Design Of Ultra-High Throughput Rate NB-LDPC Decoder

Prepared by *Hassan HARB*

11/07/2018

Supervisors:

Prof. Emmanuel Boutillon

Dr. Ali Chamas Al Ghouwayel

Dr. Laura Conde-Canencia

Prof. Ali Alaeddine

Jury:

Prof. Olivier Berder (President)

Prof. Andreas Burg (Reviewer)

Prof. Zhengya Zhang (Reviewer)

Prof. Catherine Douillard

Guest:

Dr. Bertrand Le Gal

Principle of Channel Coding

Redundancy

Source

K bits

Channel encoding

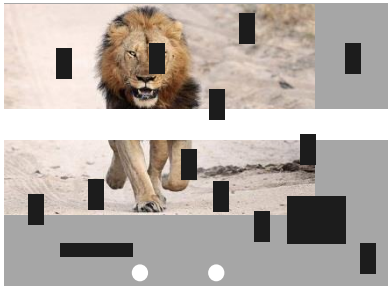
N bits

Channel (noise)

K bits

Channel decoding

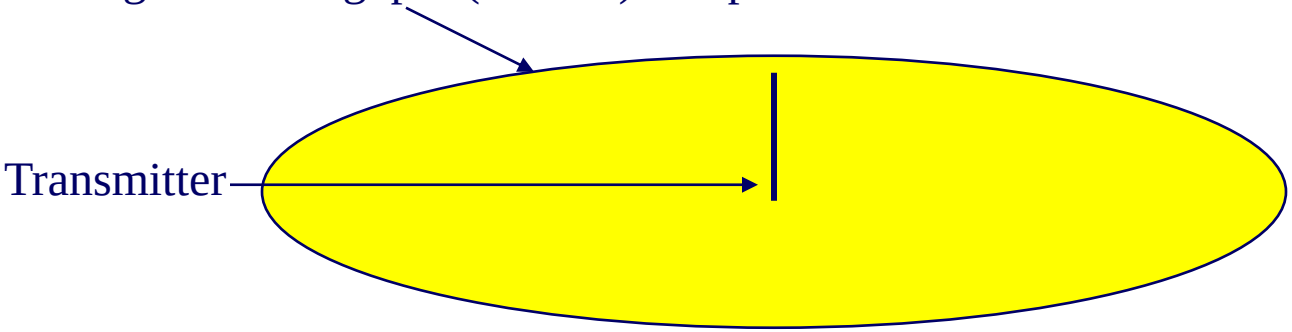
N noisy samples



Rate of the code CR = *K*/*N*

A very short history of channel coding

Prediction of Shannon (1950):
potential “error free communication area”
for a given throughput (bits/Hz) and power emission.

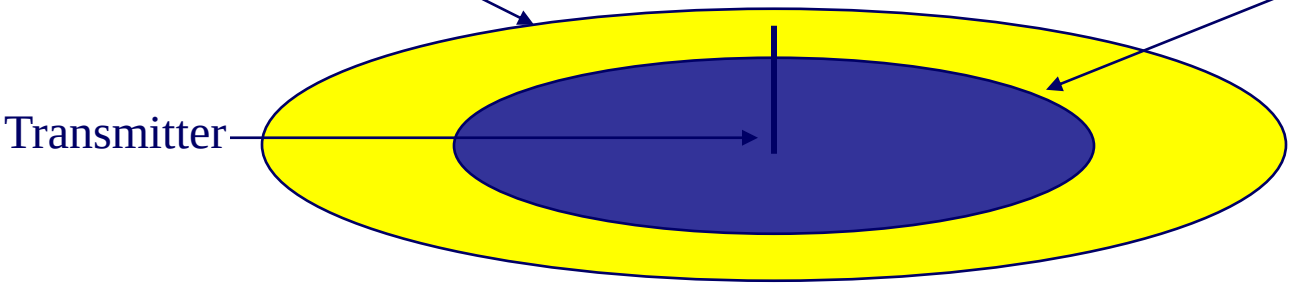


BPSK: Binary Phase Shift Keying

A very short history of channel coding

Prediction of Shannon (1950):
potential "error free communication area"
for a given throughput (bits/Hz) and power emission.

Situation in 1990
Real "quasi error free" area

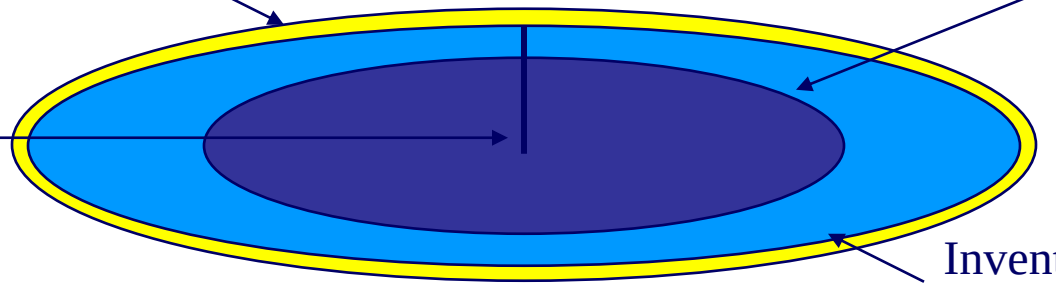


BPSK: Binary Phase Shift Keying

A very short history of channel coding

Prediction of Shannon (1950):
potential “error free communication area”
for a given throughput (bits/Hz) and power emission.

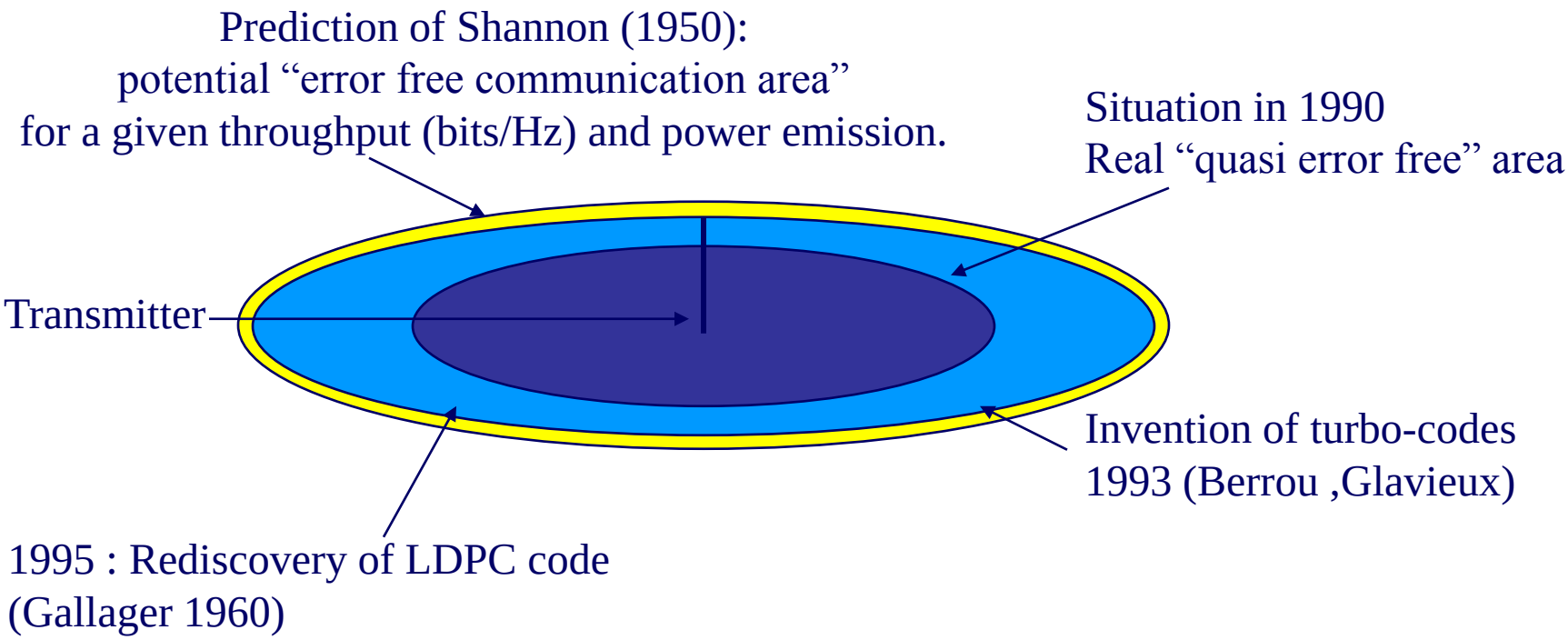
Situation in 1990
Real “quasi error free” area



Invention of turbo-codes
1993 (Berrou ,Glavieux)

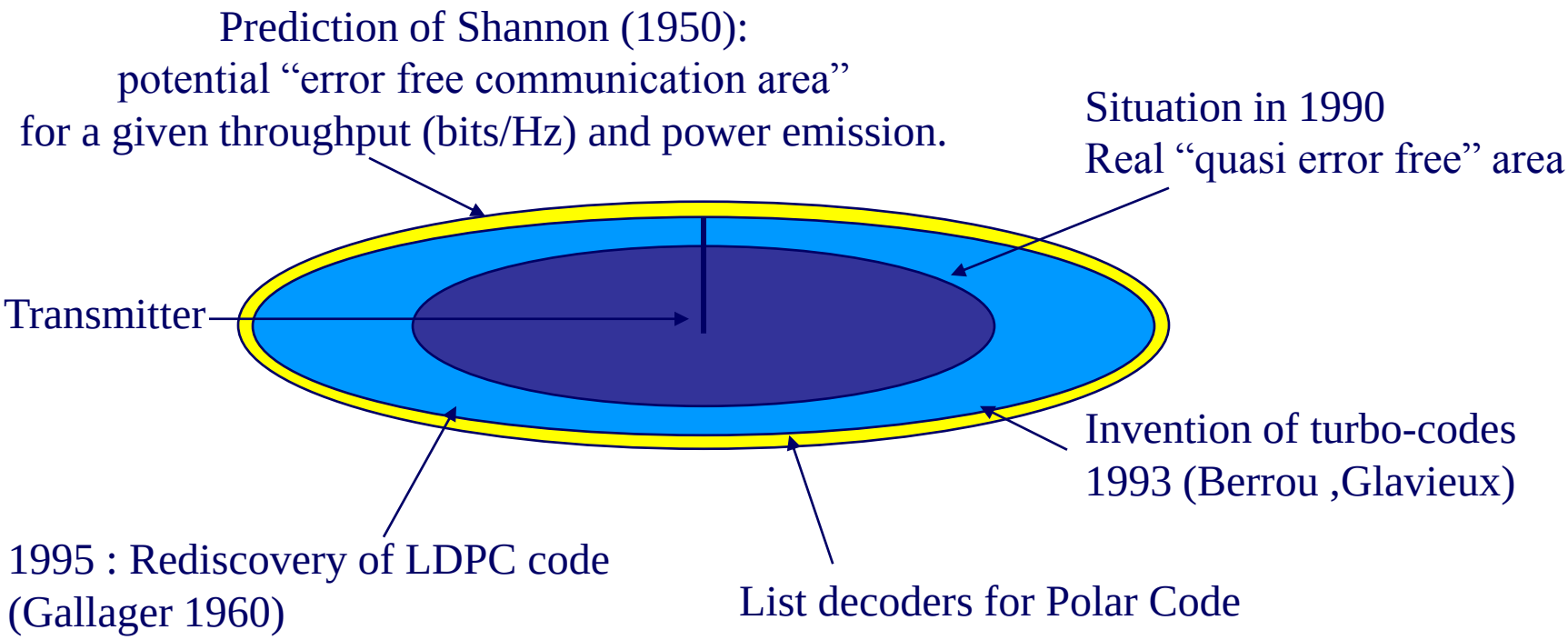
BPSK: Binary Phase Shift Keying

A very short history of channel coding



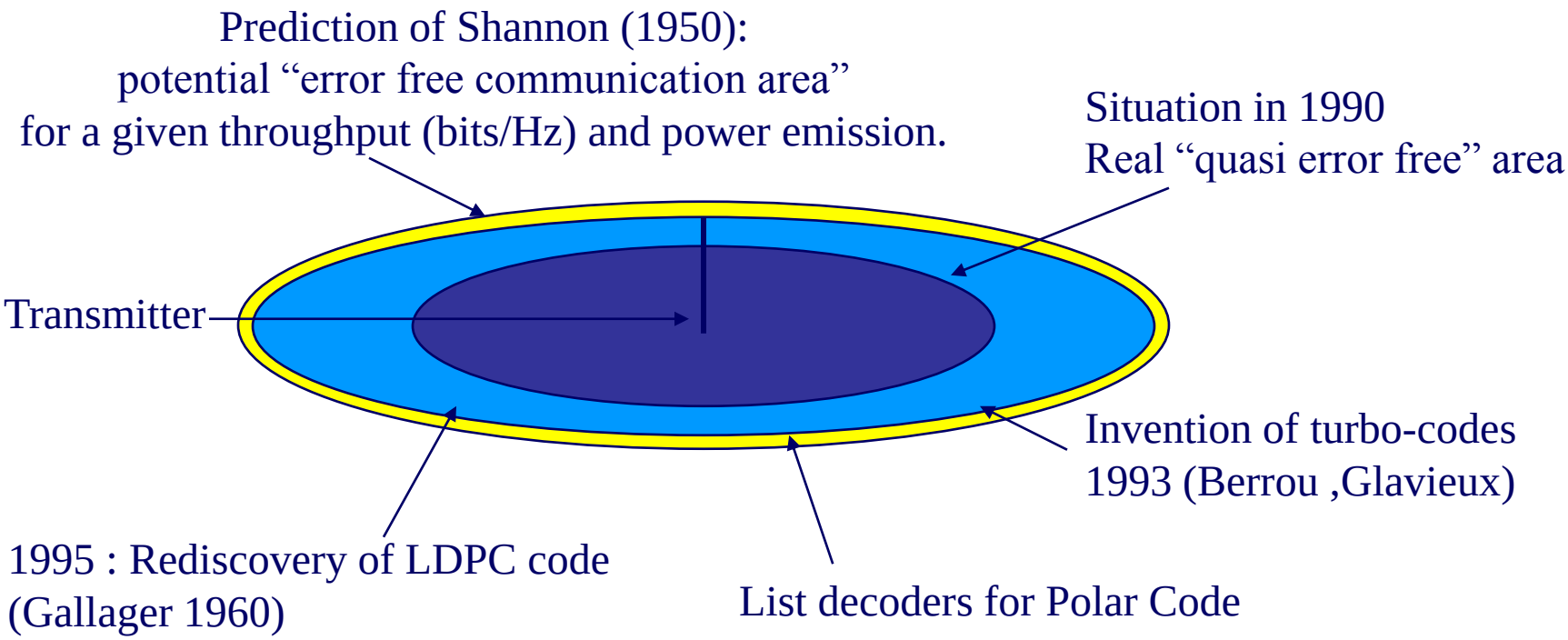
BPSK: Binary Phase Shift Keying

A very short history of channel coding



BPSK: Binary Phase Shift Keying

A very short history of channel coding

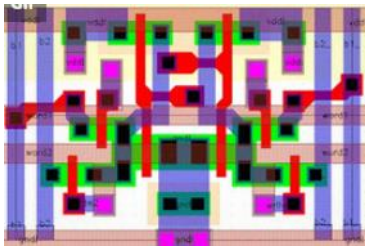


Binary codes very efficient for Gaussian channel, BPSK modulation

BPSK: Binary Phase Shift Keying

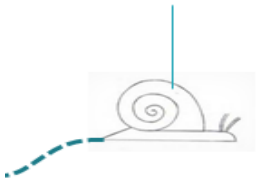
Where NB-LDPC codes can be efficient

NB-LDPC code in CCSDS standard (space communication)

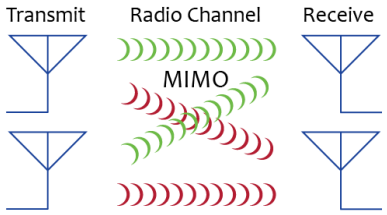


© iis-people.ee.ethz.ch

NB-LDPC for "no error floor" application: Memory



Small packet for IoT at SNR around -15 dB
CCSK + NB-LDPC



Potential gain of 1.5 db with 2x2 MIMO system around 4 bits/s/hertz

But NB-LDPC remains limited in practical applications due to high decoding complexity!

PhD Objective: Low complexity and high throughput NB-LDPC decoder

Outline

- Introduction
- NB-LDPC Codes
- EMS Algorithm and Architectures
- Proposed Parallel and Pipelined NB-LDPC Decoder Architecture
- Extra Related Works
- Conclusion, Perspectives and Publications

Outline

- Introduction
- NB-LDPC Codes
  - Galois Field
  - Definition
  - Decoding Algorithms
- EMS Algorithm and Architecture
- Proposed Parallel and Pipelined NB-LDPC Decoder Architecture
- Extra Related Works
- Conclusion, Perspectives and Publications

Galois Field

Galois Filed of characteristic q (GF(q)) is a finite field that contains q elements. All the operations (+,*,-, /) are performed "modulo q", where q is a power of prime number.

