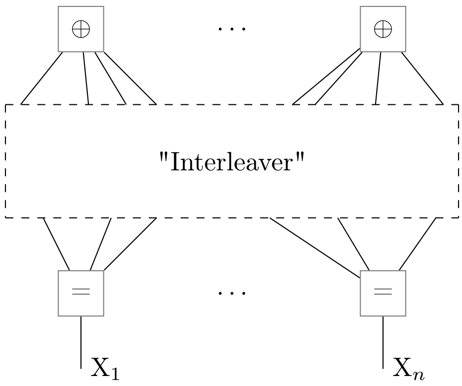


Figure 3.13: An FFG of LDPC

This node is fully symmetric so this can be rewritten for whatever output node.

It is evident, that number of required operations for computing the output message grows exponentialy with number of incoming messages. The solution of this unpleasant problem can be procedure suggested in [**?**, p. 215]. Let's have the probability, that on node a and b is even number of 1's:

$$
\begin{aligned}
Pr[a \oplus b] &= \mu_a(1) * \mu_b(1) + (1 - \mu_a(1))(1 - \mu_b(1)) && (3.37)\\
&= 1 - \mu_a(1) - \mu_b(1) + 2\mu_a(1)\mu_b(1) && (3.38)\\
&= \frac{1}{2}(2 - 2\mu_a(1) - 2\mu_b(1) + 4\mu_a(1)\mu_b(1)) && (3.39)\\
&= \frac{1}{2}[1 + (1 - 2\mu_a(1))]. && (3.40)
\end{aligned}
$$

If we will continue with composing more and more input nodes (further denoted as $x_1 \dots x_n$) we will realize, that it can be written in generic form:

$$
Pr[x_1 \oplus \dots \oplus x_n = 0] = \frac{1}{2} + \frac{1}{2}\prod_{i=1}^{n}(1 - 2\mu_{x_i}(1)). \tag{3.41}
$$

Then the message $\mu(c)$:

$$
\mu_c(0) = \frac{1}{2} + \frac{1}{2}\prod_{i=1}^{n}(1 - 2\mu_{x_i}(1)). \tag{3.42}
$$

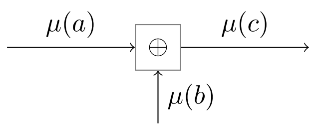


Figure 3.14: An FFG of LDPC

3.4.4 Message passing on graph with cycles

We have already mentioned cycles in graph in subsection 3.4.2, but they were also visible on the first sight in figure of Hamming block code 3.4.1. Provided that graph has cycles, the SPA can be still used, but the resulting belief or other required value is only an approximation, becase iteration are necessary, and in real system, only a finite number of iteration is possible. The number of required iterations strictly depends on the application. It should be compromise between reasonable computational time and desired accuracy of result. In case of decoding block codes, for example LDPC, we can execute iterations and always after reasonable number of iterations check, wether all check nodes conditions are satisfied, and if aren't, then start computing again.But as a precaution against the endless loop, we should anyway define the overall maximum permitted number of iterations.

Chapter 4

Wireless physical layer network coding

4.1 Introduction

4.1.1 Network coding

In case of standard NC, a network node applies a joint coding function on a set of incoming data streams instead of standard switching between them as in conventional network layer. The best example is two way relay channel, where the case of standard routing is depicted in figure 4.1 and network coding case is apparent from figure 4.2).

The classical network coding considers incoming signals in discrete channels. Then coding is applied on discrete symbols and thus there is no demand on further advanced technique such as interference cancelation etc. The data are then fully decoded on destination.The drawback against the standard switching is necessity of having complete knowledge about the network topology.

4.1.2 Wireless network coding

In case of wireless channel, where we want be able to comunicate in one channel and during the same timeslot together with the other users, the situation is much more complicated and we have to accede to Wireless physical layer network coding which has potential to solve these problems.

WPLNC works directly on physical layer. The main idea standing behind is trying to utilize the superposed incoming signal (the one changing electromagnetic field) in proper way to reach the additional potential troughput benefit. On the other hand, realizing, that the wireless communication channel is absolutely non-deterministic and we have to face out the phenomena such as attenuation of signal, phase rotation, time delay, multipath spreading, dispersion in the frequency etc., this is real challenge and most of this problematic is still under research.

Figure 4.1: Traditional routing strategy for the two-way relay channel requires four time slots to deliver 2 packets(bits/symbols/etc.), which means troughput 1/2 packets per channel use. (a): During the first time slot, user A sends its message to the relay. (b) During the second time slot, user B sends its message to the relay. (c) During the third time slot, the relay sends the message from A to user B. (d) During the fourth time slot, the relay sends the message from B to user A.

Figure 4.2: A network coding strategy for the two-way relay channel requires three time slots. During the first time slot (a), user A sends its message to the relay. (b) During the second time slot, user B sends its message to the relay. (c) During the third time slot, the relay sends the sum of the messages $A \oplus B$ to both users. The final troughput is 2/3 packets per channel use.

Figure 4.2:A Wireless physical layer network coding strategy (WPNC or just WNC) for the two-way relay channel that requires two time slots. (a) During the first time slot users communicate with relay and the combination of packets is done naturally and for free in wireless envitonment. (b) During the second time slot, the relay sends the messages after some processing (AF,DF,JDF ...) back to users. Here, the troughput is 1 packet per channel use, and for moreover, inference from the opposite side (here not depicted) source can be useful.

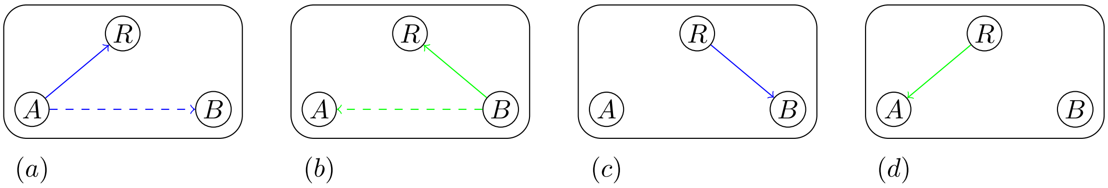


Figure 4.1: Traditional routing strategy

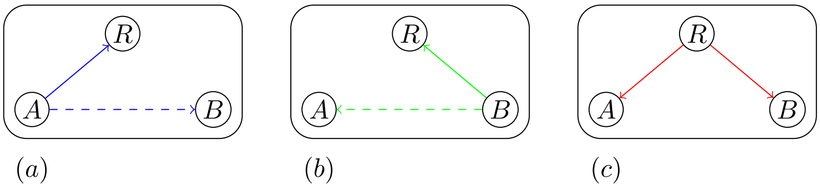


Figure 4.2: A network coding strategy

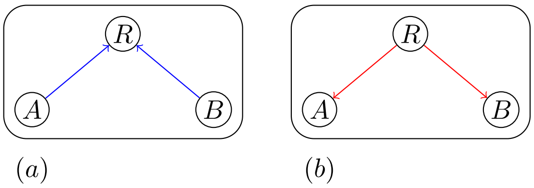


Figure 4.3: A Wireless physical layer network coding strategy

4.2 Basic Therms and Fundamental Principles

4.2.1 Basic therms

For notational clarity, let b (or $\mathbf{b}$) be general notation for code or data symbols (vectors). We will use this in cases, where both is possible.This notation holds for rest of this work.

Definition. Hierarchical MAC stage Multiple interacting relay inputs from multiple sources processed w.r.t. HI

Definition. Hierarchical BC stage Relay broadcasts processed hierarchical information to the next stage.

Having set of all source symbols $\tilde{b} = \{b_A, b_B, \ldots\}$, then complementary set to b_A is denoted as $\tilde{b}_{\bar{A}} = \tilde{b} \setminus b_A$. Now lets define the notation for information content of obsevations.

Definition 4.2.1. Hierarchical Information b is HI w.r.t. desired data b_A iff

$$(b_A; b|\tilde{b}_{\bar{A}}) > 0 \tag{4.1}$$

Definition 4.2.2. Hierarchical (Complementary) Side-Information (C-SI)
$\bar{b}$ is H-SI w.r.t. b_A iff

$$I(b_A; \bar{b}|\tilde{b}_{\bar{A}}) = 0 \text{ and } I(\bar{b}; b_{AB}|b_A) > 0. \tag{4.2}$$

$\bar{b}$ is not HI w.r.t. b_A but affects its HI b trough $\tilde{b}_{\bar{A}}$. In fact, it carriers complementary information $I(\tilde{b}_{\bar{A}}; \bar{A}|b_A) > 0$ and thus H-SI is considered as friendly interference.

Definition 4.2.3. Interference (harmful) β is interference w.r.t. b_A iff

$$I(b_A; \beta|\tilde{b}_{\bar{A}}) = 0 \text{ and } I(\beta; b|b_A) = 0 \tag{4.3}$$

β is not HI neither H-SI w.r.t. b_A.

4.2.2 Relaying Strategies

Under the term of relying strategies, we refer to the method, how relay process the incomming signals, and then forwards to its destinations. Here, we breafly mention the well known methods, and the utilized for purposes of this work will be described later in more details.

Amplify & Forward (AF) It is the simplest method. Relay just amplifies the incomming signal and no more sofisticated processing is done. Although this sollution is really cheap, we pay for bad performance in case of low SNR incomming signal at relay (the noise is amplified as well as signal).

Compress & Forward (CF) Sometimes called *Estimate/Quantize and Forward*, is application of generally nonlinear function to compress the received superimposed signals.