Example: m=3, q=2^m=8, GF(q=8)

GF element	bit representation
0	000
α^0	100
α^1	010
α^2	001
α^3	110
α^4	011
α^5	111
α^6	101

Addition: $X = (x_0x_1x_2)$, $Y = (y_0y_1y_2) \in$ GF(8) => $X \oplus Y = X$ XOR Y

Example: $\alpha^4 \oplus \alpha^1 = 011$ XOR $010 = 001 = \alpha^2$

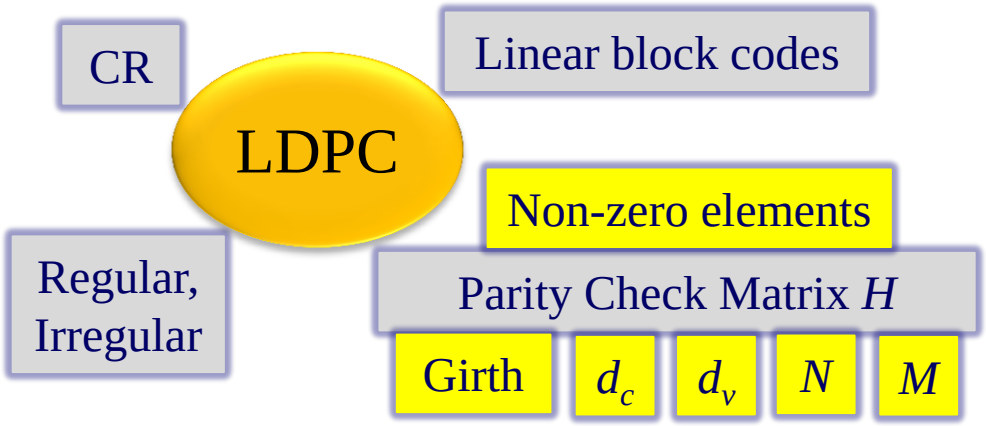
Multiplication:

$$0.\alpha^i = 0$$

$$\alpha^i.\alpha^j = \alpha^{(i+j)\bmod(q-1)}$$

Example: $\alpha^4.\alpha^3 = \alpha^{7\bmod(7)} = \alpha^0$

Definition



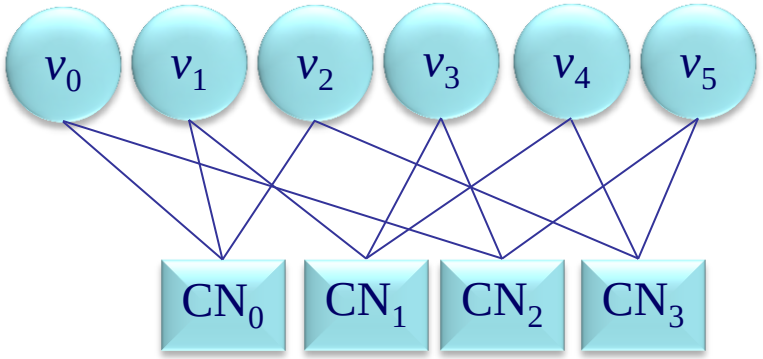
Example: M=4, N=6, d_c=3 and d_v=2

$$\begin{array}{c} \\ CN_0 \\ CN_1 \\ CN_2 \\ CN_3 \end{array} \begin{array}{c} \begin{array}{cccccc} v_0 & v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix} \end{array}$$

- h_{ij} belong to:
  - GF(2): B-LDPC
  - GF($q=2^m$), $m > 1$, NB-LDPC

$$v_0.h_{0,0} \oplus v_1.h_{0,1} \oplus v_2.h_{0,2} = 0.$$
$$v_1.h_{1,1} \oplus v_3.h_{1,3} \oplus v_4.h_{1,4} = 0.$$
$$v_0.h_{2,0} \oplus v_3.h_{2,3} \oplus v_5.h_{2,5} = 0.$$
$$v_2.h_{3,2} \oplus v_4.h_{3,4} \oplus v_5.h_{3,5} = 0.$$

$\oplus$: GF addition



Tanner graph [2]

#: Number of
VN: Variable Node
CN: Check Node
N: Code-length, # columns (VNs)
M: # rows (CNs)
d_c: # non-zero elements per row
d_v: # non-zero elements per column
CR=K/N: Code Rate
B: Binary
NB: Non-Binary
GF: Galois Field

[1] R. Gallager, 1963.

[2] R.M. Tanner. 1981.