Decode & Forward (DF) This is a name for whole family of strategies. Relay has more degrees of freedom with decisions, that can be employed. These are the most interesting ones.

Joint Decode & Forward (JDF) Relay makes decisions on individual source symbols from received superimposed signal, then decode each of them separately and in last step applies on individual decoded data the combining function (network code). In other words, relay in this strategy tries to convert this situation into the classical 'separated channel per each user' case, which can have in wireless environment significant impact on performance. In order to hierarchically encoded data without error, the rates of each source must be such, that relay is able to reliably decode each of them.

Hierarchical Decode & Forward (HDF) The main difference againts JDF is, that relay tries no more to estimate individual data, but he applies all processing (demodulation $\rightarrow$ decoding $\rightarrow$ encoding $\rightarrow$ modulation) on superimposed codewords, denoted as **hierarchical codewords**. This can offer a capacity gains over JDF, especially in the high SNR regimes.

De-Noise & Forward (DNF) a scheme proposed in [6]. This scheme is very similar to the previous one although in the subsequent works [7],[8] it is mainly focussing on symbol by symbol adaptive relay processing dealing with wireless channel parametrization

Compute & Forward (CmpF) a scheme proposed in [9] that directly processes the PHY superposition of the signals but utilising properties of lattices [10]. A problem of selection of multiplying coefficient is another formulation of the local encoding function selection

4.3 Hierarchical Network Code

Hierarchical Network Code is denoted as a function $\mathcal{X}(\ldots)$, which is utilized by Relay based on HDF strategy. Providing appropriate HNC in previous stages, the destination is able to obtain desired data. The purpose is to map the separate data streams from individual users b_A, b_B to the hierarchical (network-coded) data stream b_{AB}.

$$b_A, d_B : \mathcal{X}(b_A, b_B) = b_{AB} \tag{4.4}$$

But in general, it can be many to one function and there is no restriction on the domain it can be applied on.

In order to guarantee, that the desired information on the given destination can be from HNC fully decodeable, the HNC must fulfill exclusive law

$$\begin{aligned} b_{AB} = \mathcal{X}(b_A, d_B) \neq \mathcal{X}(b'_A, b_B), \ \forall b_A \neq b'_A \\ b_{AB} = \mathcal{X}(b_A, d_B) \neq \mathcal{X}(b_A, b'_B), \ \forall b_B \neq b'_B \end{aligned} \tag{4.5}$$

A code satisfying the above criteria is called a Hierarchical eXclusive Code (HXC).

Then relay hierarchical codebook cardinality must satisfy at least the minimal hierarchical codebook cardinality. Codebook cardinalities are defined in next subsection.

4.3.1 HNC map cardinality

Let's suppose source symbols $b_A \in \mathscr{B}_A$ and $b_B \in \mathscr{B}_B$ and the hierarchical symbol $b_{AB} = \mathscr{B}_{AB}$. Then we define four classes of HNC map.

Definition 4.3.1. Lossy HNC map $|\mathscr{B}_{AB}| < \max(|\mathscr{B}_A|, |\mathscr{B}_B|)$
In situation requirest perfect H-SI and additional HI, or on independent HI

Definition 4.3.2. Minimal HNC map $|\mathscr{B}_{AB}| = \max(|\mathscr{B}_A|, |\mathscr{B}_B|)$
In this case is requirement on perfect H-SI with no additional HI, or on independent HI

Definition 4.3.3. Fully HNC map $|\mathscr{B}_{AB}| = |\mathscr{B}_A| \times |\mathscr{B}_B|$
All pairs are fully decodable at relay and no other H-SI is required

Definition 4.3.4. Extended HNC map $\max(|\mathscr{B}_A|, |\mathscr{B}_B|) < |\mathscr{B}_{AB}| < |\mathscr{B}_A| \times |\mathscr{B}_B|$
All pairs are fully decodable at relay and no other H-SI is required

4.3.2 Linear mapping function

The mapping function can be generally expressed as look- up table with dimension given by the cardinalities of incoming signals. For purposes of this work, we restrict ourselves onlu on linear mapping function. The good example is look-up table of XOR HNC depicted in figure 4.4. It can be seen, that cardinality of both input symbols is four and cardinality of output (look-up table), given by number of different colors, is also four. Thus, from previous subsection is obvious, that the XOR HNC is the minimal one.

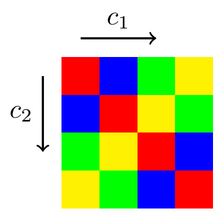


Figure 4.4: XOR HNC look up table

From restriction on linear mapping function also follows, that HNC can be expressed in matrix form:

$$\mathbf{b} = \sum_i \mathbf{X}_{ij} \mathbf{b}_i = \mathbf{X}_j \tilde{\mathbf{b}}. \tag{4.6}$$

To be able to solve this equation on the destination, in order to get the desired data, matrix $\mathbf{X}$ defined on $\mathbf{GF}(M^n)$ must be full rank over GF(M^n).

4.3.3 Layered XOR HNC Design

Layered design is proposep in [?]. The main idea standing behind this is, that under appropriate conditions, the processing of incoming superposed symbols can be divided into two layers, where the first, called as inner layer, has responsibility for adequate HNC map on the input, and the second - outer layer provides the standard error correcting processing like turbo, LDPC ... decoding.
Behind the Layered design stands the following two lemmas.
Lemma 1 (Coding distributes over the exclusive law):
Assume arbitrary linear one-to-one code mappings with a common codebook

$$\mathbf{c}_A = \mathcal{C}, (\mathbf{c}_B)\mathcal{C}(\mathbf{d}_B), \mathbf{c}_{AB}\mathcal{C}(\mathbf{d}_{AB}) \tag{4.7}$$

where $\mathbf{d}_A, \mathbf{d}_B, \mathbf{d}_{AB} \in GF(M^n)$ and $\mathbf{c}_A, \mathbf{c}_B, \mathbf{c}_{AB} \in GF(M^{\tilde{n}})$,$\tilde{n} > n$. Then there exists two minimal exclusive mappings (for data and codewords)

$$\mathbf{d}_{AB} = \mathcal{X}_d(\mathbf{d}_A, \mathbf{d}_B), \mathbf{c}_{AB} = \mathcal{X}_c(\mathbf{c}_A, \mathbf{c}_B) \tag{4.8}$$

such that the following holds

$$\mathcal{C}(\mathcal{X}(\mathbf{d}_A, \mathbf{d}_B)) = \mathcal{X}_c(\mathcal{C}(\mathbf{d}_A), \mathcal{C}(\mathbf{d}_B)) \tag{4.9}$$

Lemma 2 (Exclusive law decomposition over symbols):
Assuming that symbol mapping obeys the exclusive law for each individual symbols, then the exclusive law also hold for complete codeword

$$c_{AB} = c_A, c_B \Longleftrightarrow \mathbf{c}_{AB} = \mathcal{X}_c(\mathbf{c}_A.\mathbf{c}_B) \tag{4.10}$$

4.4 Hierarchical Decode & Forward

For purposes of this work, we are interested in relay hierarchical demodulator output metric μb_{AB} which has to be computed in H-MAC phase. The derivation was originally posted in [3] and we will derive it again step by step, becasuse understanding of the following steps is crucial for our implementation and then possible approximation of this.

4.4.1 Hierarchical soft output metric

The derivation of soft output metric μb_{AB} ($p(x|b_{AB})$) consist from the following steps:

$$\begin{aligned} p(x|b_{AB}) &= p\left(x| \bigcup_{b_A,b_B:\chi_b(b_A,bB)=b_{AB}} b_A, b_B\right) \\ &= \frac{p\left(x \cap \left(\bigcup_{b_A.b_B:\chi_b(b_A,b_B)=b_{AB}} \{b_A, b_B\}\right)\right)}{p\left(\bigcup_{b_A.b_B:\chi_b(b_A,b_B)=b_{AB}} \{b_A, b_B\}\right)}. \end{aligned} \tag{4.11}$$

Pairs b_A, b_B form a partition (disjoint subsets).Then

$$p(x|b_{AB}) = \frac{\sum_{b_A,b_B:\chi_b(b_A,b_B)=b_{AB}} p(x|b_A, b_B)p(b_A, b_B)}{\sum_{b_A,b_B:\chi_b(b_A,b_B)=b_{AB}} p(b_A, b_B)}. \tag{4.12}$$

We can apply Kronecker delta function $\delta[b_{AB} - \chi_b(b_A, b_B)]$ and then summing over all b_A and b_B

$$p(x|b_{AB}) = \frac{\sum_{b_A,b_B} p(x|b_A, b_B)p(b_A, b_B)\delta[b_{AB} - \chi_b(b_A, b_B)]}{\sum_{b_A,b_B} p(b_A, b_B)\delta[b_{AB} - \chi_b(b_A, b_B)]}. \tag{4.13}$$

In the next step, we consider, that $p(b_A, b_B) = \text{const}$.

$$p(x|b_{AB}) = \frac{\sum_{b_A,b_B} p(x|b_A, b_B)\delta[b_{AB} - \chi_b(b_A, b_B)]}{\sum_{b_A,b_B} \delta[b_{AB} - \chi_b(b_A, b_B)]}. \tag{4.14}$$

In case of minimal hierarchical exclusive code, where sum $\sum_{b_A,b_B} \delta[b_{AB}\chi_b-(b_A, b_B)] = M_b$ derivation results in

$$p(x|b_{AB}) = \frac{1}{M_B} \sum_{b_A,b_B} p(x|b_A, b_B)\delta[b_{AB} - \chi_b(b_A, b_B)]. \tag{4.15}$$