Definition: Iterative decoding process

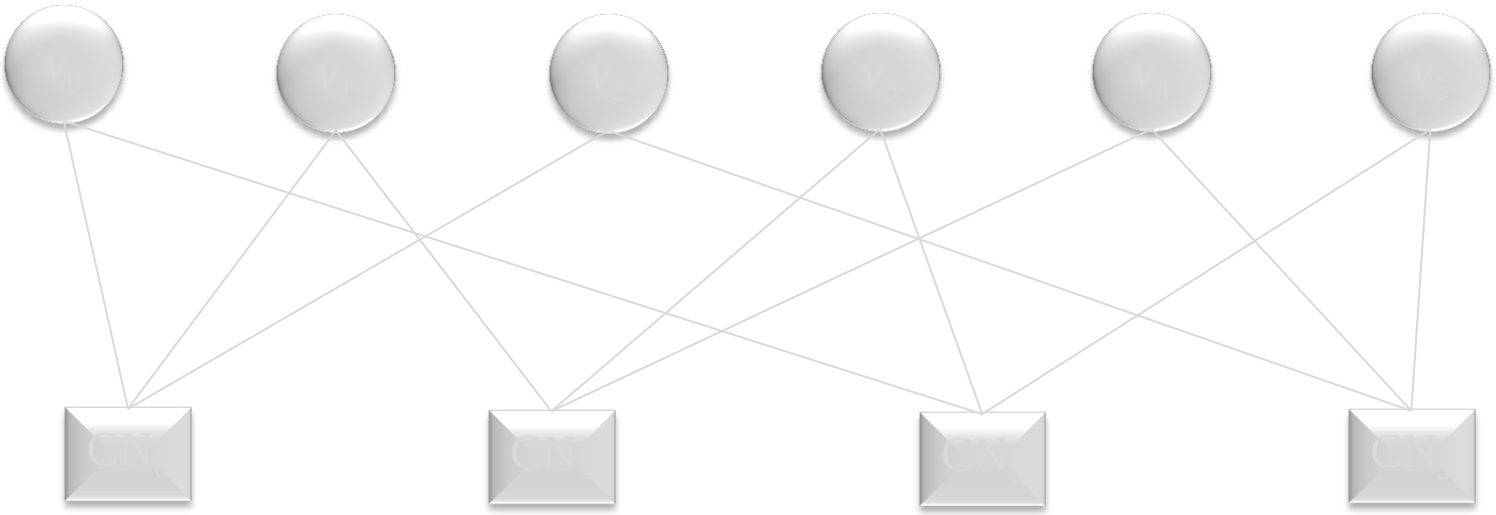
Transmitted CW

decoded CW

Received CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

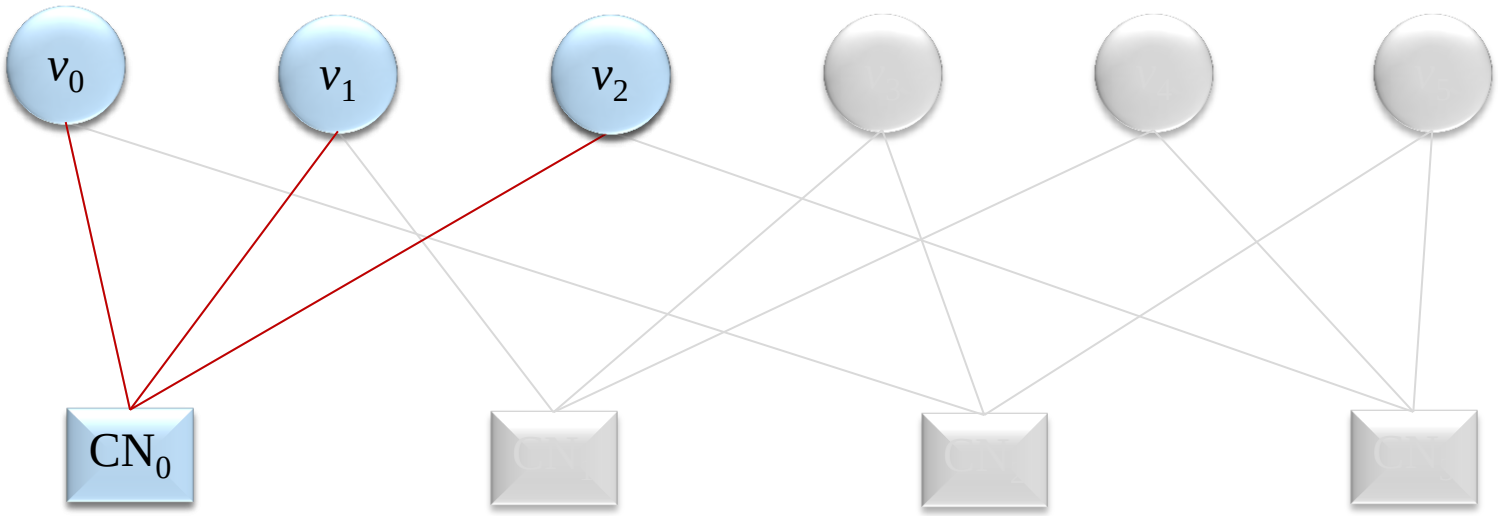
Transmitted CW

decoded CW

Received CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

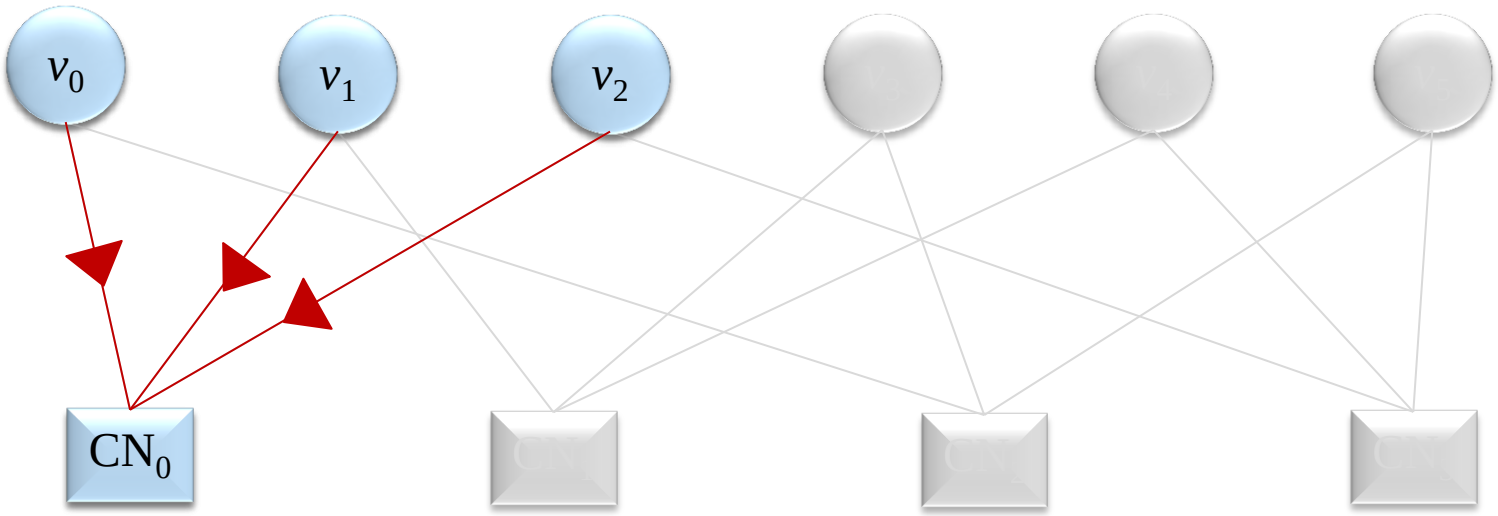
Transmitted CW

decoded CW

Received CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

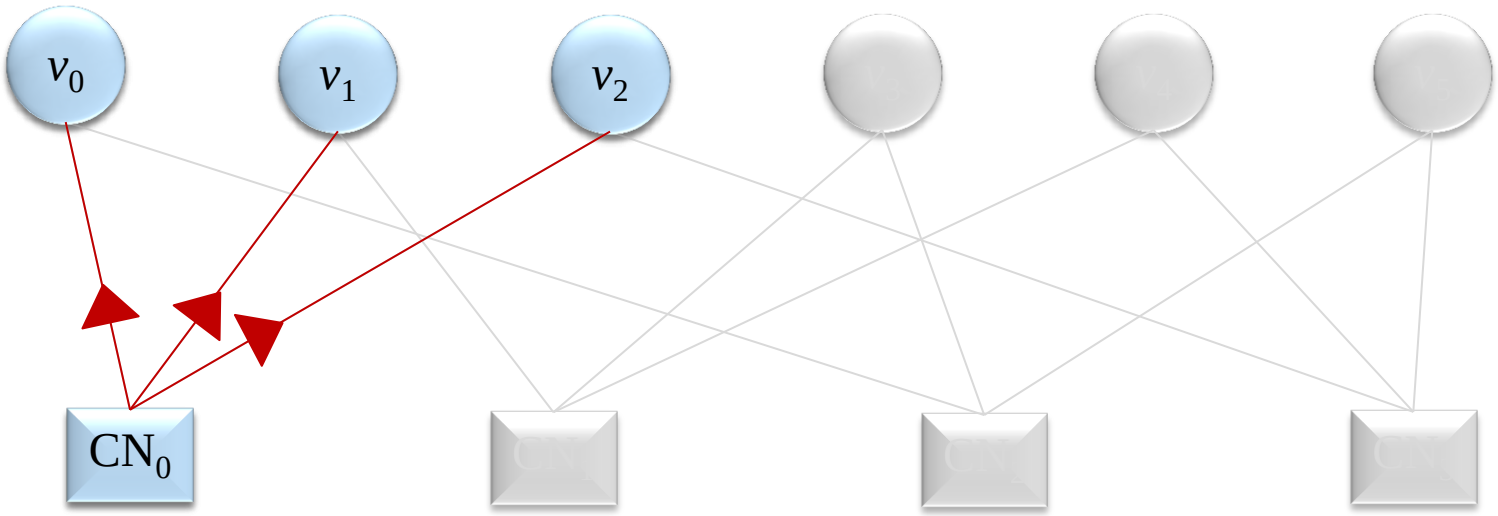
Transmitted CW

decoded CW

Received CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

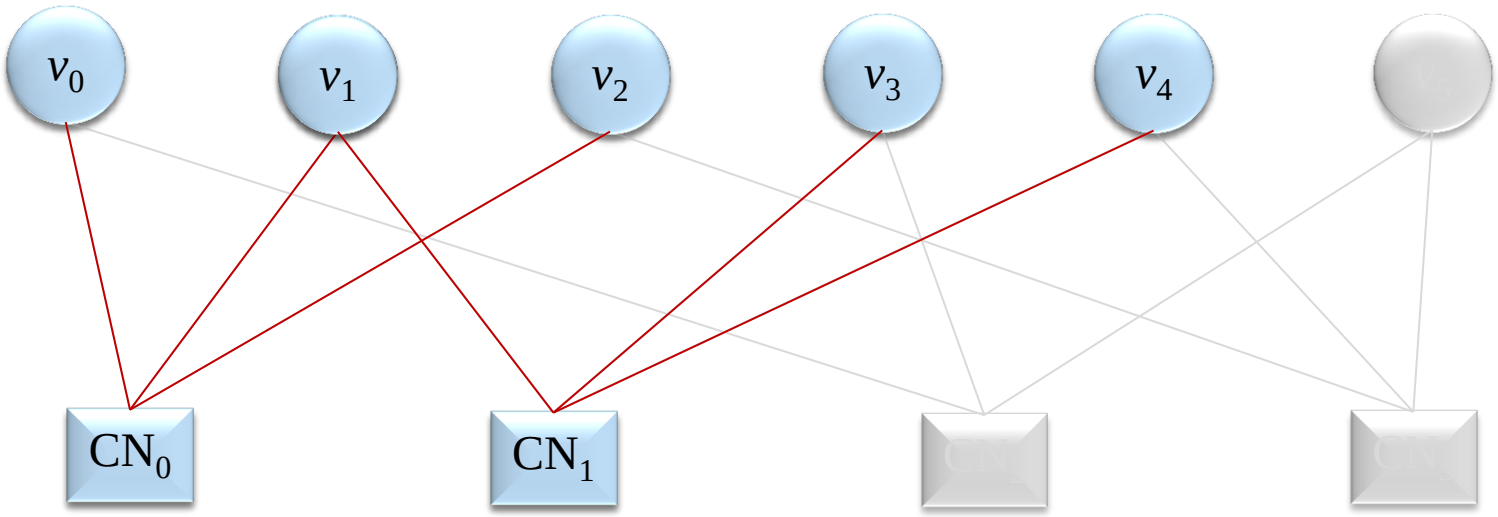
Transmitted CW

decoded CW

Received CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

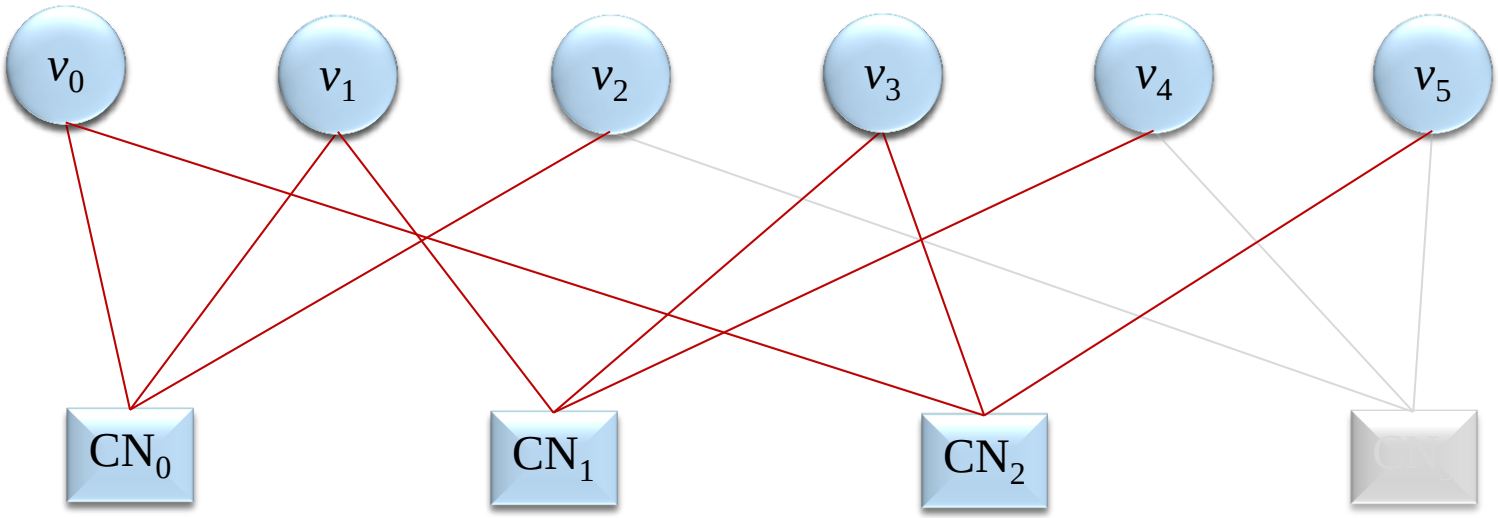
Transmitted CW

decoded CW

Received CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Lab-STICC

cnrs

Definition: Iterative decoding process

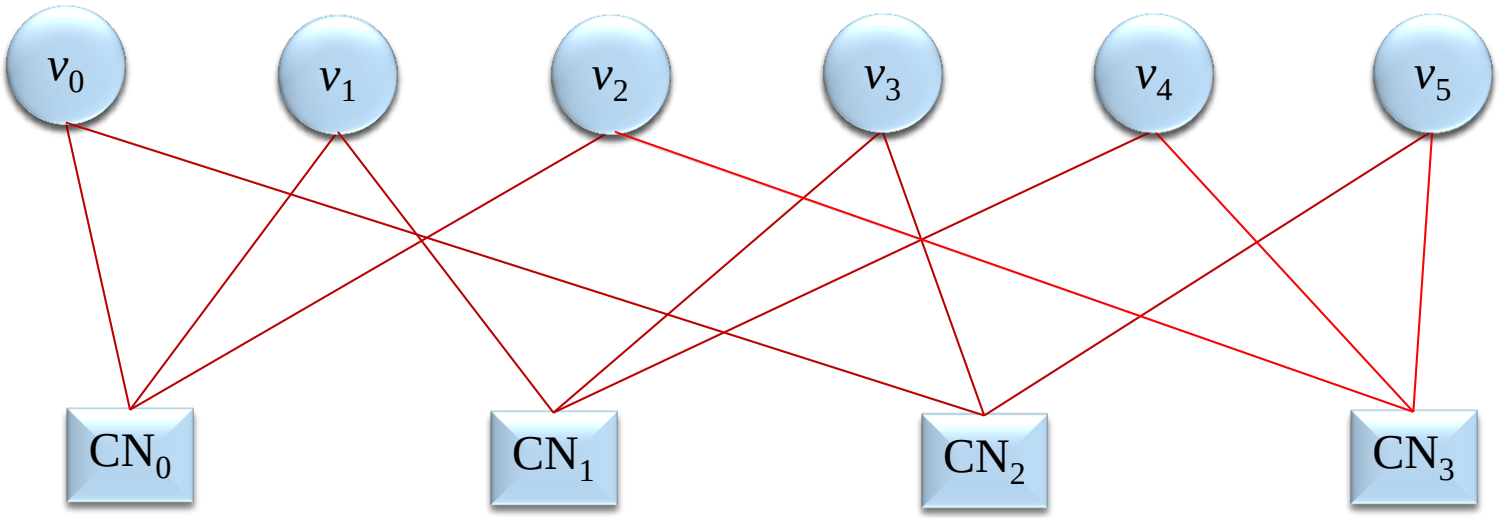
Transmitted CW

decoded CW

Received CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

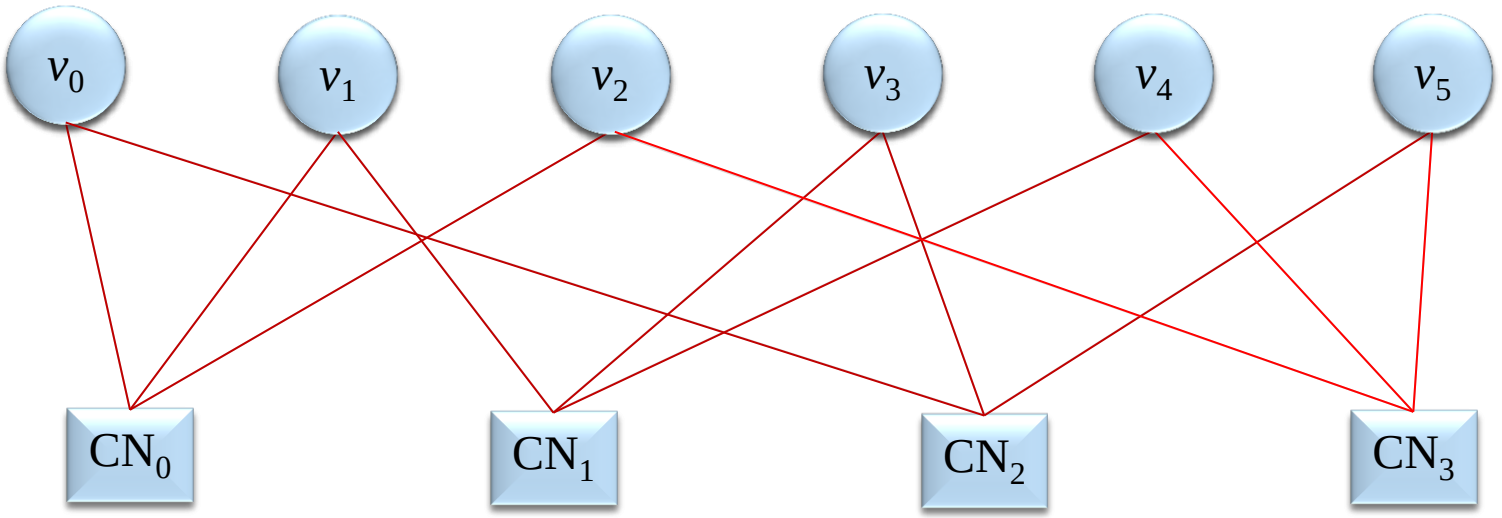
Transmitted CW

Received CW

decoded CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

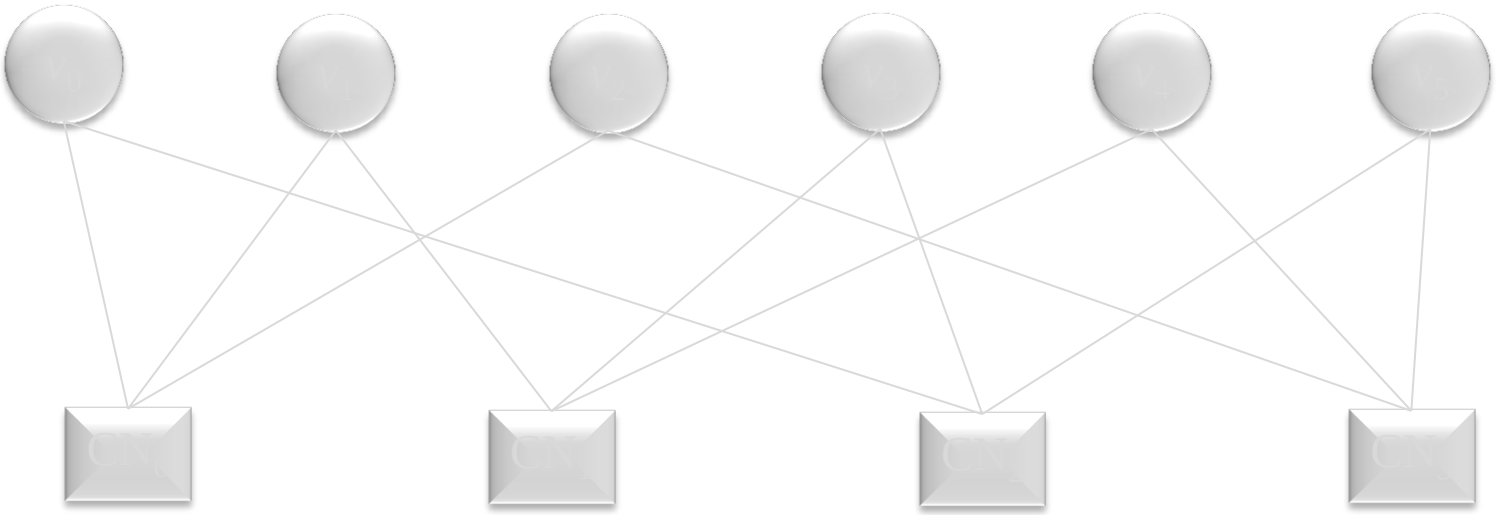
Transmitted CW

Received CW

decoded CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*+1



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

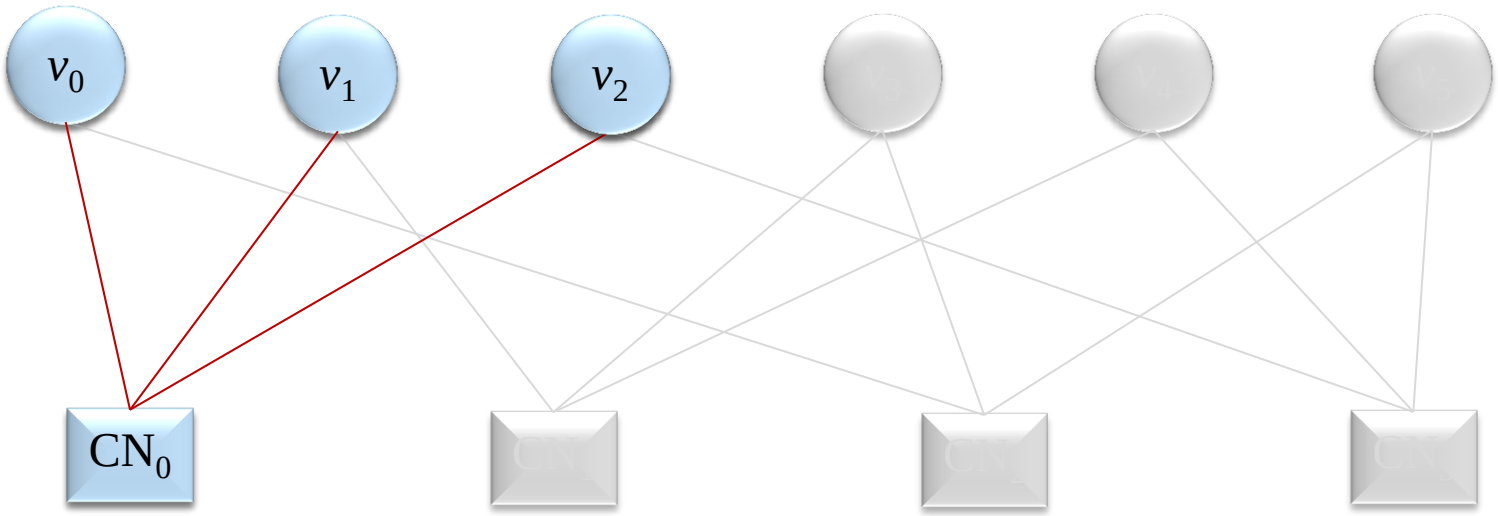
Transmitted CW

Received CW

decoded CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*+1



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

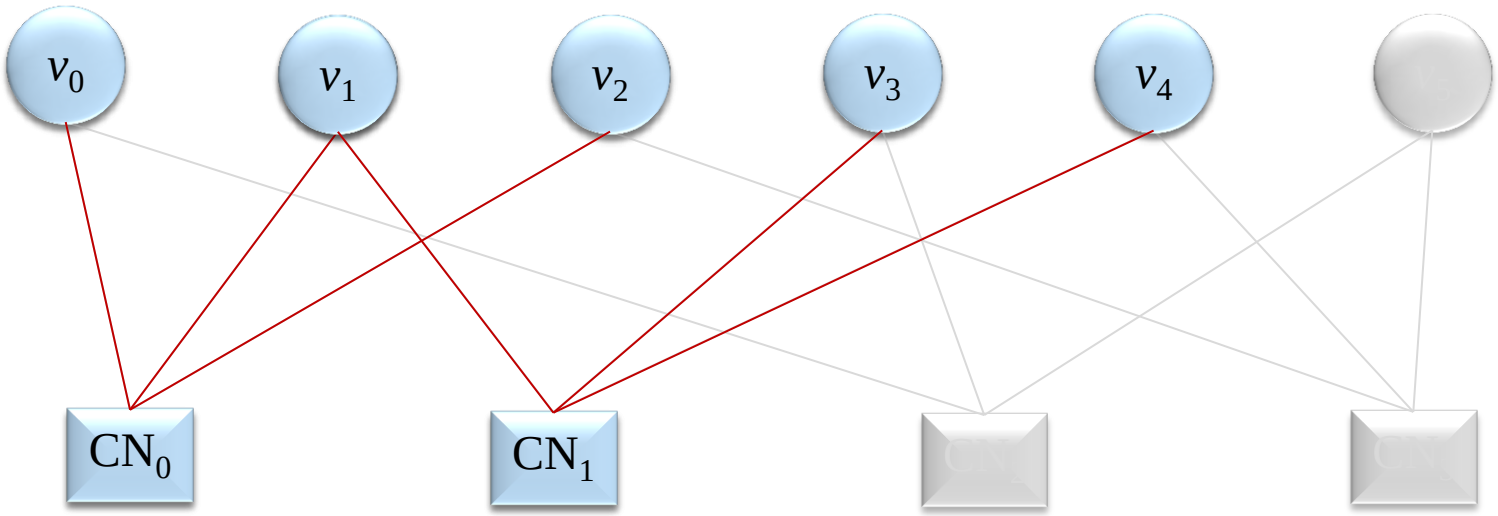
Transmitted CW

Received CW

decoded CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*+1



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

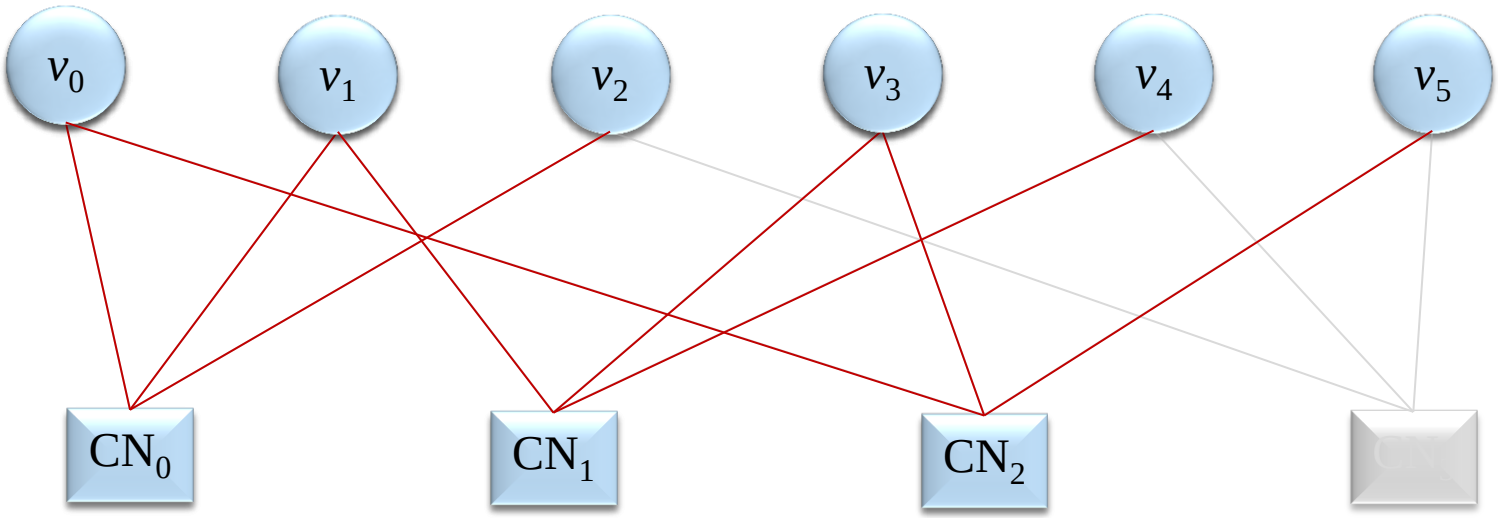
Transmitted CW

Received CW

decoded CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*+1