And for Gaussian channel

$$p(x|b_{AB}) = \frac{1}{M_B} \sum_{b_A,b_B} p_w(x - ubc_A, b_B))\delta[b_{AB} - \chi_b(b_A, b_B)] \tag{4.16}$$

where u function maps symbols b_a and b_b into hierarchical constellation point and $p_w(w)$ is complex rotationally invariant Gaussian noise

$$p_w(w) = \alpha exp(- \parallel w \parallel^2 / \sigma_w^2) \tag{4.17}$$

We can also make an approximation of this metric by considering only the dominating exponential

$$p(x|c_{ab}) \approx \frac{\alpha}{M_b} exp\Big(- \frac{1}{\sigma_w^2} \parallel x - u_0(b_A, b_B) \parallel^2 \Big). \tag{4.18}$$

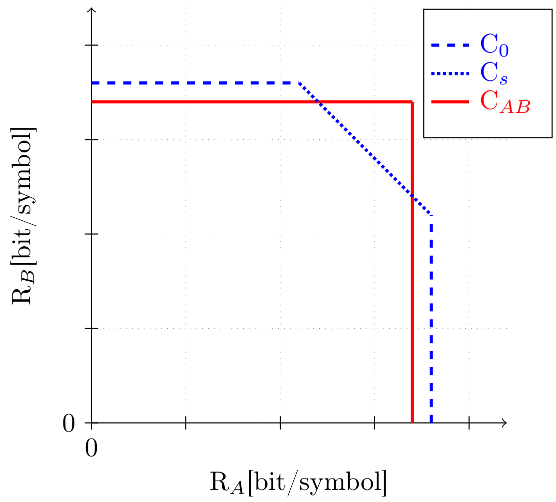


Figure 4.5: Comparision of the MAC capacity egions.

4.4.2 Troughput Rate Region

The main reason for employing HDF relaying strategy is the possible achievement of rectangular capacity region. The situation is depicted in figure 4.5. This serves only for illustration and so we can see only the scale of individual capacities. The legend will be explained in the following text.

From the system model

$$x = u(s(c_A) + hs(c_B)) + w \tag{4.19}$$

where symbol $s(\boldsymbol{.}) \in \mathscr{A}$, code symbol $c \in \mathscr{C}$, u is again the mapping function into hierarchical constellation point and $h \in \mathbb{C}$ is the channel parametrization. The hierarchical mutual information

$$C_{AB} = I(c_{AB}; x) = H[x] - h[x|c_{AB}] \tag{4.20}$$

For the computation of received signal entropy we need p(x), which can be obtained from 4.15

$$p(x_R) = \sum_{c_A, c_B} p(x_R|c_{AB})p(c_{AB}) = \frac{1}{M_c^2} \sum_{c_A, c_B} p(x_R|c_A, c_B) = \frac{1}{M_c^2} \sum_{c_A, c_B} p_w(x - u(c_A, c_B)) \tag{4.21}$$

For comparision, we give constrained first order rate region as limiting performance bound with uniform input alphabet.

$$C_0 = I(c_A; x|c_B) \tag{4.22}$$

and the second order cut-set bound

$$C_s = \frac{1}{2} I(c_A, c_B; x). \tag{4.23}$$

Part II

The thesis contribution

Chapter 5

Thesis contribution

5.1 Motivation

For introduction, let's consider the network depicted in figure 5.1. We have source nodes labeled $S_1, S_2 \dots S_n$ transmitting channel symbols of data or codewords. The node of our interest is *Dest*, which is receiving superimposed signal from this sources marked as *Direct observation*.

Then we have available next observation, the *Side observation*(this can be also in plural), which is superposition of signals of any a subset from a set of $S_1, S_2 \dots S_n$. *Side observation* is orthogonal (transmitted at different time period, different frequency etc.) with regard to *Direct obserwation*.

Dest can be relay performing one of the relay strategies described in subsection 4.2.2, or it can be considered as the final destination. Our goal is to design an universal solution, which will be able to cope with various scenarious.

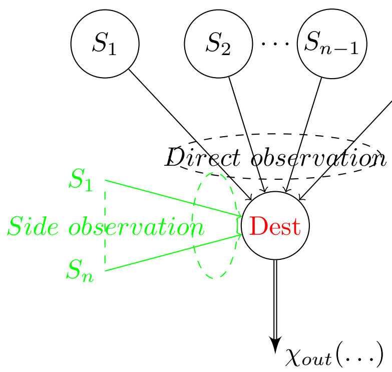


Figure 5.1: Example of network, where node Dest is of our interest

5.2 System model description

The initial situation is depicted in figure 5.2. Now let's suppose, that sources S_1,S_2, $\dots S_n$ produce data vectors $\mathbf{d}_{\bullet}$ from the same alphabet $\mathscr{A}_d\{0, 1, \dots, M_d - 1\}$ with cardinality $|\mathscr{A}_d| = M_d$. Then these data can, but not strictly, be encoded by encoder $\mathscr{C}$ from alphabet $\mathscr{A}_C = \{0, 1, \dots, M_C - 1\}$ which is the same for all sources into codewords with cardinality $|\mathscr{A}_C| = M_C - 1$. It is important to note, that the encoders are identical and thus giving us no degree of freedom in a code rate.

Now, by using the general notation for codeword or data symbols b, we want to express, that we don't strictly employ coder and we send data/code symbols mapped into constallation space. The symbols b are symbol by symbol mapped into the signal space points by common mapper $\mathscr{A}_s$ such that $s_{S_1} = s(b_{S_1})$, $\dots$,$s_{S_n} = s(b_{S_n})$.

The symbols from individual stages are transmitted trough the channels, which are distinquished by colours. Each color could be considered one hierarchical observation and the rest could be perceived as hierarchical side information. But, in more complicated scenarious, this sorting is meaningless, therefore we will mostly call every such observation as Hierarchical Observation (HO). In each "Hierarchical" channel, we consider relative channel parameters **h** with relative relation to one given source. For this thesis $h \in \mathbb{C}$ we restrict ourselves on $|h| = 1$. It is essential to make statement, that vector of channel parameters **h** is perfectly known on the receiver side. The superposition of individual sources channel symbols which can be affected by h is denoted as u.

Per one channel, where is superposition of signals from given sources, is present AWGN $w_\bullet$ with real and imaginary part N $(0,\sigma_\bullet)$. We define SNR $\gamma_\bullet = \frac{\bar{\varepsilon}_{S_\bullet}}{N_{0_\bullet}} = \frac{E[|s_\bullet^2|]}{2\sigma_\bullet^2}$ Each superposition of signals u is affected

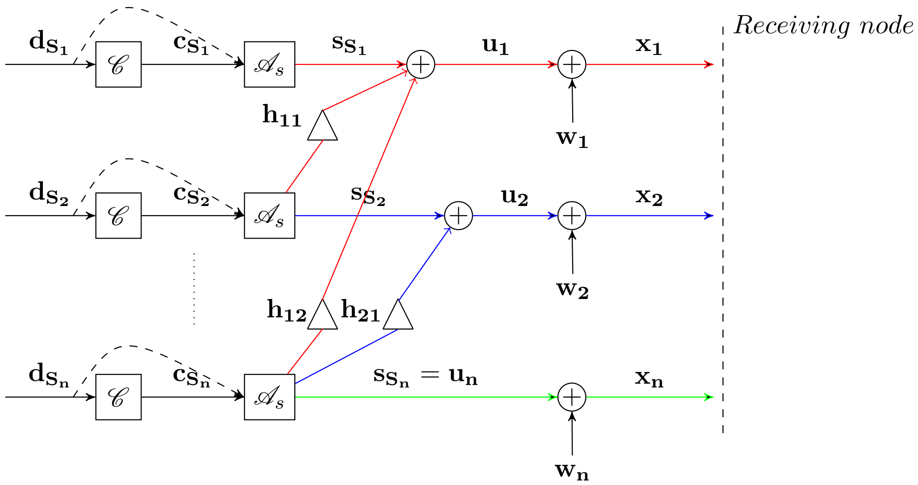


Figure 5.2: General solution of network based on FG

5.3 LDPC Code Implementation

Implementation of the LDPC code was the first step for simulation purposes of this thesis. Since the aim was not a design of LDPC generating matrix **G** and parity check matrix **H**, we used the framework [5].

We used the Gallager regular (3,4) LDPC code with data word length equal to 1000 and the codeword length 1750. Then rate of this code

$$R = \frac{1000}{1750} = 0.5714 \qquad (5.1)$$

The check matrix is generated according to algtorithm originally proposed in [] which is called as Staircase solution. What this means is that code is systematic and thus in our case first 1000 bits of codeword are data and the rest are parity bits.

5.3.1 Decoding on factor graph

We refer on 3.4 where we described mathematical functions according to those we can then make the implementation. Taking the cycles in graph into the consideration, we executed computation of messages according to Flooding algorithm. The major steps of implementation are described in the following algrithms.