$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

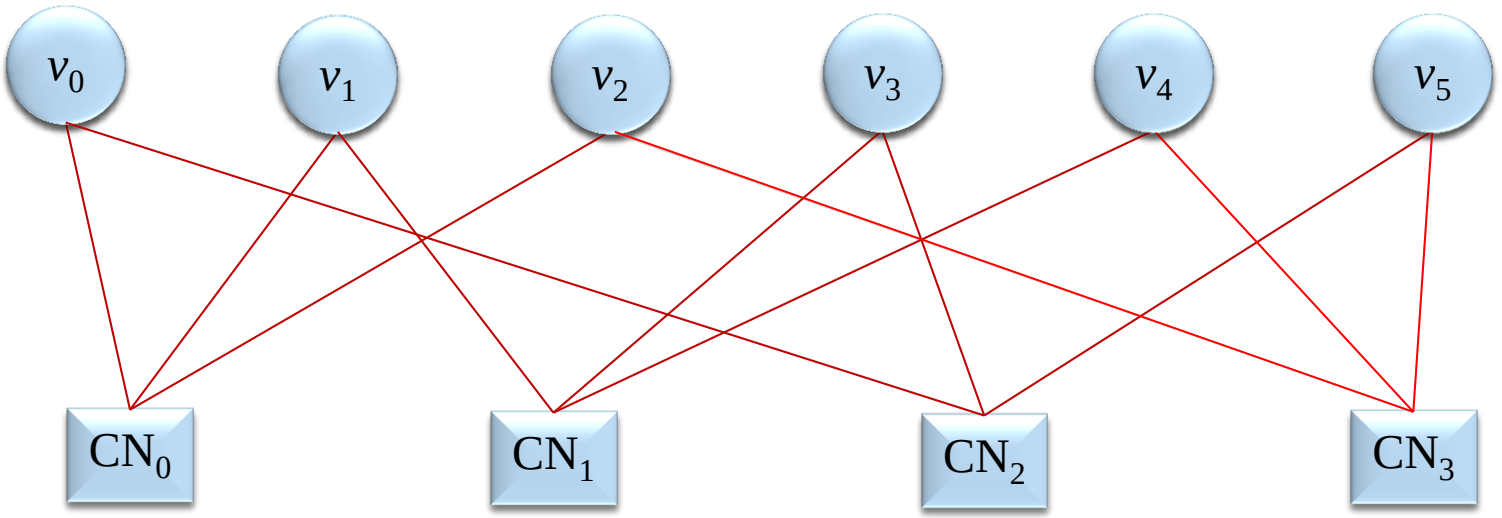
Transmitted CW

Received CW

decoded CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*+1



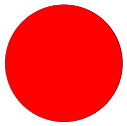
$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

Definition: Iterative decoding process

Transmitted CW

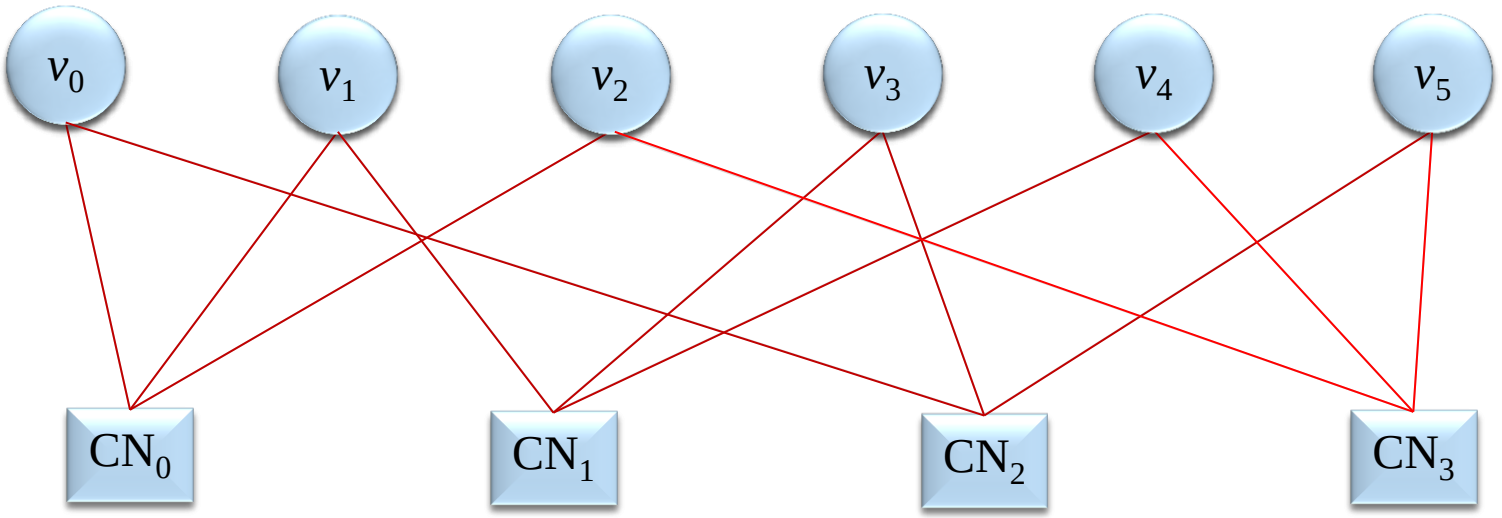


decoded CW

Received CW

$$\begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & 0 & 0 & 0 \\ 0 & h_{1,1} & 0 & h_{1,3} & h_{1,4} & 0 \\ h_{2,0} & 0 & 0 & h_{2,3} & 0 & h_{2,5} \\ 0 & 0 & h_{3,2} & 0 & h_{3,4} & h_{3,5} \end{bmatrix}$$

Iteration *i*+1

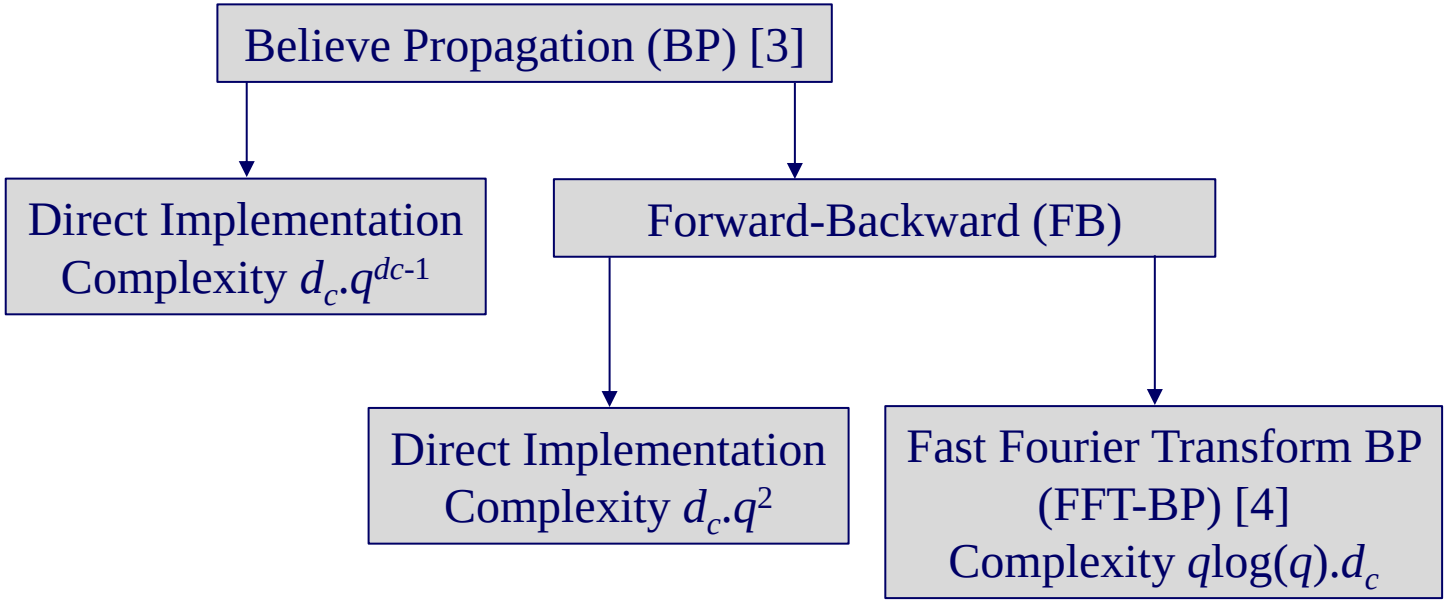


$i = 1, \ldots, n_{it}$

n_{it}: Maximum number of iterations

N=6 and M=4

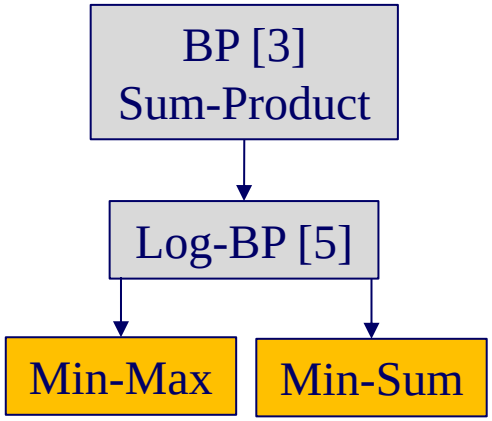
Optimal NB-LDPC Decoding Algorithms



- No mathematical approximations
- Not feasible to implement on hardware

[3] M.C. Davey and D. MacKay, 1998.
[4] L. Barnault and D. Declercq, 2003.
[5] H. Wymeersch, H. Steendam, and M. Moeneclaey, 2004
[6] Hongxin Song and J.R. Cruz. 2003.

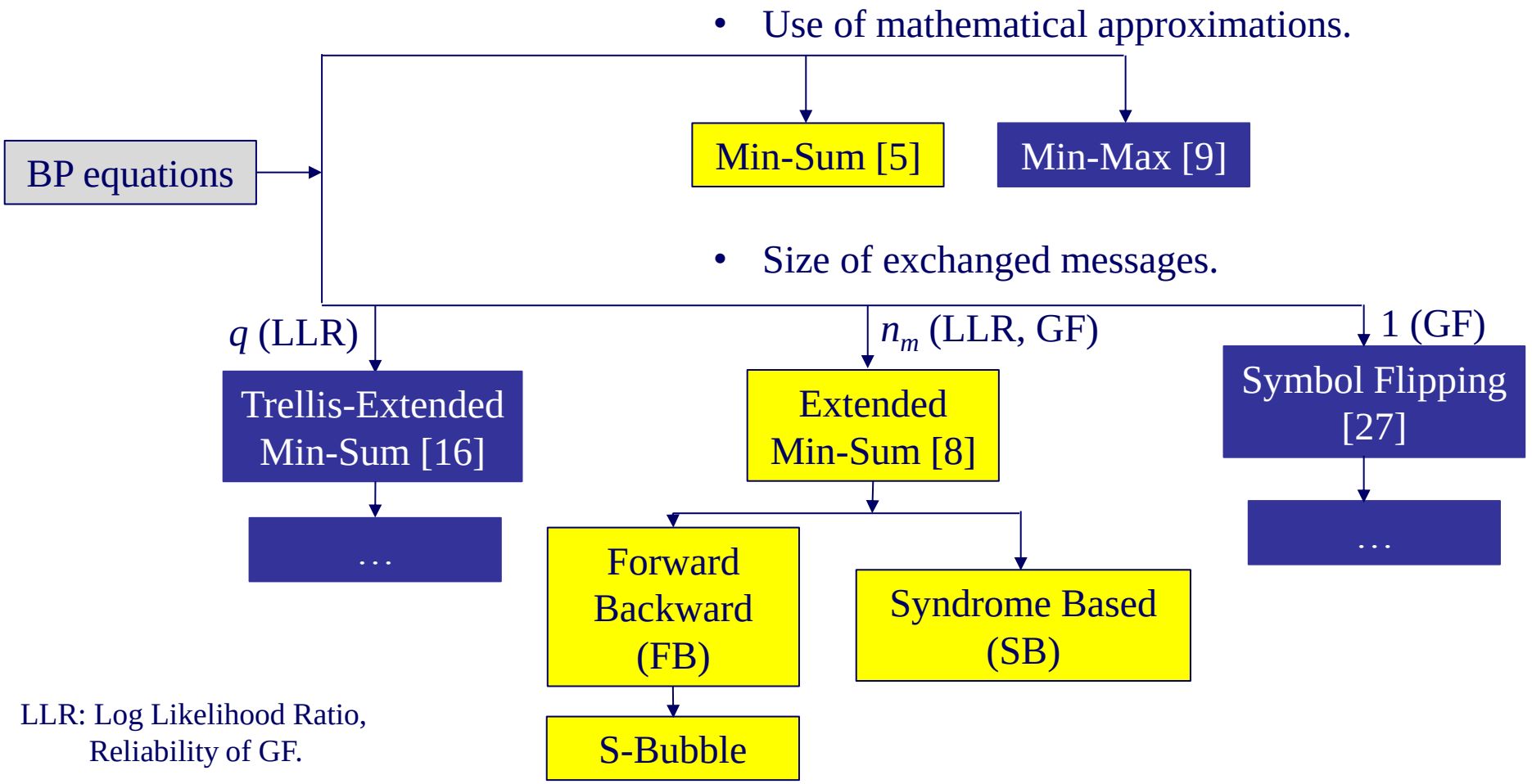
Optimal and Sub-Optimal NB-LDPC Decoding Algorithms



- Mathematical approximations
- Hardware Friendly

[3] M.C. Davey and D. MacKay, 1998.
[4] L. Barnault and D. Declercq, 2003.
[5] H. Wymeersch, H. Steendam, and M. Moeneclaey, 2004
[6] Hongxin Song and J.R. Cruz. 2003.

Sub-Optimal NB-LDPC Decoding Algorithms



LLR: Log Likelihood Ratio, Reliability of GF.

[8] Voicila, A., Declercq, D., Verdier, F., Fossorier, M., and Urard, P. June 2007.
[9] V. Savin. 2008.
[15] J. O. Lacruz, F. Garcia-Herrero, M. J. Canet, and J. Valls. Aug 2016.
[16] Erbao Li, D. Declercq, and K. Gunnam. July 2013.
[27] G. Sarkis, S. Mannor, and W. J. Gross, June 2009.

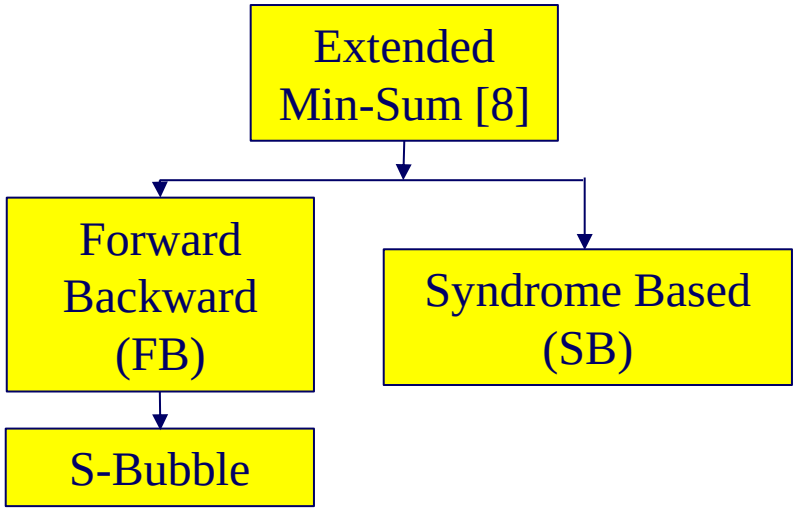
Sub-Optimal NB-LDPC Decoding Algorithms

- Use of mathematical approximations.