Algorithm 1 Soft decision on codeword in probability domain

Input: $\mathbf{H}$ - parity check matrix with c columns and r rows containing n_r number of 1s per each row and n_c number of 1s per each column
$\mathbf{H}_h$ - matrix with dimension $r \times n_r$ including indexes of ones per each row of $\mathbf{H}$
$\mathbf{H}_v$ - matrix with dimension $r \times n_r$ including indexes of ones per each row of $\mathbf{H}$
$\mathbf{x}$ - input probabilities that symbols are in state 0

Output: c-propability of code symbols being in state 0

1: Create matrix $\mathbf{H}_{eq}$ with $\dim(\mathbf{H})$ representing message from equality to check nodes.
2: Create matrix $\mathbf{H}_{check}$ $\dim(\mathbf{H})$ representing message from equality to check nodes with dimension.
3: Update matrix elements of $\mathbf{H}_{eq}$ with indexes according to $\mathbf{H}_v$ providing vector x and matrix $\mathbf{H}_{check}$ according to 3.3.2.
4: Update matrix elements of $\mathbf{H}_{check}$ with indexes according to $\mathbf{H}_h$ providing vector x and matrix $\mathbf{H}_{eq}$ according to **??**
5: Check wether the maximum number of iteration is exceeded or $c\mathbf{H}^T mod 2 = 0$ is satisfied. If yes, stop the computation, else go to step 3.

If we take in consideration, that for computation as described in the previous algorithm we have to store two matrices with size of parity check matrix $\mathbf{H}$, where most of the elements are zeros, and we exactly now, where the zeros are, this solution is computantionally inefficient. But for our purpose, where we are not strictly limited by memory, this is the easiest solution.

5.4 HDF implementation

We will try to explain our approach by providing figure 5.3. We are comming from the fact, that this scheme should be consistent with the layer design described in 4.3.3. At first, we will propose the solution of the part denoted in dashed yellow rectangle. This is nothing else then H-MAC phase. Then we describe the composition of this results into final output metric.

The minimal HNC map is considered as matrix $\mathbf{X}$, based on GF($2^{\mathring{n}}$). The size of matrix $\mathscr{X}_{H_{in_1}}$, which belongs according to the index to function $\mathscr{M}_{\chi_{H_{in_1}}(S_1,\ldots,S_m)}$, is given as $\mathring{n} \times m$.

We have to still keep in mind, that the maps $\mathscr{X}_{H_{in_{\ldots}}}$ has to be solvable with regard to solvability of $\mathscr{X}_{out}$ to get b'.

We will give an example.

Let's have three sources S_1,S_2 and S_3 sending b_1, b_2 and b_3. We want b' be b_1. Then we have available three hierarchical observations with hierarchical maps $\mathscr{X}_{H_{in_\cdot}}$ (the order of elements in maps is $[b_1 \ldots]$) on GF(2).

- $\chi_{out} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$
- $\mathscr{X}_{H_{in_1}} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$
- $\mathscr{X}_{H_{in_2}} = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix}$
- $\mathscr{X}_{H_{in_3}} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$

Due to the linearity, we can write this as the one joint hierarchical map $\mathscr{X}_{in}$

- $\mathscr{X}_{in} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$

- The rank($\mathscr{X}_{in}$)=3, which means, that the map is fully solvable and we can get b'.

The next example:

Let's have three sources S_1,S_2 and S_3 sending b_1, b_2 and b_3. We want b' be b_1. Then we have available two hierarchical observations with hierarchical maps $\mathscr{X}_{S_{in.}}$ on GF(2).

- $\chi_{out} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$
- $\mathscr{X}_{S_{in_1}} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$
- $\mathscr{X}_{S_{in_2}} = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix}$

Due to the linearity, we can write this as the one overall hierarchical map $\mathscr{M}_{\chi_{in}}$

- $\mathscr{X}_{in} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$

- The rank($\mathscr{X}_{in}$)=2, but due to the XOR properties,we are still able to get b'. Ovsem, if we are interested in $\mathscr{X}_{out} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$, then the result is not consistent.

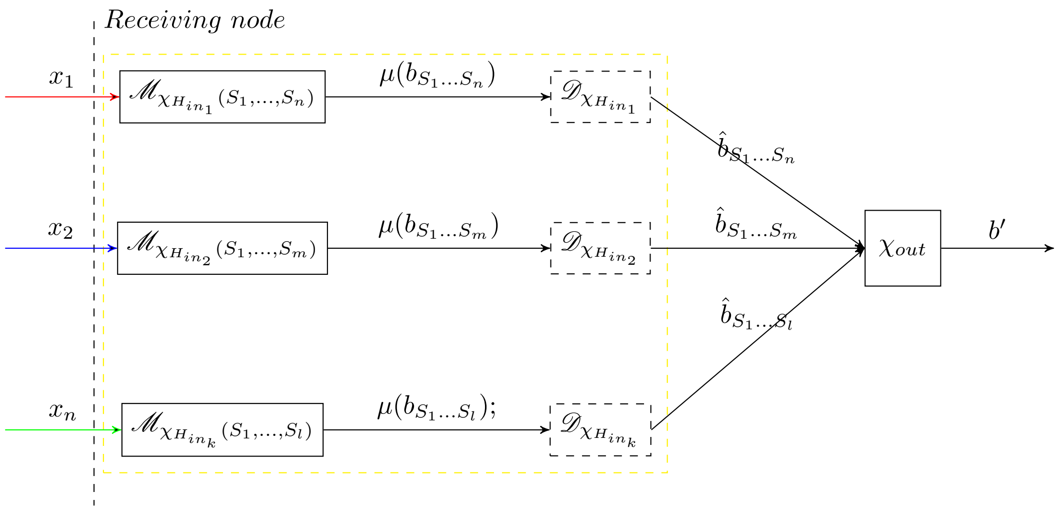


Figure 5.3: Receiver solution consistent with Layer design

5.4.1 H-MAC phase

We should mention again the formula of the soft output metric derived in 4.4.1. To be totally exact, we will consider the most generic applicable form

$$p(x|b_{AB}) = \frac{\sum_{b_A,b_B:\chi_b(b_A,b_B)=b_{AB}} p(x|b_A,b_B)p(b_A,b_B)}{\sum_{b_A,b_B:\chi_b(b_A,bB)=b_{AB}} p(b_A,b_B)} \tag{5.2}$$

For our purposes, we should realize, that this can be generalized for arbitrary number of sources as:

$$p(x|b_{A\ldots B}) = \frac{\sum_{b_A,\ldots,b_B:\chi_b(b_A,\ldots,b_B)=b_{AB}} p(x|b_A,\ldots,b_B)p(b_A,\ldots,b_B)}{\sum_{b_A,\ldots,b_B:\chi_b(b_A,\ldots,bB)=b_{AB}} p(b_A,\ldots,b_B)} \tag{5.3}$$

For notational convenience and for consistency with figure 5.3, we will further denote the probability $p(x|b_{A,\ldots,B})$, which is, of course, the metric $\mu(b_{\ldots})$, as $\mu_{\mathscr{X}_.}$. Giving example, for channel observation given by the blue arrow, we give to the hierarchical metric the appropriate subscript, such that we have $\mu_{\mathscr{X}_2}$.

Algorithm 2 Computation of $\mu(b_{\bullet})$

Input: x - received hierarchical channel symbol
n - number of sources in channel symbol
$\mathbf{h}$ - channel parameters
σ^2 - noise variance
M - order of -PSK modulation
$\mathbf{p}_{\tilde{b}}$ - vector of apriori probabilities of $\tilde{b}$ being in state 0
$\mathbf{X}$ - matrix of HNC map
Output: $\mu(b)$-propability of hierarchical symbol being in state 0

Create matrix $\mathbf{T}$ with number of rows equal to n^M and number of columns equal to $M \times n$, where each row contains unique combination of $M \times n$ bits.
Create vector $\mathbf{u}$ containing all hierarchical points in constallation space, where each hierarchical point is computed according to given row of $\mathbf{M}$ and vector $\mathbf{h}$.
Create vector $\mathbf{m}$ with two elements equal to zero, where the first element represents the $\mu(b_{\bullet} = 0)$ and the second $\mu(b_{\bullet} = 1)$.
for $state = 0$ **to** 1 **do**
  for $i = 1$**to** n^M **do**
    if $(\mathbf{X}\mathbf{M}^T(i,:))mod2 = state$ **then**
      Compute $p_w = exp(\frac{\|x-u(i)\|^2}{\sigma^2})$, where we could neglect the scaling factor, which would be later abbreviated anyway
      $m(state)$=$m(state) + p_w$
    end if
  end for
end for
Finally $\mu(b) = \frac{m(0)}{m(0)+m(1)}$

The general algorithm for computation of metric $\mu_{\mathscr{X}_{\bullet}}$ can be seen in Algorithm 2.

The next option can be an approximation of hierarchical metric. We refer to solution proposed in [4].

Considering the simplest solution, where two sources are participated in MAC phase, then the metric is approximated as

$$p(x|b_{AB}) \approx \frac{\alpha}{M_b} exp\bigg(- \frac{1}{\sigma_w^2} \|x - u_0(b_A, b_B)\|^2 \bigg) \tag{5.4}$$

where $u_0(b_A, b_B)$ is the closest point consistent with b_{AB}. Then the approximation of the metric is called Hierarchical Minimum distance Approximation.

Definition 5.4.1. Hierarchical Minimum Distance

$$\rho^2_{AB,min}(x, b_{AB}) = \min_{b_A.b_B:\mathscr{X}(b_A,b_B)=b_{AB}} \|x - u(b_A, b_B)\|^2 \tag{5.5}$$

The algoritm for computation of this metric is in Algorithm 3.

Algorithm 3 Computation of approximated $\mu(b.)$

Input: x - received hierarchical channel symbol
n - number of sources in channel symbol
$\mathbf{h}$ - channel parameters
σ^2 - noise variance
M - order of -PSK modulation
$\mathbf{X}$ - matrix of HNC map
Output: $\mu(b)$-approximated propability of hierarchical symbol being in state 0

Repeat the steps 1 - 3 from Algorithm 2
Create vector $p_{w_{previous}}(state)$ with two zero elements
for $state = 0$ **to** 1 **do**
  for $i = 1$**to** n^M **do**
    if $(\mathbf{XM}^T(i,:))mod2 = state$ **then**
      Compute $p_w = exp(\frac{\|x-u(i)\|^2}{\sigma^2})$, where we could neglect the scaling factor, which would be later abbreviated anyway
      if $p_w > p_{w_{previous}}(state)$ **then**
        $p_{w_{previous}}(state) = p_w$
      else
        continue
      end if
    end if
  end for
end for
Finally $p_{w_{previous}}(0) = \frac{p_{w_{previous}}(0)}{p_{w_{previous}}(0)+p_{w_{previous}}(1)}$

5.4.2 Output metric

Let us think of an example, where we have two Hierarchical observations x_1 and x_2 and our target is metric $\mu(b')$ of an output hierarchical map. It is again consistent with figure 5.2, but simplified for mathematical convenience in the following derivation. Now, we will try to develop signal processing for this example.

Considering, that the metric $\mu(b')$ is bayes, then

$$\mu(b') = p(x_1, x_2, b') = p(x_1, x_2|b')p(b') \tag{5.6}$$

We can generalize it to:

$$\mu(b') = \sum_{\tilde{b}:b'} p(x_1, x_2, \tilde{b}) = \sum_{\tilde{b}:b'} p(x_1, x_2|\tilde{b})p(\tilde{b}) \tag{5.7}$$

Considering the independency of x_1 and x_2

$$\mu(b') = \sum_{\tilde{b}:b'} p(x_1|\tilde{b})p(x_2|\tilde{b})p(\tilde{b}) \tag{5.8}$$

For solvable minimal hierarchical map must hold:

$$\tilde{b} \equiv \{b, \bar{b}\} \text{ (this notation was introduced in section 4.2.1)} \tag{5.9}$$

and then

$$\mu(b') = \sum_{\tilde{b}:b'} p(x_1|b, \bar{b})p(x_2|b, \bar{b})p(\tilde{b}) \tag{5.10}$$

Now we consider the approximation of hierarchical observation, which is given by the hierarchical metric

$$p(x_1|b, \bar{b}) \cong p(x_1|b) \tag{5.11}$$

and similar for $p(x_2|b,\bar{b})$

$$p(x_2|b,\bar{b}) \cong p(x_2|\bar{b}). \tag{5.12}$$

Example: The simplest example scenario is,

- $\tilde{b} = \{b_A, b_B\}$
- the input hierarchical map: $b = b_A \oplus b_B$
- complementary side information: $\bar{b} = b_B$
- esired output map: $b' = b_A$

$\tilde{b} = \{b_A, b_B\}$, the input hierarchical map $b = b_A \oplus b_B$, complementary side information $\bar{b} = b_B$ and our desired output map is $b' = b_A$, then:

$$\mu(b_A) = \sum_{b_A, b_B : b_A} p(x_1|b_{AB}(b_A, b_B))p(x_2|b_B)p(b_A, b_B) \tag{5.13}$$

$$= \sum_{b_B} p(x_1|b_{AB}(b_A, b_B))p(x_2|b_B)p(b_A, b_B). \tag{5.14}$$

If we expand the summation further :

$$\mu(b_A = 0) = p(x_1|b_{AB=0}(b_A = 0, b_B = 0))p(x_2|b_B = 0)p(b_A = 0, b_B = 0) \tag{5.15}$$

$$+ p(x_1|b_{AB=1}(b_A = 0, b_B = 1))p(x_2|b_B = 1)p(b_A = 0, b_B = 1) \tag{5.16}$$

$$\mu(b_A = 1) = p(x_1|b_{AB=0}(b_A = 1, b_B = 0))p(x_2|b_B = 0)p(b_A = 1, b_B = 0) \tag{5.17}$$

$$+ p(x_1|b_{AB=0}(b_A = 1, b_B = 1))p(x_2|b_B = 1)p(b_A = 1, b_B = 1) \tag{5.18}$$

we can see, that this satisfies the exclusive-or function.

The situation in figure 5.3 should be consistent with Hierarchical Layered HXC design solution

5.5 JDF strategy based on factor graph

The behaviour of system before the receiving node of our interest is again depicted in figure 5.2. The FG is again the Forney Factor Graph model, but we will call it shortly Factor Graph (FG). The notation for variables is the same as in the previous section. For JDF strategy, which is strictly based on Factor Graph, is essential to know, wether we make the hierarchical decoing on the received channel symbols carying pure data, or wether some code structure is present.The main reason is that factor graph has different structure for each inividual case, and we will present both of them. But the fundamental program solution for both examples is identical and we will breafly introduce it in the appendix. Now we will give some important assumptions, which holds for both solutions.

As was proposed in [4], JDF Relay Strategy can be in MAC phase designed in two variants, which differ in the manner of computing the metric of symbols of individual sources.

- Decoders with separate marginalized metric

$$[\hat{d}_A, \hat{d}_B] = \left[arg \max_{d_A} \mu(d_A), arg \max_{d_B} \mu(d_B)\right] \tag{5.19}$$

- Composite hypothesis decoders

$$[\hat{d}_A, \hat{d}_B] = \left[arg \max_{d_A, d_B} \mu(d_A, d_B)\right] \tag{5.20}$$

Now is very important to note, that since our design on factor graph is strictly based on the Sum-Product Algorithm, we are't consistent with none of them. But we suppose, that the second mentioned solution based on composite hypothesis decoders can be considered as the approximation of SPA. Next assumption is, that the HNC is based on GF(2).

5.5.1 Factor Graph of JDF Strategy with uncoded data

We present our design in figure 5.4. This FG consists of standard blocks, which could be in model of standard communication chain. With the red dashed rectangle are highlighted the observation factor nodes of individual HO, with the green one the apriori factor nodes of symbols sent by individual sources, expect the node $p(d)$, which denotes the apriori factor node of the hierarchical symbol. Now we will pay the attention to the factor node denoted as $p(d|d_{S_1}, \ldots, d|d_{S_n})$, which should respect the hierarchical map. In case of minimal HNC map, the factor node $p(d|d_{S_1}, \ldots, d_{S_n})$ is exactly the parity check factor node defined in 3.4.3 with update rule 3.42. The highlighted part with the black dashed rectangle is part, where cycle is present. Thus the computation on factor graph is iterative and blocks $p(x|\tilde{d})$ and $p(y|\tilde{d})$ (the block $p(d|d_{S_1}), \ldots, p(d|d_{S_n})$ in case of minimal HNC map and with no apriori information adds no additional information) exchange the information via the equality factor nodes. We fully described all update rules of all the participating factor nodes in this FFG in subsection 3.2.5.

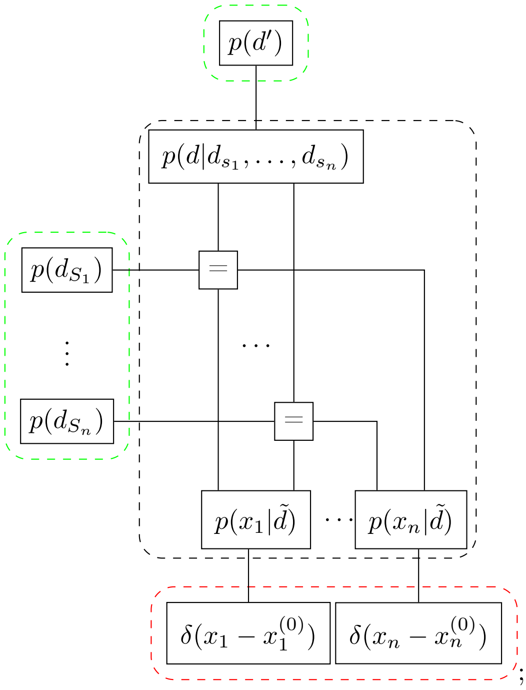


Figure 5.4: Factor graph of JDF strategy

5.5.2 JDF with code structure

Solution is exemplified on the easiest example as possible. We have only one HO available, where codewords $\mathbf{c_A}$ and $\mathbf{c_B}$ are participated. As we made the assumption, that the error correction code is linear systematic block code, the FG is depicted in the figure 5.5.

The decoders interchanges the soft information via the factor node $p(\mathbf{x}|\mathbf{c_A}, \mathbf{c_B})$, and the next iterations are done in decoder factor node $\mathscr{D}_A$ and $\mathscr{D}_B$. In our implemention, per one incoming message from equality to decoder node, several iterations in the decoder are done.

The factor node $\mathscr{X}_{out}(\mathbf{d}_A, \mathbf{d}_B)$ generates the soft information of sequence of hierarchical symbols $d_{AB_1}, \ldots, d_{AB_n}$. In our implementation, we restricted ourselves on minimal HNC on GF(2) and thus $\mu(d_{AB_1})$ is computed according to 3.42 with input messages from the edges $\hat{d}_{A_1}$ and $\hat{d}_{B_1}$. In case of GF($2^{n>2}$), the more complicated structure has to be considered to provide the appropriate mapping. If we take into account, that the decoders $\mathscr{D}_A$ and $\mathscr{D}_B$ are identical, we could also provide the soft information about parity check bits to HNC factor node. In the picture are parity variable nodes highlited by dashed green rectangle and denoted as $c_{...}$, which is not fully consistent with FFG, but for sake of clarity we made this exception. In this case making the conclusion, wether we obtain some benefit is out of scope of this work.