Min-Sum [5]

- Size of exchanged messages.

n_m (LLR, GF)



[8] Voicila, A., Declercq, D., Verdier, F., Fossorier, M., and Urard, P. June 2007.
[9] V. Savin. 2008.
[15] J. O. Lacruz, F. Garcia-Herrero, M. J. Canet, and J. Valls. Aug 2016.
[16] Erbao Li, D. Declercq, and K. Gunnam. July 2013.
[27] G. Sarkis, S. Mannor, and W. J. Gross, June 2009.

Outline

- Introduction
- NB-LDPC Codes
- EMS Algorithm and Architectures
  - Definition of LLR value and Intrinsic Candidates
  - CN and VN processing
  - CN and Pre-Sorting: state of the art and proposed architectures
- Proposed Parallel and Pipelined NB-LDPC Decoder Architecture
- Extra Related Works
- Conclusion, Perspectives and Publications

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2} \qquad U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

$U^{\oplus}$	1	0	0

a_i: Observed bit i
y_i : LLR value of a_i
$i = 0, \ldots, 2$

$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$

BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1

Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2} \qquad U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

a_i: Observed bit i
y_i : LLR value of a_i
$i = 0, \ldots, 2$

$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$

BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1

Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2} \qquad U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

Y	-2.3	-1.7	3.4

a_i: Observed bit i
y_i : LLR value of a_i
$i = 0, \ldots, 2$

$U^{\oplus} \in$ GF(8), U^{+}: LLR value of $U^{\oplus}$
BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1
Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2} \qquad U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

$\Delta(U^{\oplus}, Y)$ =	0	1	0

Y	-2.3	-1.7	3.4

a_i: Observed bit i
y_i : LLR value of a_i
i = 0, …, 2

$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$
BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1
Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2} \qquad U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

$\Delta(U^{\oplus}, Y)$ =	0	1	0

Y	-2.3	-1.7	3.4

a_i: Observed bit i
y_i : LLR value of a_i
i = 0, …, 2

$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$

BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1

Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2} \qquad U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

$\Delta(U^{\oplus}, Y)$ =	0	1	0

Y	-2.3	-1.7	3.4

a_i: Observed bit i
y_i : LLR value of a_i
i = 0, …, 2

$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$

BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1

Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2} \qquad U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

$\Delta(U^{\oplus}, Y)$ =	0	1	0

Y	-2.3	-1.7	3.4

a_i: Observed bit i
y_i : LLR value of a_i
i = 0, …, 2
$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$
BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1
Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2} \qquad U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

Y	-2.3	-1.7	3.4

$\Delta(U^{\oplus}, Y)$ =	0	1	0

$\lvert Y \rvert$ =	2.3	1.7	3.4

$\lvert Y \rvert . \Delta(U^{\oplus}, Y)$	0	1.7	0

a_i: Observed bit i
y_i : LLR value of a_i
i = 0, …, 2

$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$

BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1

Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2} \qquad U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

Y	-2.3	-1.7	3.4

$\Delta(U^{\oplus}, Y)$ =	0	1	0

$\lvert Y \rvert$ =	2.3	1.7	3.4

$\lvert Y \rvert . \Delta(U^{\oplus}, Y)$	0	1.7	0

$U^+ = 1.7$

a_i: Observed bit i
y_i : LLR value of a_i
$i = 0, \ldots, 2$

$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$
BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1
Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2}$$

$$U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

MS Algorithm

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

Y	-2.3	-1.7	3.4

8 Intrinsic messages I								
GF	000	001	010	011	100	101	110	111
LLR	4	7.4	2.3	5.7	1.7	5.1	0	3.4

$\Delta(U^{\oplus}, Y)$ =	0	1	0

$\lvert Y \rvert$ =	2.3	1.7	3.4

$\lvert Y \rvert . \Delta(U^{\oplus}, Y)$	0	1.7	0

$U^+ = 1.7$

a_i: Observed bit i
y_i : LLR value of a_i
$i = 0, \ldots, 2$

$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$
BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1
Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Definition of LLR value and Intrinsic Candidates

q=8, m=3, GF(8)

$$y_i = \ln\left(\frac{P(x_i \neq 0 / a_i)}{P(x_i = 1 / a_i)}\right) = \frac{2a_i}{\sigma^2}$$

$$U^+ \approx \sum_{i=0}^{m-1} | y_i | \Delta(x_i, y_i)$$

EMS Algorithm

$U^{\oplus}$	1	0	0

BPSK($U^{\oplus}$) =	-1	1	1

Y	-2.3	-1.7	3.4

8 Intrinsic messages I								
GF	000	001	010	011	100	101	110	111
LLR	4	7.4	2.3	5.7	1.7	5.1	0	3.4

$\Delta(U^{\oplus}, Y)$ =	0	1	0

$\lvert Y \rvert$ =	2.3	1.7	3.4

$\lvert Y \rvert . \Delta(U^{\oplus}, Y)$	0	1.7	0

$U^+ = 1.7$

a_i: Observed bit i
y_i : LLR value of a_i
$i = 0, \ldots, 2$

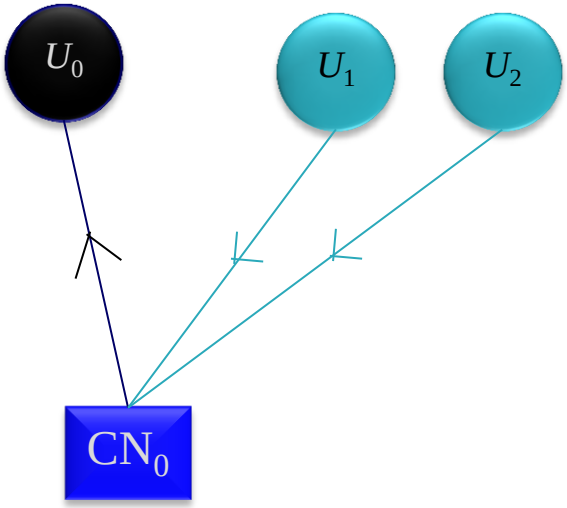
$U^{\oplus} \in$ GF(8), U^+: LLR value of $U^{\oplus}$
BPSK: Binary Phase Shift Keying. BPSK(1) = -1, BPSK(0) = 1
Δ(a, b) = 0 if sign(a) = sign(b), Δ(a, b) = 1 otherwise

Min-Sum Algorithm: CN update

GF(q = 4)

$$MIN\left(U_1^+[i]+U_2^+[j]\right)/U_1^{\oplus}[i]+U_2^{\oplus}[j]=0,\alpha^k$$

u^+: LLR value of v
$u^{\oplus}$: GF value of v
$i, j = 0, \ldots, q\text{-}1$
$k = 0, \ldots, q\text{-}2$



$U_2^{\oplus}$

$U_1^{\oplus}$ \ $\oplus$	0	α^0	α^1	α^2
0	0	α^0	α^1	α^2
α^0	α^0	0	α^2	α^1
α^1	α^1	α^2	0	α^0
α^2	α^2	α^1	α^0	0

U_2^{+}

U_1^{+} \ +	18	7	9	0
3	21	10	12	3
0	18	7	9	0
12	30	19	21	12
6	24	13	15	6

0	
α^0	
α^1	
α^2	

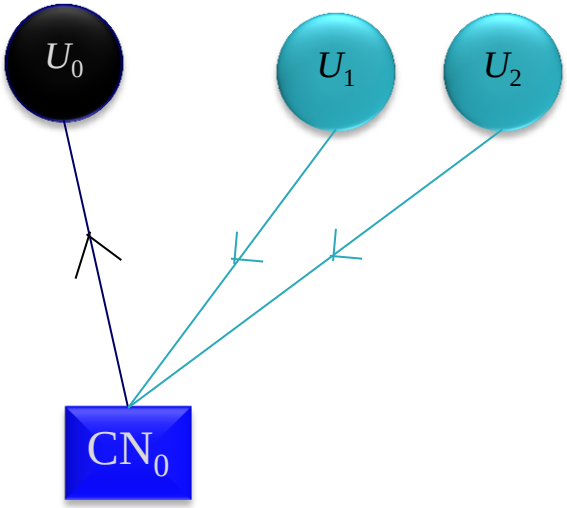
Extrinsic messages CN_0 to U_0

Min-Sum Algorithm: CN update

GF(q = 4)

$$MIN\left(U_1^+[i]+U_2^+[j]\right)/U_1^{\oplus}[i]+U_2^{\oplus}[j]=0,\alpha^k$$

u^+: LLR value of v
$u^{\oplus}$: GF value of v
$i, j = 0, \ldots, q\text{-}1$
$k = 0, \ldots, q\text{-}2$



$U_2^{\oplus}$

$\oplus$	0	α^0	α^1	α^2
0	0	α^0	α^1	α^2
α^0	α^0	0	α^2	α^1
α^1	α^1	α^2	0	α^0
α^2	α^2	α^1	α^0	0

$U_1^{\oplus}$

U_2^+

+	18	7	9	0
3	21	10	12	3
0	18	7	9	0
12	30	19	21	12
6	24	13	15	6

U_1^+

0	6
α^0	
α^1	
α^2	

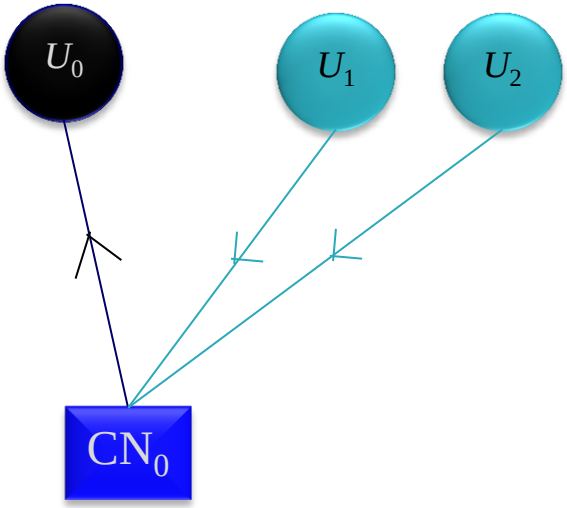
Extrinsic messages
CN_0 to U_0

Min-Sum Algorithm: CN update

GF(q = 4)

$$MIN\left(U_1^+[i]+U_2^+[j]\right)/U_1^{\oplus}[i]+U_2^{\oplus}[j]=0,\alpha^k$$

u^+: LLR value of v
$u^{\oplus}$: GF value of v
$i, j = 0, \ldots, q\text{-}1$
$k = 0, \ldots, q\text{-}2$



$U_2^{\oplus}$

$U_1^{\oplus}$ \ $\oplus$	0	α^0	α^1	α^2
0	0	α^0	α^1	α^2
α^0	α^0	0	α^2	α^1
α^1	α^1	α^2	0	α^0
α^2	α^2	α^1	α^0	0

U_2^+

U_1^+ \ +	18	7	9	0
3	21	10	12	3
0	18	7	9	0
12	30	19	21	12
6	24	13	15	6

0	6
α^0	10
α^1	
α^2	

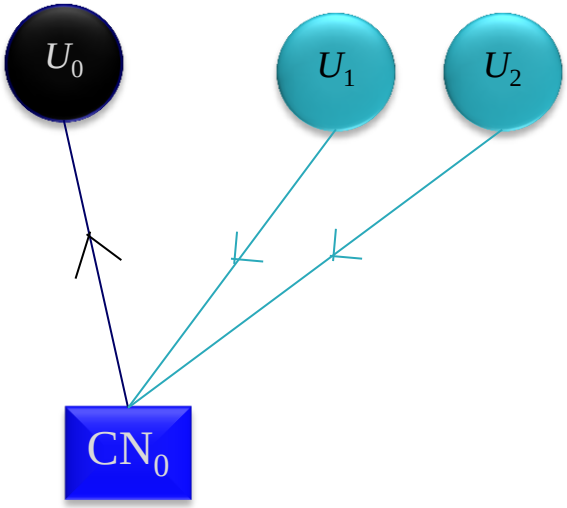
Extrinsic messages
CN_0 to U_0

Min-Sum Algorithm: CN update

GF(q = 4)

$$MIN\left(U_1^+[i]+U_2^+[j]\right)/U_1^{\oplus}[i]+U_2^{\oplus}[j]=0,\alpha^k$$

u^+: LLR value of v
$u^{\oplus}$: GF value of v
$i, j = 0, \ldots, q\text{-}1$
$k = 0, \ldots, q\text{-}2$



$U_2^{\oplus}$

$\oplus$	0	α^0	α^1	α^2
0	0	α^0	α^1	α^2
α^0	α^0	0	α^2	α^1
α^1	α^1	α^2	0	α^0
α^2	α^2	α^1	α^0	0

$U_1^{\oplus}$

U_2^+

+	18	7	9	0
3	21	10	12	3
0	18	7	9	0
12	30	19	21	12
6	24	13	15	6

U_1^+

0	6
α^0	10
α^1	0
α^2	

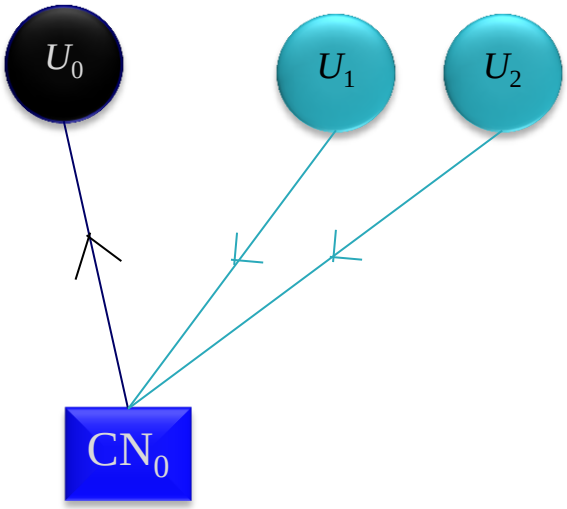
Extrinsic messages
CN_0 to U_0

Min-Sum Algorithm: CN update

GF(q = 4)

$$MIN\left(U_1^+[i]+U_2^+[j]\right)/U_1^{\oplus}[i]+U_2^{\oplus}[j]=0,\alpha^k$$

u^+: LLR value of v
$u^{\oplus}$: GF value of v
$i, j = 0, \ldots, q\text{-}1$
$k = 0, \ldots, q\text{-}2$



$U_2^{\oplus}$

$\oplus$	0	α^0	α^1	α^2
0	0	α^0	α^1	α^2
α^0	α^0	0	α^2	α^1
α^1	α^1	α^2	0	α^0
α^2	α^2	α^1	α^0	0

$U_1^{\oplus}$

U_2^+

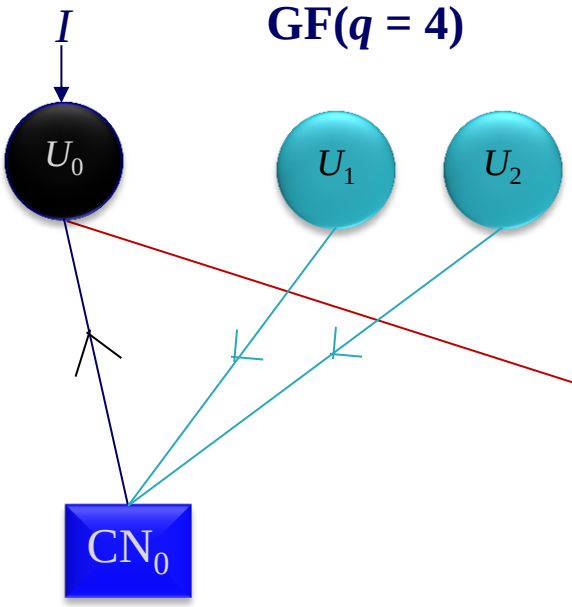
+	18	7	9	0
3	21	10	12	3
0	18	7	9	0
12	30	19	21	12
6	24	13	15	6

U_1^+

0	6
α^0	10
α^1	0
α^2	3

Extrinsic messages
CN_0 to U_0

Min-Sum Algorithm: VN update



$$U_2^+[i] + I[i]$$

U^+: LLR value of v
$U^{\oplus}$: GF value of v
$i, j = 0, \ldots, q\text{-}1$
$k = 0, \ldots, q\text{-}2$

qxd_c combinations

0	6
α^0	10
α^1	0
α^2	3

Extrinsic messages
CN_0 to U_0

+

0	0
α^0	7
α^1	1
α^2	5

Intrinsic U_0
I

=

0	6
α^0	17
α^1	1
α^2	8

A priori messages
U_0 to CN_2

Normalization ⇒

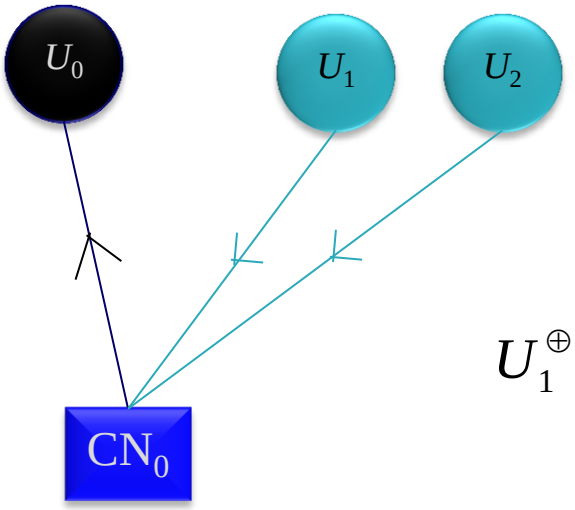
0	5
α^0	16
α^1	0
α^2	7

EMS Algorithm: CN update

GF(q=4), n_m=2

$$MIN\left(U_1^+[i]+U_2^+[j]\right)/U_1^{\oplus}[i]+U_2^{\oplus}[j]=0,\alpha^k$$

U^+: LLR value of v
$U^{\oplus}$: GF value of v
$i, j = 0, \ldots, n_m\text{-}1$
$k = 0, \ldots, q\text{-}2$