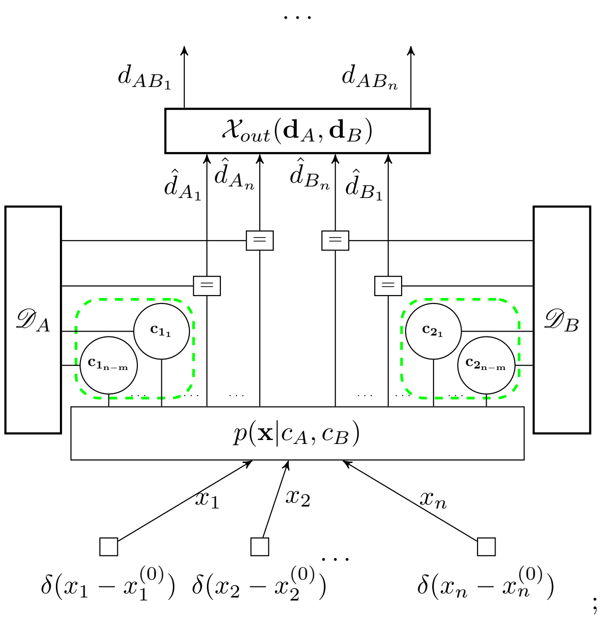


Figure 5.5: Factor graph of JDF strategy with code structure

5.5.3 HDF based on factor graph

This is our last proposed solution, but at the beginning, we should mention, that many problems remains here unsolved and can be considere as subject suitable for scientific research.

Our original idea was to modify the factor graph of JDF strategy as depicted in figure 5.4 in such way, that we add the additional factor node which would be able to respect the metric of hierarchical observation. The resulting FG is in figure 5.6, where for the sake of clarity only one HO is considered. Our presumption was, that the additional factor node denoted as χ_{in} (considering the minimal XOR map) will provide the message (metric) about the hierarchical symbol to factor node $p(x_1|\tilde{d}, d)$, which could lead to better performance. Now we will show the problems standing behind this decision.

Let's have two sources S_1 and S_2 generating data symbols from alphabet $\mathscr{A}_d = \{0, 1\}$, which are mapped into the constellation space from the alphabet $\mathscr{A}_s = \{-1, 1\}$. Now let's have the observation given by equation

$$x = s_{S_1} + hs_{S_2} + w$$

where $|h| = 1$, $arg(h) = 0$ and w is AWGN with zero mean and the scaling factor equal to a. Then the hierarchical constellation space consists of set $x = \{-1, 0, 1\}$. The second element of x causes the uncerainty about the output metric outgoing from factor node $p(x|d_A, d_B)$. But as we will show, respecting the hierarchical symbol in way which is proposed in figure 5.6 gives also ambigous results.

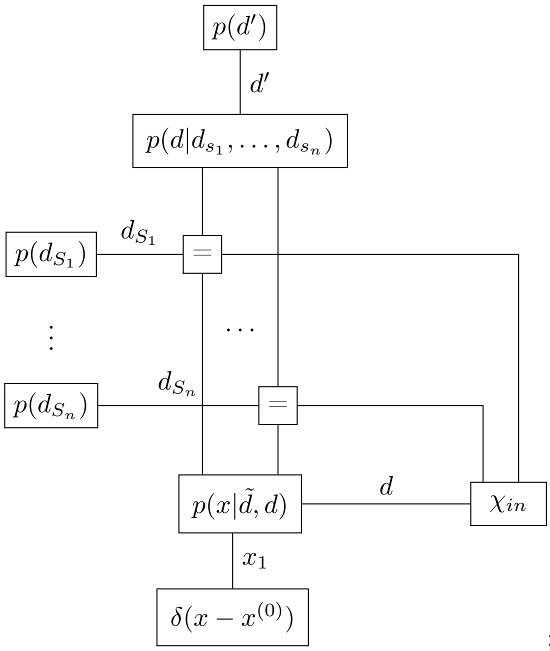
Let's suppose the table 5.1 where are all possible states that can happen, if we try to decode the hierarchical constellation symbol 0.

The first problem arrives when we try to define the red highlited states. This is intractable problem. And even, if we try to ignore this state, this factor graph is not able to solve this problem, because assumption, that we have no apriori probabilities available, it is really easy to show, that even after arbitrary number of iterations the all messages remain in (considering the probability domain) state 0.5.

We will let this problem open for further research.

$p(x_0\|d_1,d_2,d)$	d	d_A	d_B
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
2a	1	0	1
2a	1	1	0
0	1	1	1

Table 5.1: Table of likelohood



;

Figure 5.6: Factor graph of HDF strategy

5.6 Simulation results

The simulation are based on the observation model depicted in figure 5.2, where we restrict ourselves on $\mathscr{A}_s = \{-1, 1\}$(BPSK modulation is used) and $\mathscr{A}_c = \{0, 1\}$.

In all subsections where individual simulation scenarious are presented, BER will be related to hierarchical symbol given by XOR HNC map on GF(2).

5.6.1 HDF - influence of metric approximation on BER

This simulation models the situation, where one hierarchical observation of three sources A, B, C is available, with no relative channel parametrization and $\chi_{in} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$

$$b_{ABC} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_A & b_B & b_C \end{bmatrix}^T \tag{5.21}$$

The figure 5.7 depicts the situation where $s_A = sd_A, \ldots, s_C = sd_C$ and the situation in figure 5.8 is $s_A = sc_A, \ldots, s_C = sc_C$ with error correction.

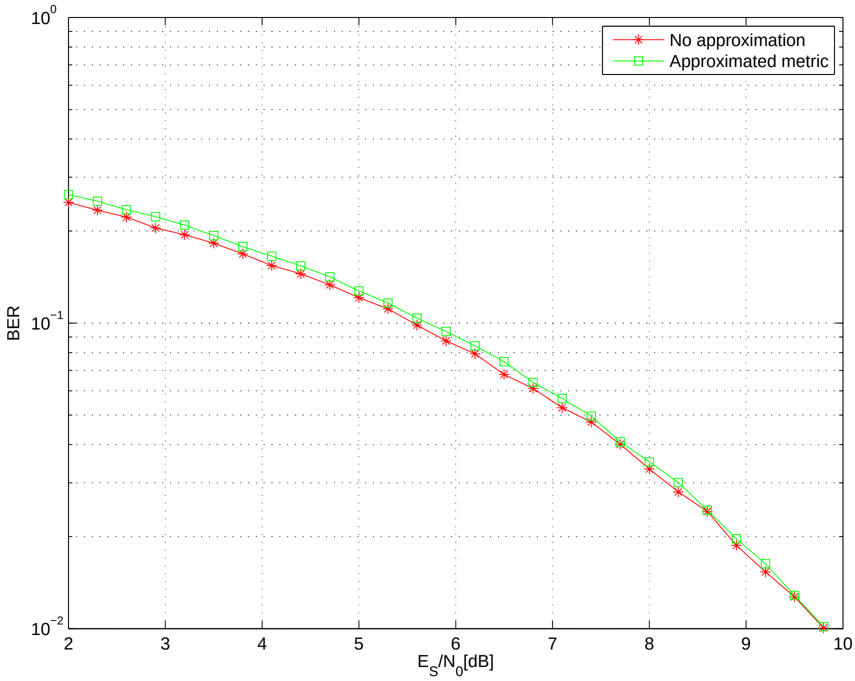


Figure 5.7: Dependence of bit error rate on the computation method of hierarchical metric in case of received data symbols

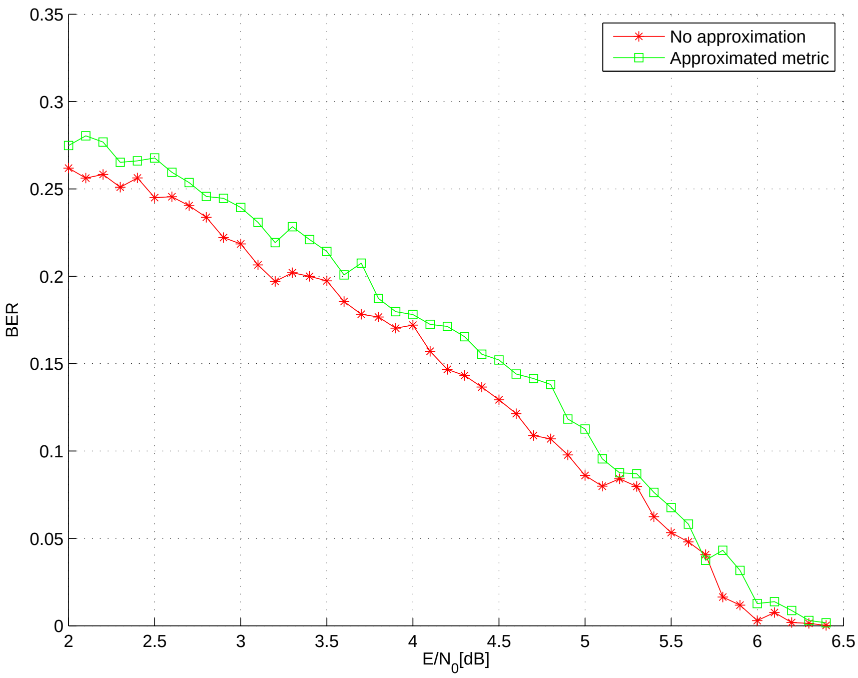


Figure 5.8: Dependence of bit error rate on the computation method of hierarchical metric in case of received code symbols

5.6.2 HDF - influence of available hierarchical observations on BER

The figure 5.9 depicts the situation, where HNC map of available hierarchical observation $\chi_{x_1} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$ and $\chi_{x_2} = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix}$. The output hierarchical map $\chi_{out} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$.