$U_2^{\oplus}$

$U_1^{\oplus}$ \ $\oplus$	0	α^0	α^1	α^2
0				
α^0				
α^1				
α^2				

U_2^{+}

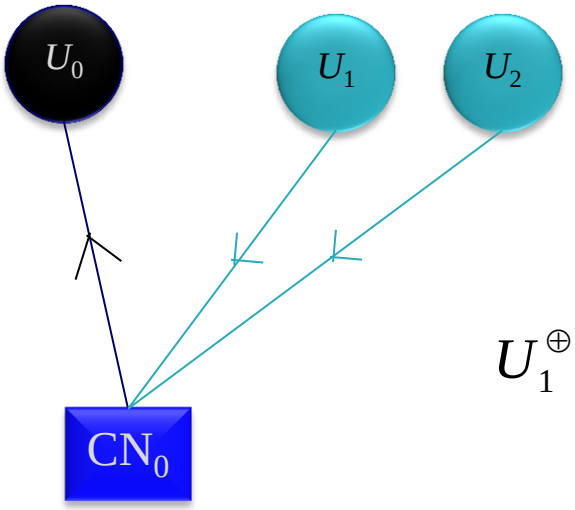
U_1^{+} \ +	18	7	9	0
3				
0				
12				
6				

EMS Algorithm: CN update

GF(q=4), n_m=2

$$MIN\left(U_1^+[i]+U_2^+[j]\right)/U_1^{\oplus}[i]+U_2^{\oplus}[j]=0,\alpha^k$$

U^+: LLR value of v
$U^{\oplus}$: GF value of v
$i, j = 0, \ldots, n_m\text{-}1$
$k = 0, \ldots, q\text{-}2$



$U_2^{\oplus}$

$U_1^{\oplus}$ \ $\oplus$	0	α^0	α^1	α^2
0				
α^0				
α^1				
α^2				

$\oplus$	α^2	α^0	α^1	0
α^0				
0				
α^2				
α^1				

U_2^+

U_1^+ \ +	18	7	9	0
3				
0				
12				
6				

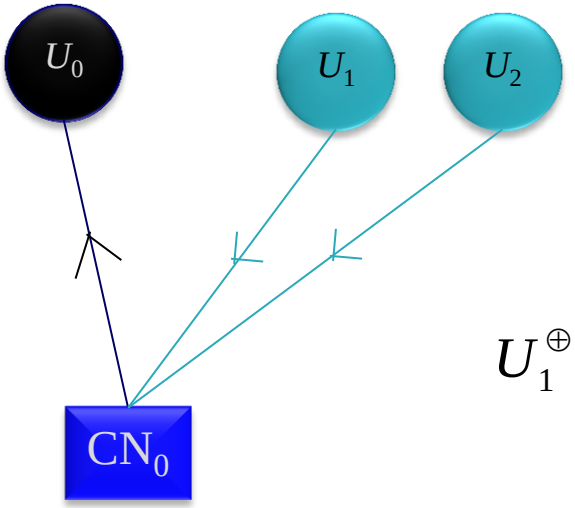
+	0	7	9	18
0				
3				
6				
12				

EMS Algorithm: CN update

GF(q=4), n_m=2

$$MIN\left(U_1^+[i]+U_2^+[j]\right)/U_1^{\oplus}[i]+U_2^{\oplus}[j]=0,\alpha^k$$

U^+: LLR value of v
$U^{\oplus}$: GF value of v
$i, j = 0, \ldots, n_m\text{-}1$
$k = 0, \ldots, q\text{-}2$



$U_2^{\oplus}$

$U_1^{\oplus}$

$\oplus$	α^2	α^0		
α^0	α^1	0		
0	α^2	α^0		

$\oplus$	α^2	α^0	α^1	0
α^0				
0				
α^2				
α^1				

U_2^+

U_1^+

+	0	7		
0	0	7		
3	3	10		

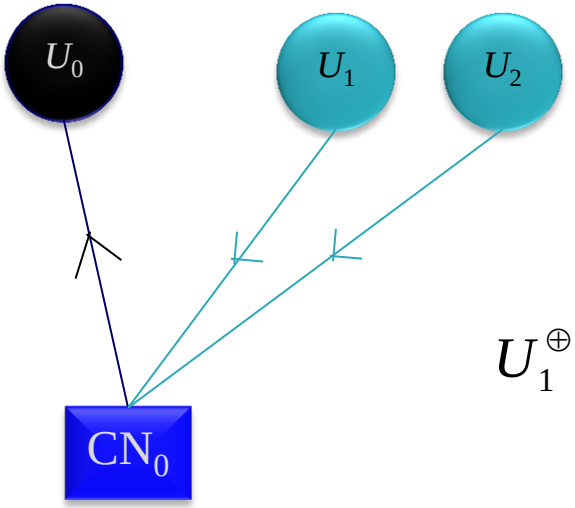
+	0	7	9	18
0				
3				
6				
12				

EMS Algorithm: CN update

GF(q=4), n_m=2

$$MIN\left(U_1^+[i]+U_2^+[j]\right)/U_1^{\oplus}[i]+U_2^{\oplus}[j]=0,\alpha^k$$

U^+: LLR value of v
$U^{\oplus}$: GF value of v
$i, j = 0, \ldots, n_m\text{-}1$
$k = 0, \ldots, q\text{-}2$



$U_2^{\oplus}$

$\oplus$	α^2	α^0		
α^0	α^1	0		
0	α^2	α^0		

$U_1^{\oplus}$

α^1	0
α^2	3

Extrinsic messages CN_0 to U_0

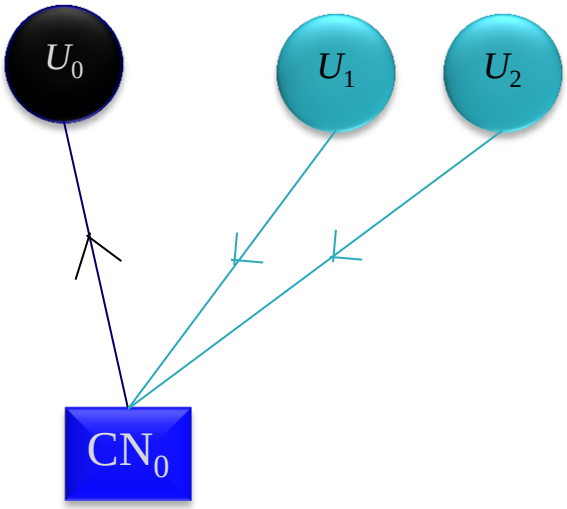
U_2^+

+	0	7		
0	0	7		
3	3	10		

U_1^+

EMS Algorithm: VN update

GF(q=4), n_m=2



0	0
α^0	7
α^1	1
α^2	5

Intrinsic U_0
MS algorithm

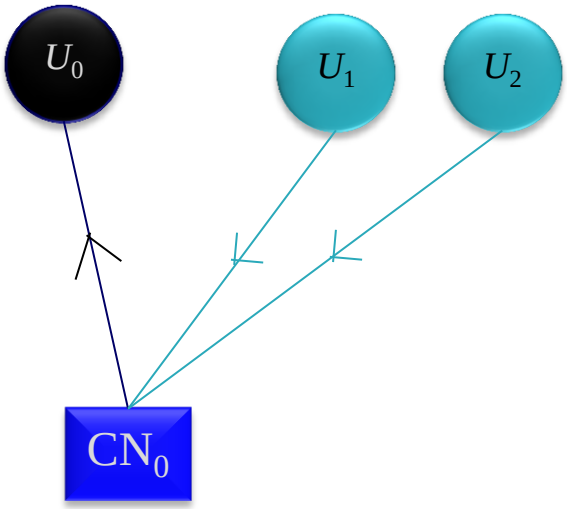
α^1	0
α^2	3

Extrinsic messages
CN_0 to U_0

α^1	1
α^2	8

EMS Algorithm: VN update

GF(q=4), n_m=2



α^1	1
0	0

Intrinsic U_0

0	1
α^0	7
α^1	1
α^2	5

Intrinsic U_0
MS algorithm

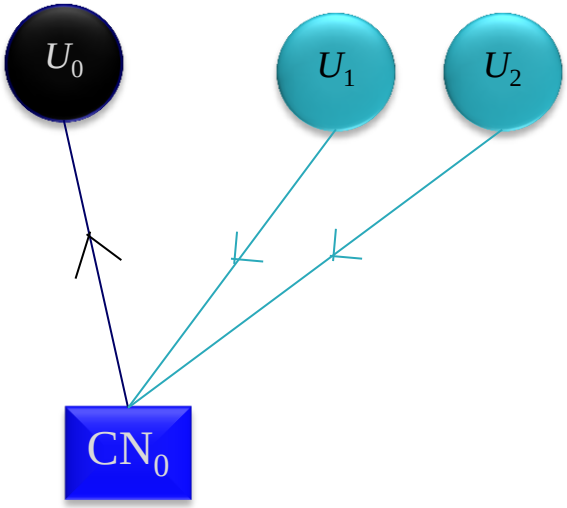
α^1	0
α^2	3

Extrinsic messages
CN_0 to U_0

α^1	1
α^2	8

EMS Algorithm: VN update

GF(q=4), n_m=2



α^1	1
0	0

Intrinsic U_0

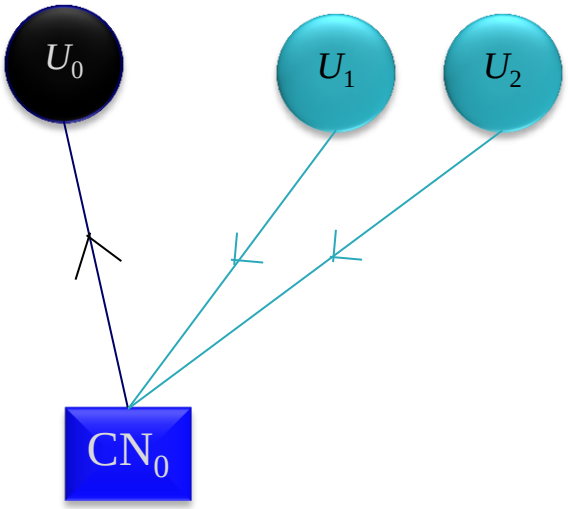
α^1	0
α^2	3

Extrinsic messages CN_0 to U_0

α^1	1
α^2	8

EMS Algorithm: VN update

GF(q=4), n_m=2



α^1	1
0	0

Intrinsic U_0

α^1	0
α^2	3

Extrinsic messages CN_0 to U_0

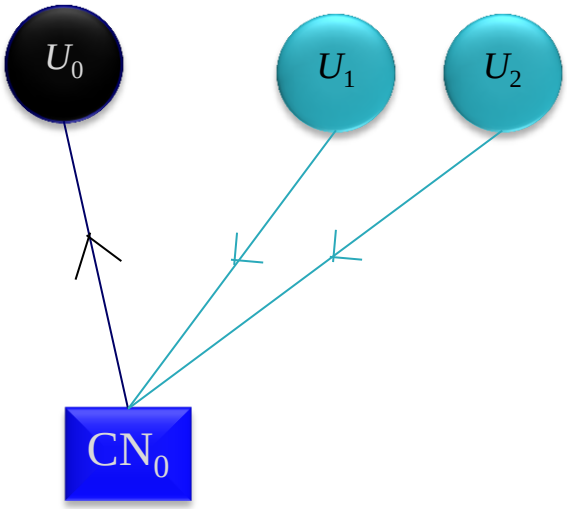
α^1	1
α^2	8

Offset

Constant offset = 1

EMS Algorithm: VN update

GF(q=4), n_m=2



α^1	1
0	0

Intrinsic U_0

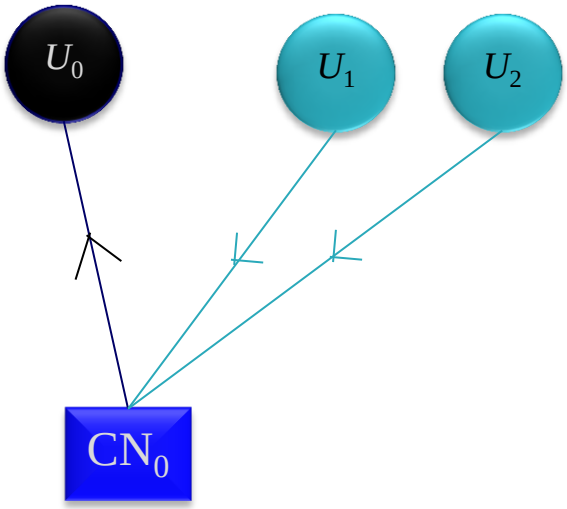
α^1	0
α^2	3

Extrinsic messages CN_0 to U_0

α^1	1
α^2	8
0	4

EMS Algorithm: VN update

GF(q=4), n_m=2



α^1	1
0	0

Intrinsic U_0

n_mxd_c cobinations

α^1	0
α^2	3

Extrinsic messages CN_0 to U_0

α^1	1
α^2	8
0	4

α^1	1
0	4

A priori messages U_0 to CN_2

Normalization

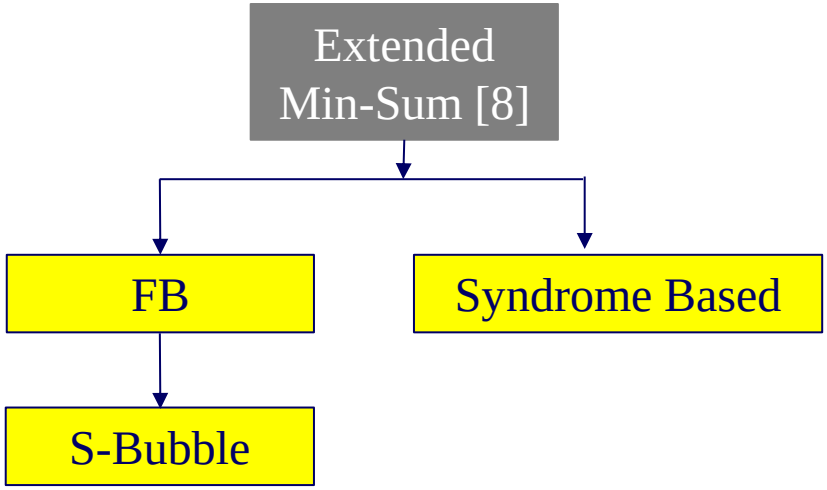
α^1	0
0	3

- Use of mathematical approximations.

Min-Sum [5]

- Size of exchanged messages.

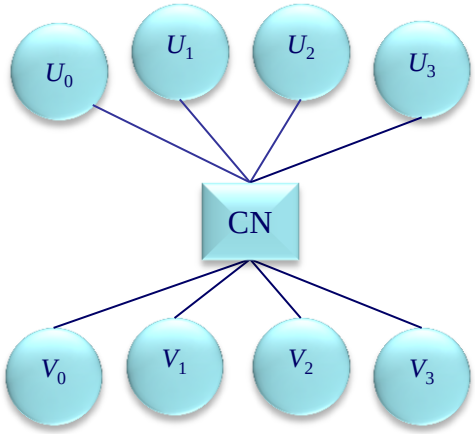
n_m (LLR, GF)



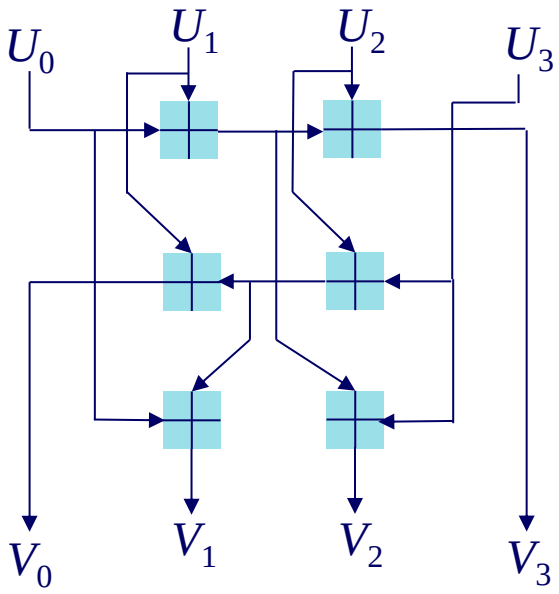
[8] Voicila, A., Declercq, D., Verdier, F., Fossorier, M., and Urard, P. June 2007.
[9] V. Savin. 2008.
[15] J. O. Lacruz, F. Garcia-Herrero, M. J. Canet, and J. Valls. Aug 2016.
[16] Erbao Li, D. Declercq, and K. Gunnam. July 2013.
[27] G. Sarkis, S. Mannor, and W. J. Gross, June 2009.

CN and Presorting: State-of-the-art: Forward Backward CN (FB-CN)

$d_c = 4$



[11]

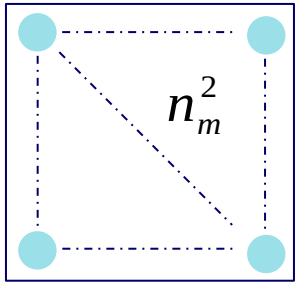


ECN

Forward Layer

Merge Layer

Backward Layer

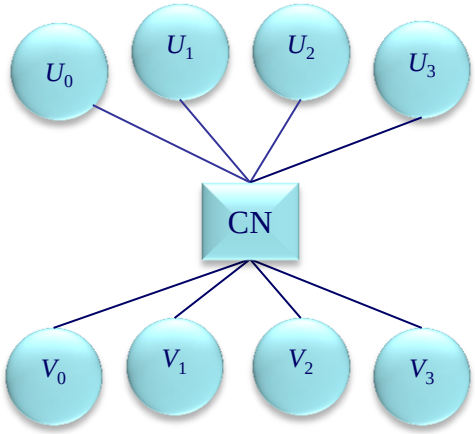


ECN: Elementary Check Node

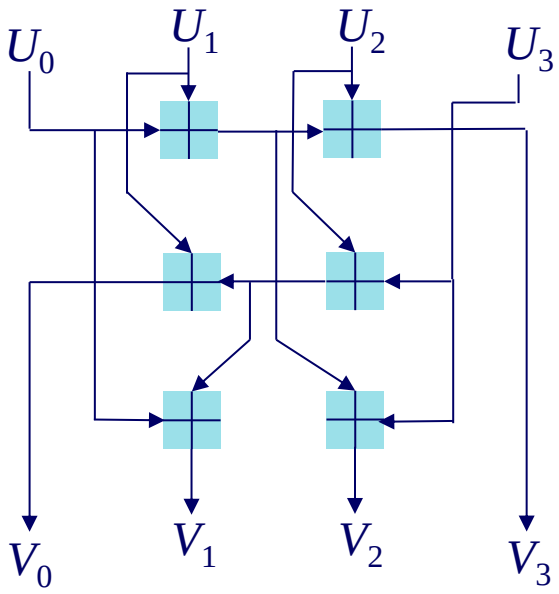
[11] H. Wymeersch, H. Steendam, and M. Moeneclaey. 2004.
[29] Oussama Abassi, Laura Conde-Canencia, Ali Al Ghouwayel, Emmanuel Boutillon. 2017.

CN and Presorting: State-of-the-art: Forward Backward CN (FB-CN)

$d_c = 4$



[11]



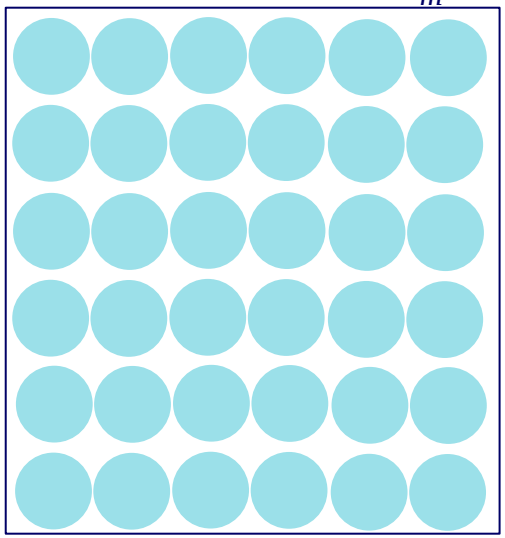
ECN

Forward Layer

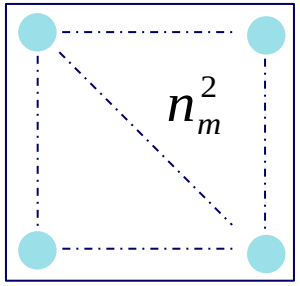
Merge Layer

Backward Layer

S-Bubble ECN [29], $n_m = 6$



$3n_m$-4

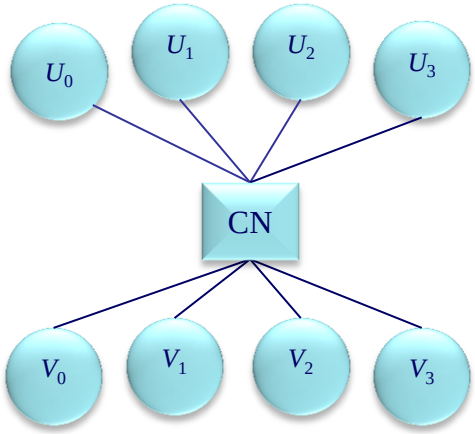


ECN: Elementary Check Node

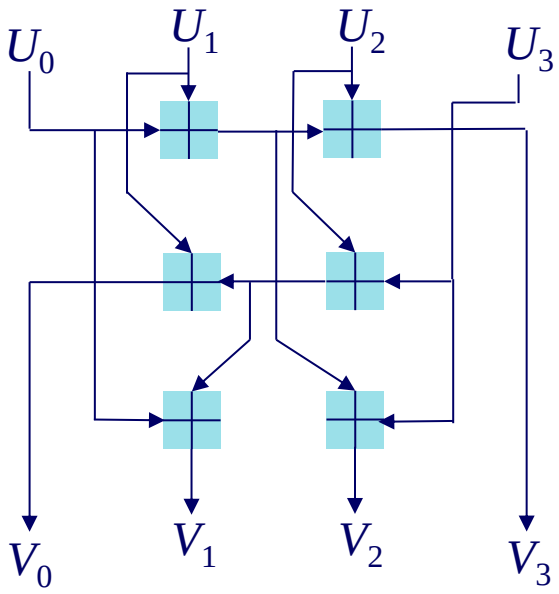
[11] H. Wymeersch, H. Steendam, and M. Moeneclaey. 2004.
[29] Oussama Abassi, Laura Conde-Canencia, Ali Al Ghouwayel, Emmanuel Boutillon. 2017.

CN and Presorting: State-of-the-art: Forward Backward CN (FB-CN)

$d_c = 4$



[11]



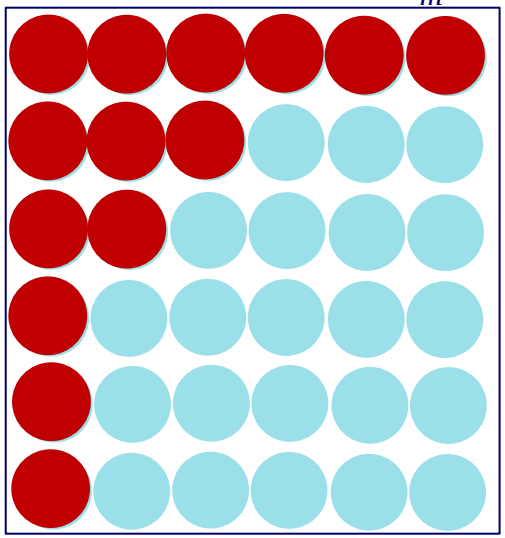
ECN

Forward Layer

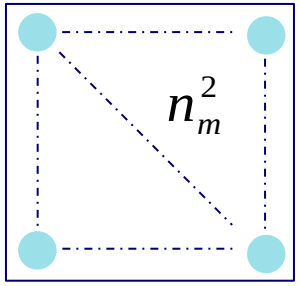
Merge Layer

Backward Layer

S-Bubble ECN [29], $n_m = 6$



$3n_m$-4

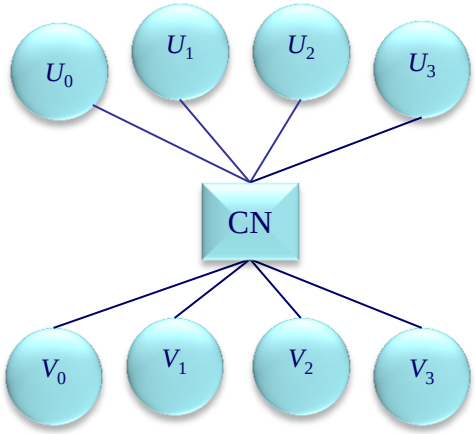


ECN: Elementary Check Node

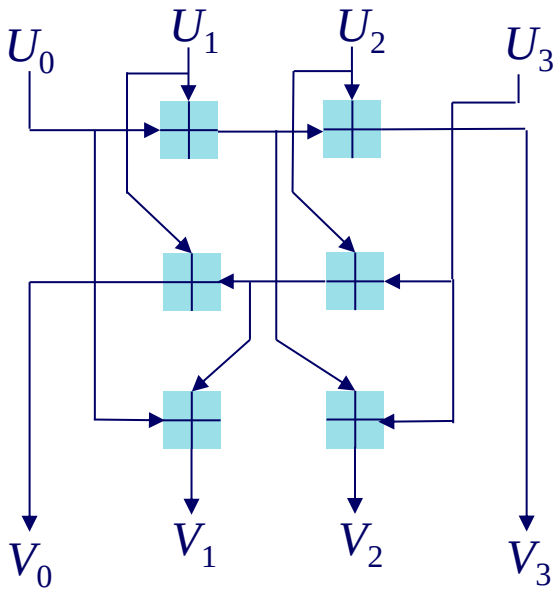
[11] H. Wymeersch, H. Steendam, and M. Moeneclaey. 2004.
[29] Oussama Abassi, Laura Conde-Canencia, Ali Al Ghouwayel, Emmanuel Boutillon. 2017.

CN and Presorting: State-of-the-art: Forward Backward CN (FB-CN)

$d_c = 4$



[11]



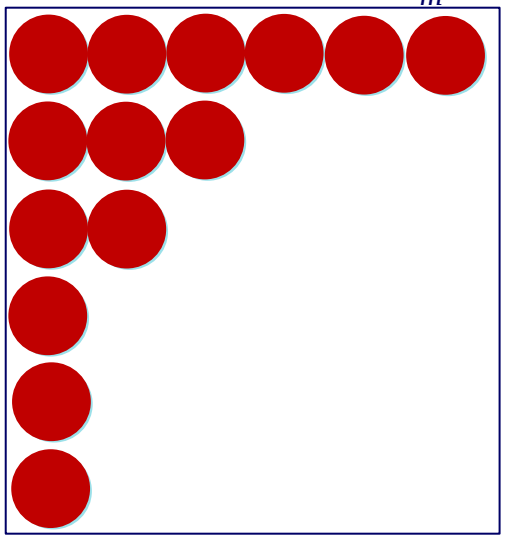
ECN

Forward Layer

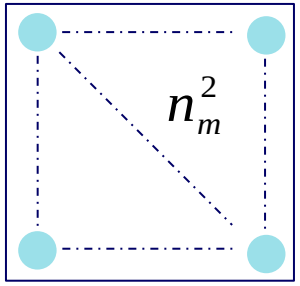
Merge Layer

Backward Layer

S-Bubble ECN [29], $n_m = 6$



$3n_m$-4

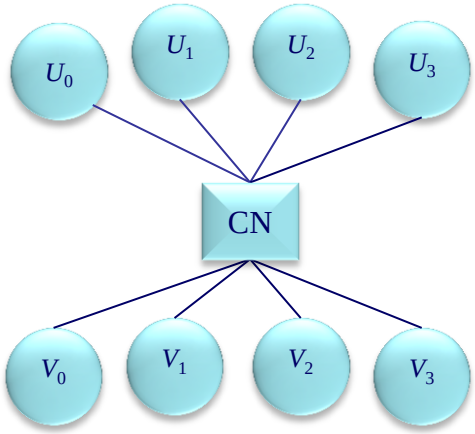


ECN: Elementary Check Node

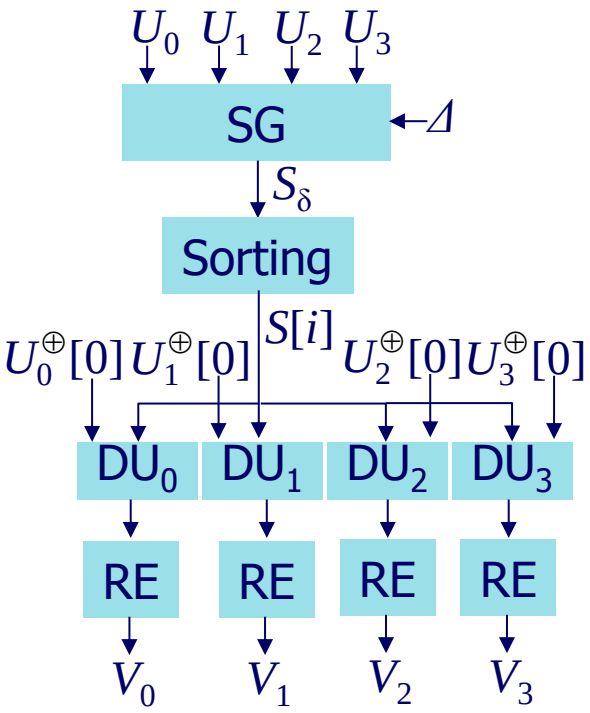
[11] H. Wymeersch, H. Steendam, and M. Moeneclaey. 2004.
[29] Oussama Abassi, Laura Conde-Canencia, Ali Al Ghouwayel, Emmanuel Boutillon. 2017.

CN and Presorting: State-of-the-art: Syndrome Based CN (SB-CN)

$d_c = 4$



SB-CN [17]



SG: Syndrome Generator
Δ: Set of deviation paths
$i = 0, \ldots, |\Delta|-1$
$|\Delta|$: Cardinality of Δ
DU: Decorrelation Unit
RE: Redundant Elimination

[17] P Schl¨afer, N Wehn, M Alles, T Lehnigk-Emden, E Boutillon. "*Syndrome based check node processing of high order NB-LDPC decoders*". International Conference on Telecommunications, Apr 2015, Sydney, Australia.

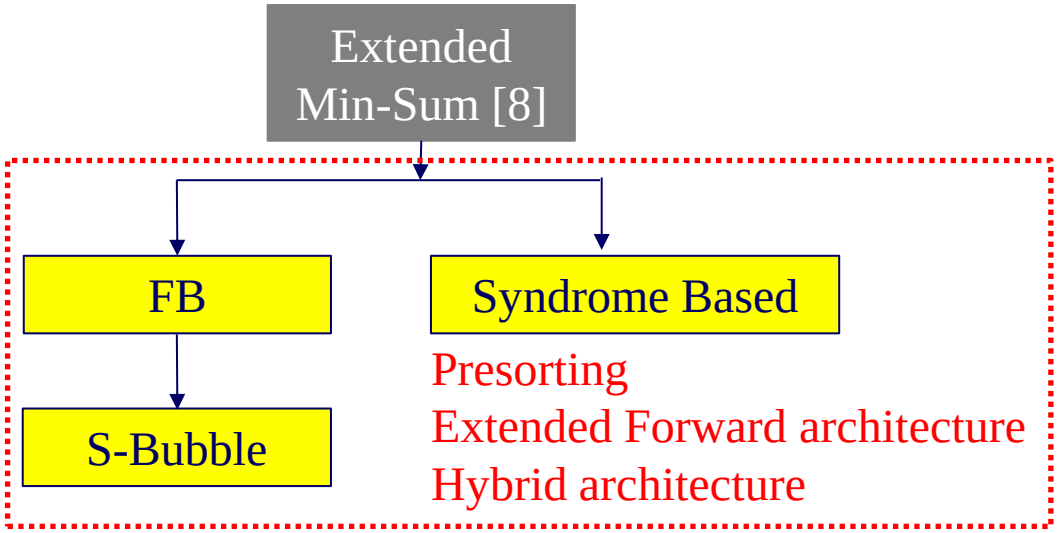
Transition exist->proposed

- Use of mathematical approximations.

Min-Sum [5]

- Size of exchanged messages.