The figure 5.10 depicts the situation, where HNC map of available hierarchical observation $\chi_{x_1} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$ and $\chi_{x_2} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$. The output hierarchical map $\chi_{out} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$.

The output hierarchical map $\chi_{out} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ The hierarchical maps are related to data symbols $\begin{bmatrix} d_A & d_B & d_C \end{bmatrix}$.

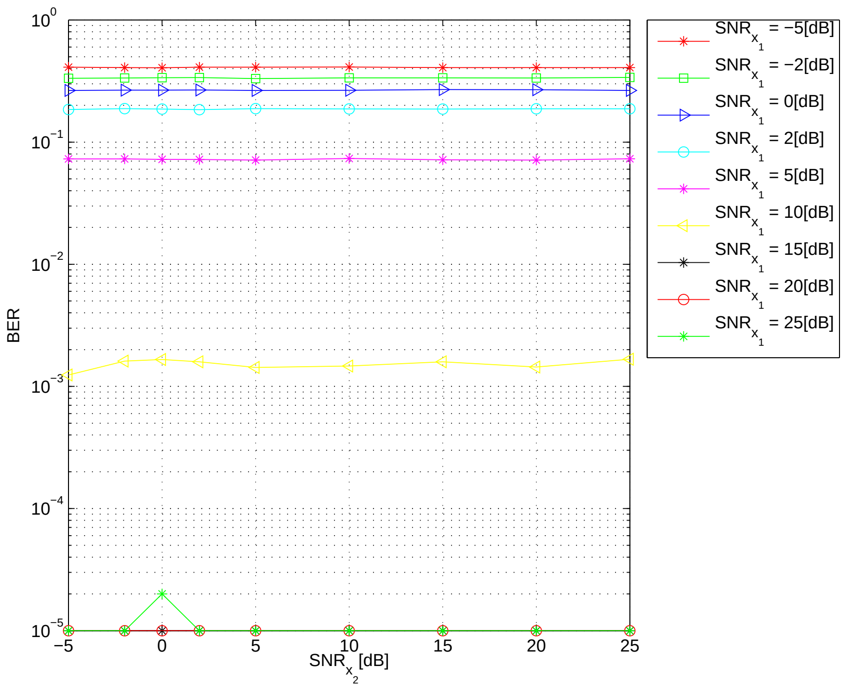


Figure 5.9: Hierarchical observations providing solvable hierarchical output map

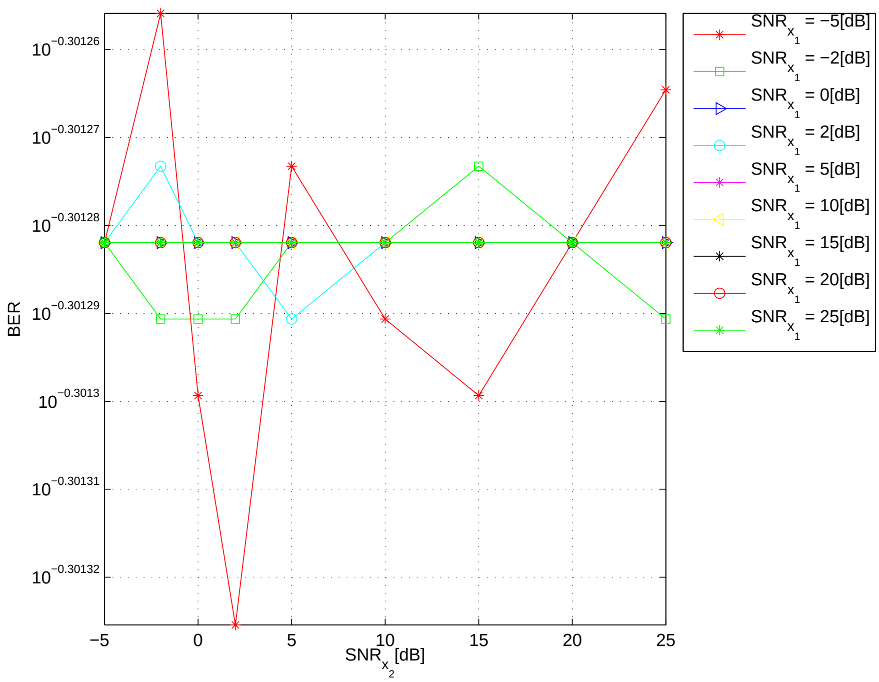


Figure 5.10: Insufficient number of hierarchical observations leading to insolvable hierarchical output map

5.6.3 HDF - BER of hierarchical map of coded/uncoded data words

The figure 5.11 depicts the dependency of SNR of individual hierarchical observations with no error correction with comparision to figure 5.12 where error correction is considered.
$\chi_{x_1} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$, $\chi_{x_2} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$ and $\chi_{x_3} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$.

The output hierarchical map $\chi_{out} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ and SNR_{x_1}=10 dB

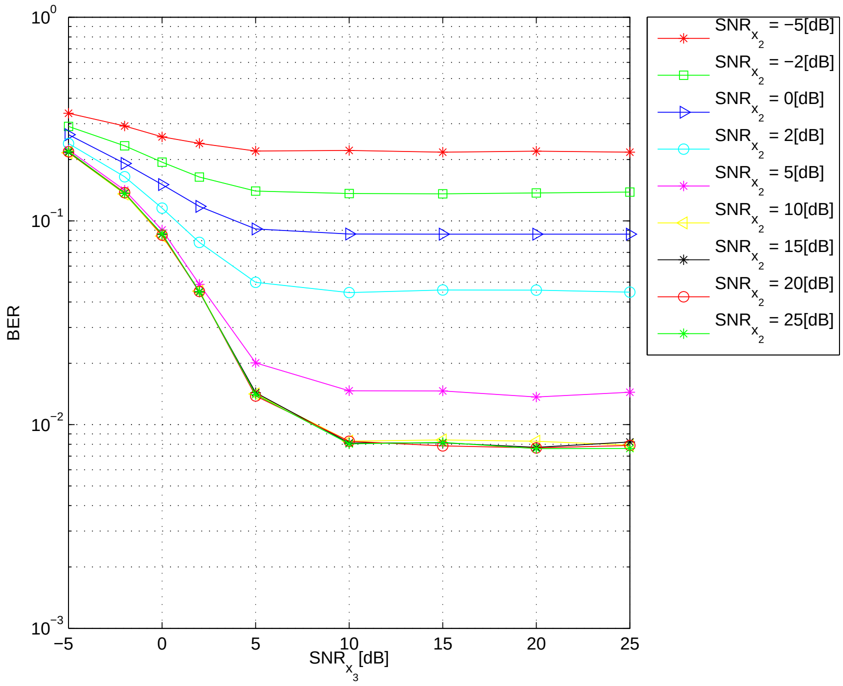


Figure 5.11: Dependency of BER on SNR of individual hierarchical observations without error correction code

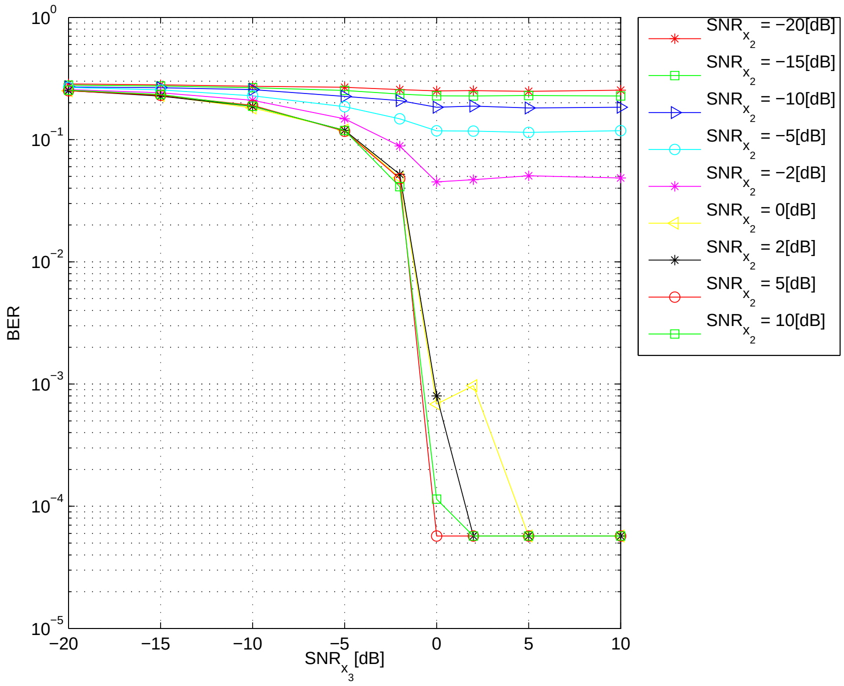


Figure 5.12: Dependency of BER on SNR of individual hierarchical observations with error correction code

5.6.4 JDF - BER with different phases - coded/uncoded data words

One hierarchical observation is observed with three data sources. The phase of the third source is fixed $arg(h_1) = 0deg$. The figure 5.13 depicts the dependency of BER of the source x_1 on phase of the others (distinguished with subscript 1 and 2), where SNR=0dB and error correction is considered.

The figure 5.14 depicts the same situation, but SNR=10dB and no error correction is done.

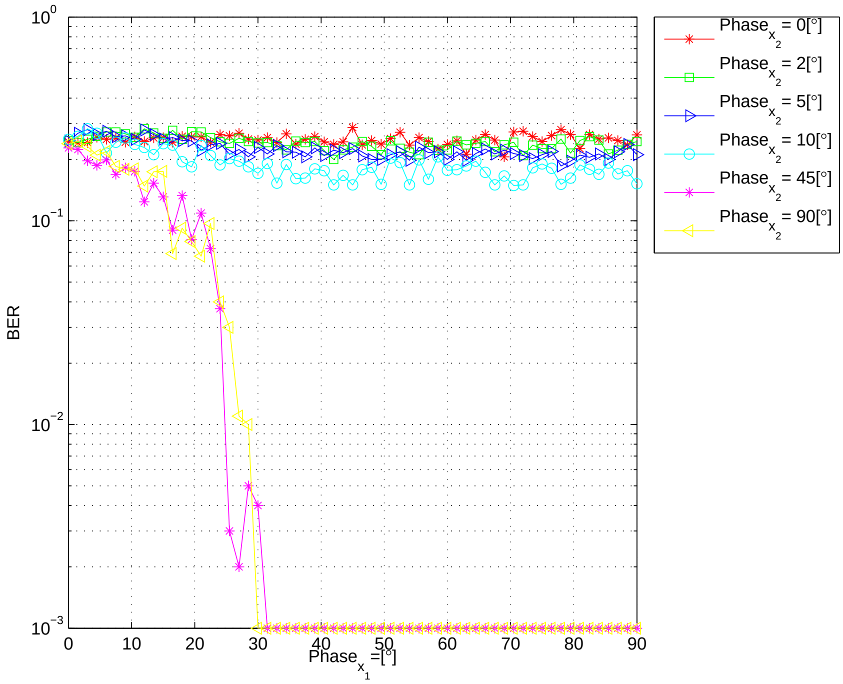


Figure 5.13: Phase dependecy of individual users with error correction code

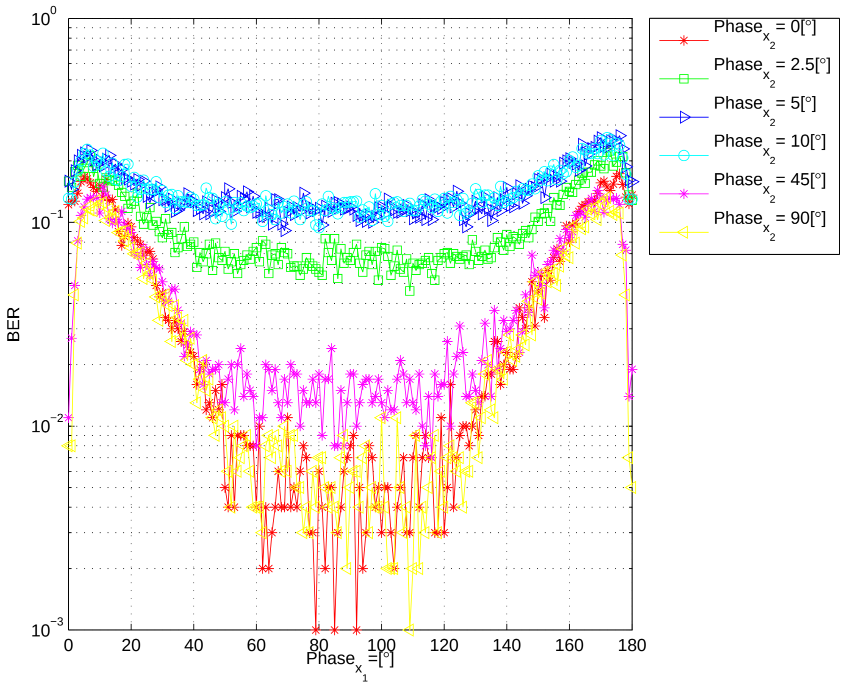


Figure 5.14: Phase dependecy of individual users without error correction code

5.6.5 HDF vs JDF - BER

The figure 5.15 gives the comparision of BER performence of JDF and HDF with no error correction code. We have only one hierarchical observation, $\chi_{x_1} = \left[\begin{array}{ccc} 1 & 1 & 1 \end{array}\right]$ and the channel parameters $arg(h_{1_1}) = 0$, $arg(h_{1_2}) = 45deg$ and $arg(h_{1_3}) = 90deg$.

The output hierarchical map $\chi_{out} = \left[\begin{array}{ccc} 1 & 0 & 0 \end{array}\right]$

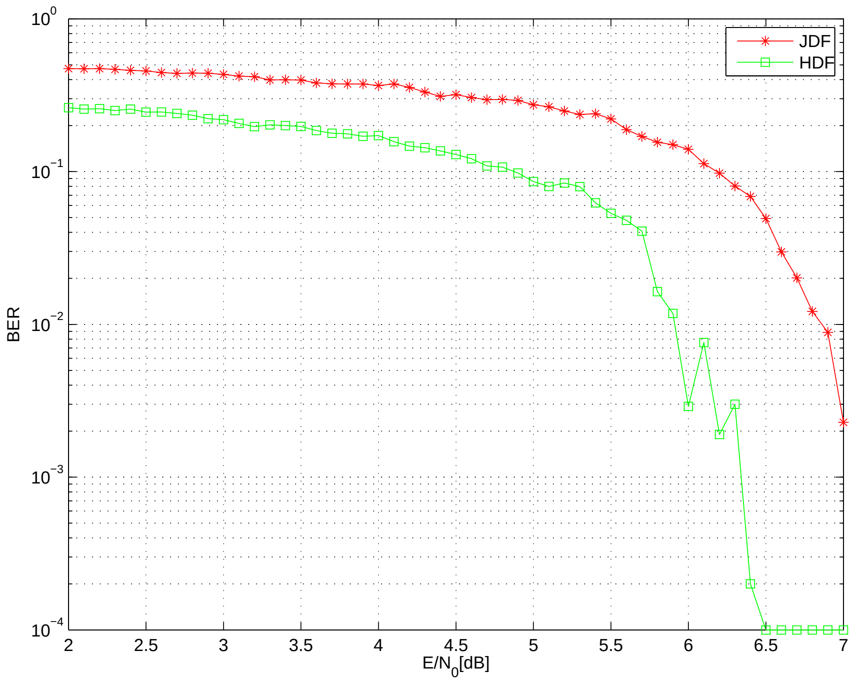


Figure 5.15: JDF vs HDF

Chapter 6

Conclusion

I have get acquainted with Wireless Physical Layer Network Coding and with the principles of iterative soft-information based decoding on Factor Graph, where the Sum Product Algorithm was applied. Then I used it to design block, which would be able to solve various scenarious of available hierarchical observations, in order to get the desired information for further processing. Then I made several simulations to show the performance of designed blocks, depending on SNR and phase rotation, which has harmful impact on the hierarchical observations.

Significant discovery was, that the solution on Factor Graph which would be consistent with Hierarchical Decode and Forward decoding strategy from Sum Product Algorithm point of view cannot be straightforward implemented, thus it can be addressed as interesting subject on the future work.

Bibliography

[1] F.Kschischang, B.Frey and H.-A. Loeliger, Factor graphs and the sum-product algorithm, *IEEE Trans. Inf. Theory,* 47(2):489-519, 2001.

[2] Tuan Ta, A Tutorial on Low Density Parity-Check Codes, *The University of Texas at Austrin*

[3] Jan Sykora and Alister Burr. Layered design of hierarchical exclusive codebook and its capacity regions for HDF strategy in parametric wireless 2-WRC. *IEEE Trans. Veh. Technol.,* 60(7):3241-3252, September 2011.

[4] Jan Sykora and Alister Burr. Advances in wireless network coding - the future of cloud communications. In *Proc. IEEE Int. Symp. on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1-190, London, UK, September 2013. tutorial.

[5] Růžička, L. (2014). Implementation and performance evaluation framework for LDPC codes. Unpublished bachelor thesis. CTU in Prag

[6] P. Popovski and H. Yomo, "Physical network coding in two-way relay channels," in *Communications, 2007. ICC'07.IEEE International Conference on,* 2007, p. 707-712.

[7] T. Koike-Akino, P. Popovski, and V. Tarokh, "Optimized constellations for two-way wireless relaying with physical network coding," *Selected Areas in Communications, IEEE Journal on*, vol. 27, no. 5, pp. 773-787 2009.

[8] T. Koike-Akino, P. Popovski and V. Tarokh, "Denoising maps and constellations for wireless network coding in two-way relaying systems," in *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE,* 2008, pp. 1-5.

[9] M. Nazer and M. Gastpar, "Compute-and-forward: Harnessing interference trough structured codes," *Information theory, IEEE Transactions on,* vol.57, no. 10, pp. 6463-6486, Oct. 2011.

[10] R.Zamir, "Lattices are everywhere," in *Information theory and applications Workshop, 2009,* 2009, pp. 392-421.