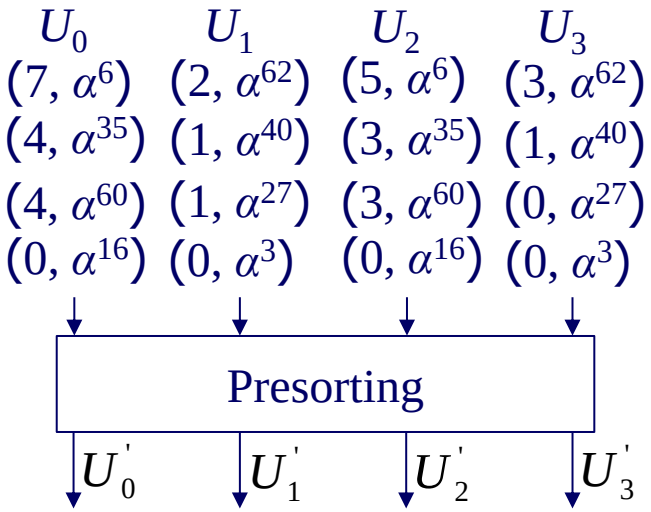
n_m (LLR, GF)



[8] Voicila, A., Declercq, D., Verdier, F., Fossorier, M., and Urard, P. June 2007.
[9] V. Savin. 2008.
[15] J. O. Lacruz, F. Garcia-Herrero, M. J. Canet, and J. Valls. Aug 2016.
[16] Erbao Li, D. Declercq, and K. Gunnam. July 2013.
[27] G. Sarkis, S. Mannor, and W. J. Gross, June 2009.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

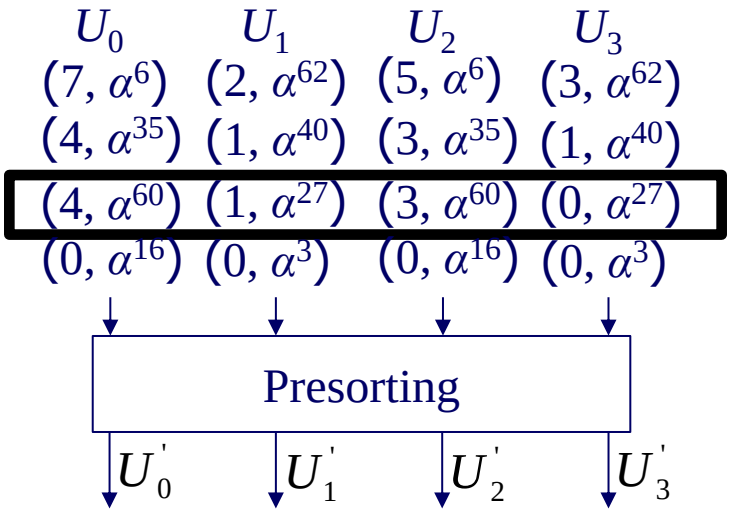


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

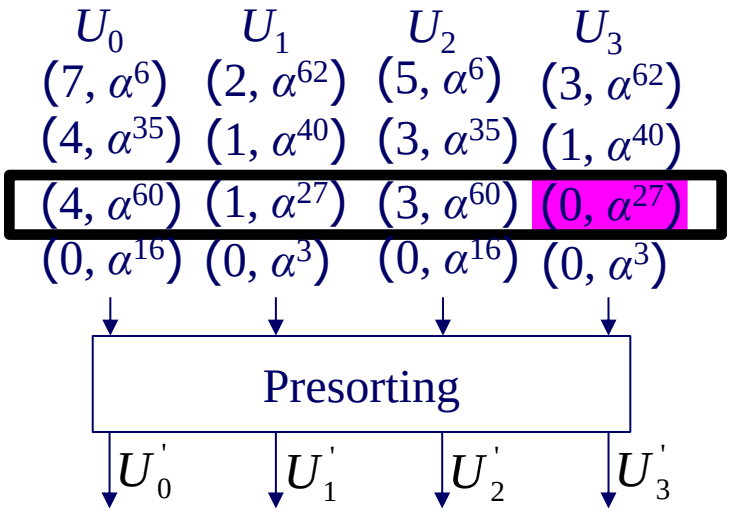


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

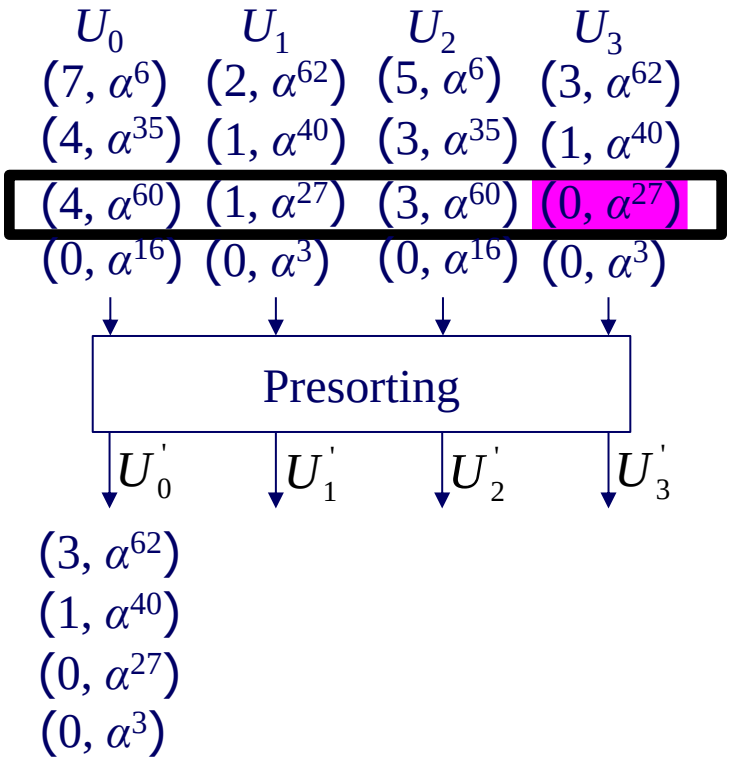


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

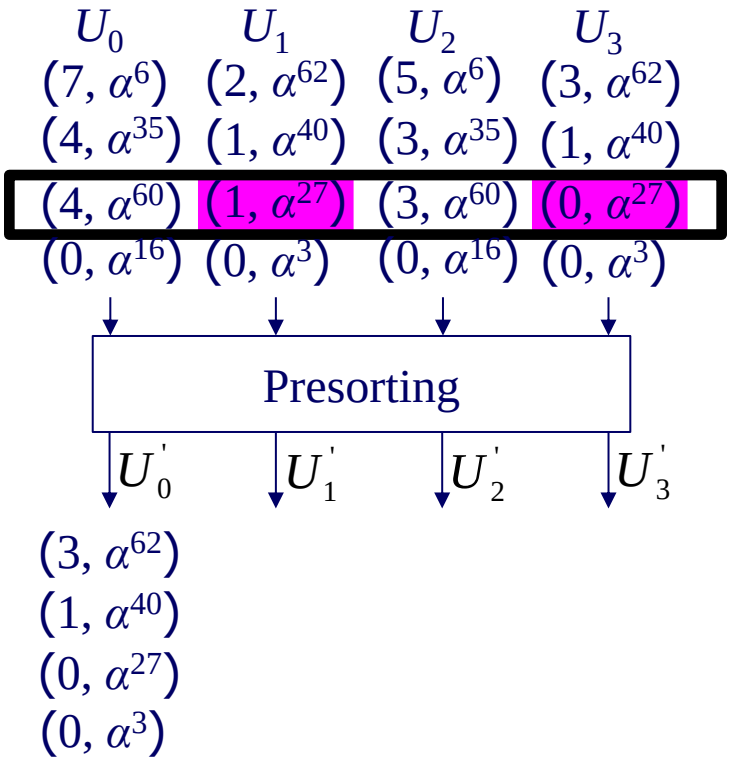


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

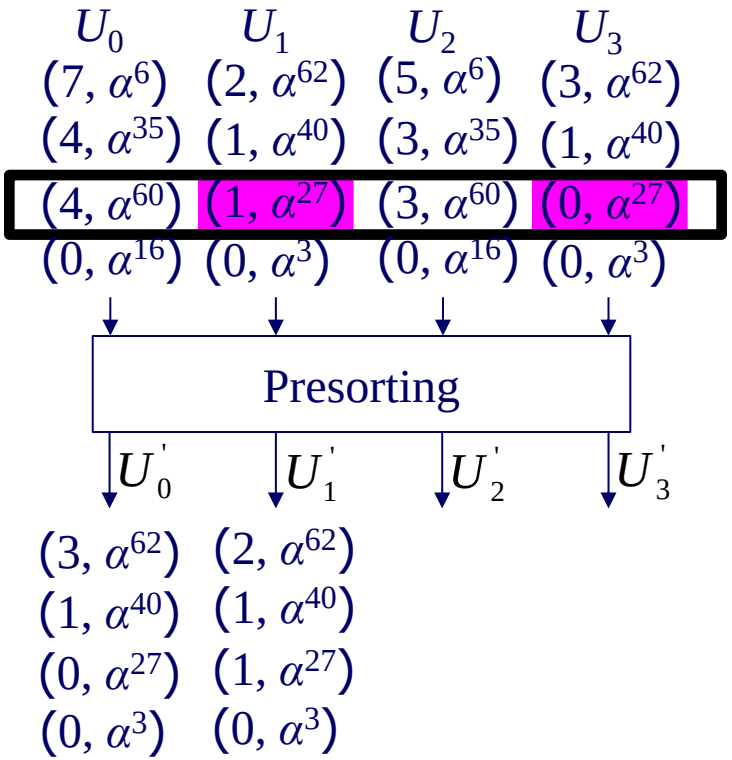


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

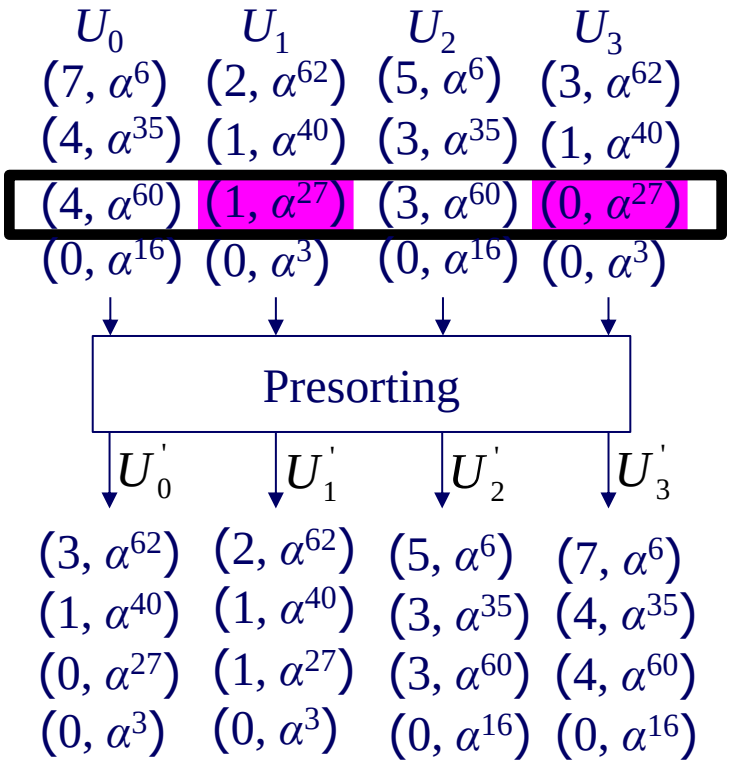


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

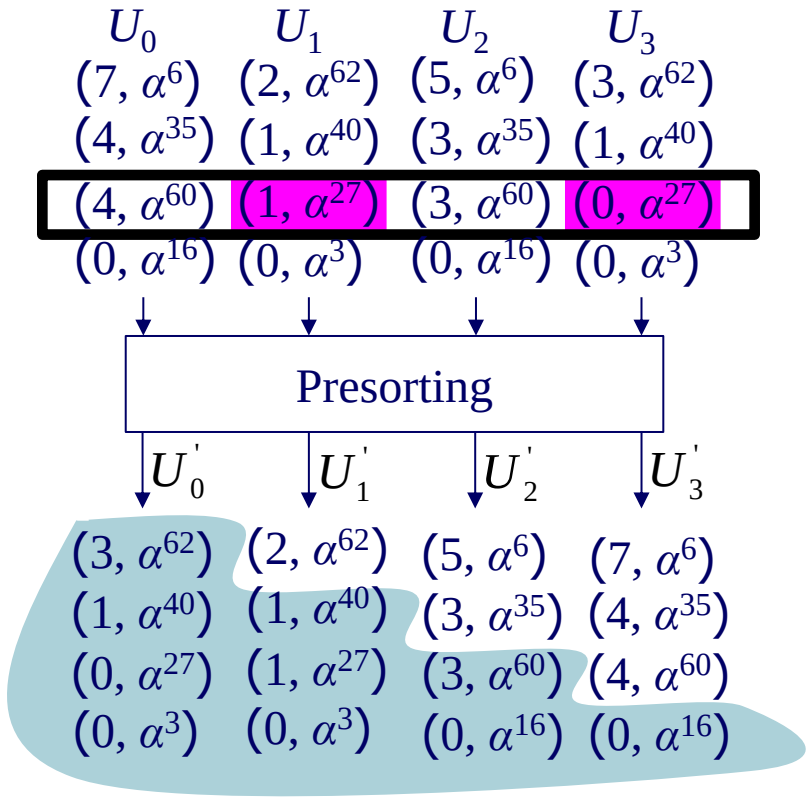


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

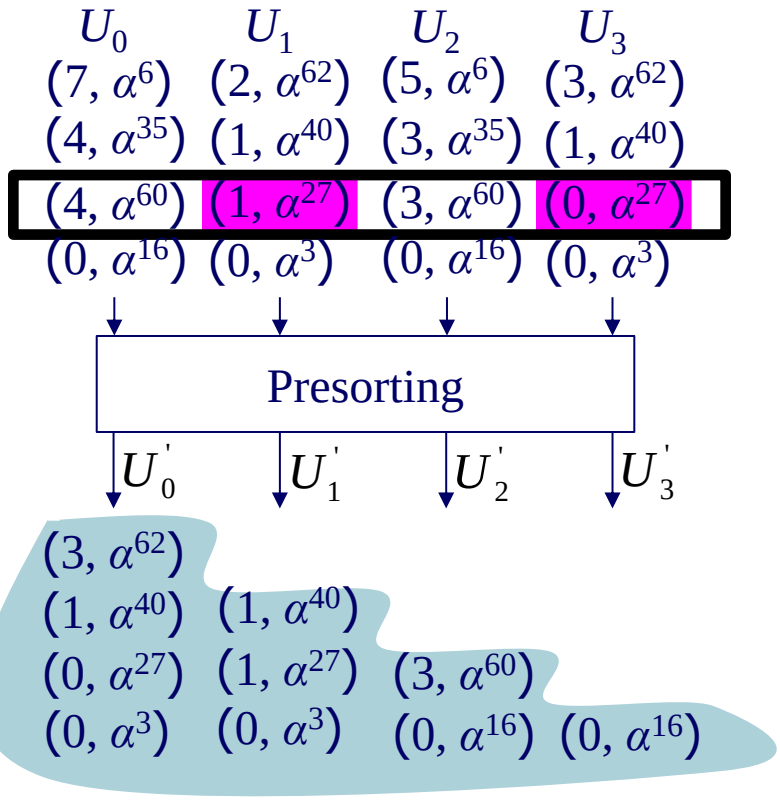


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

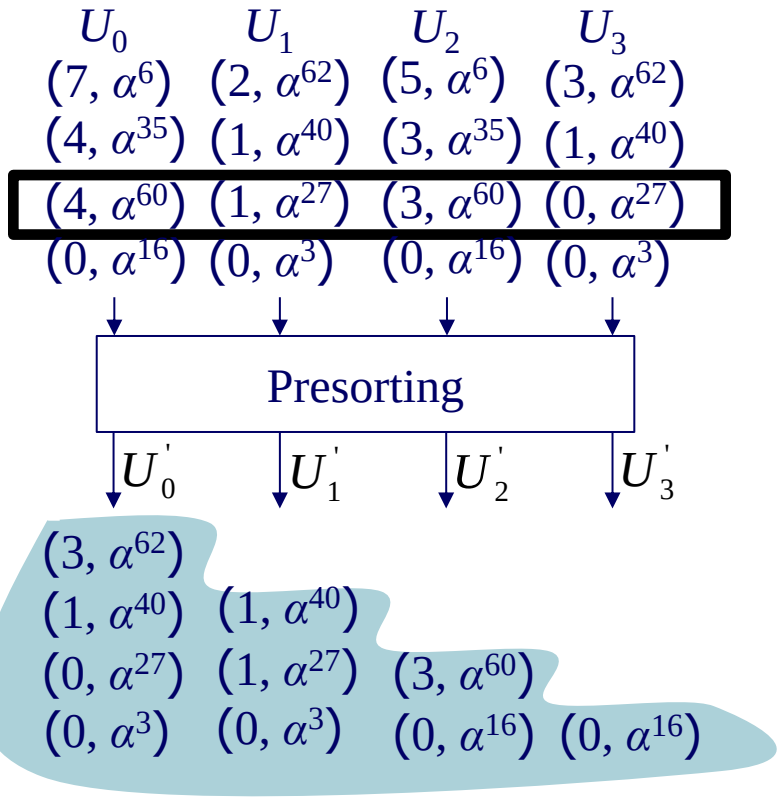


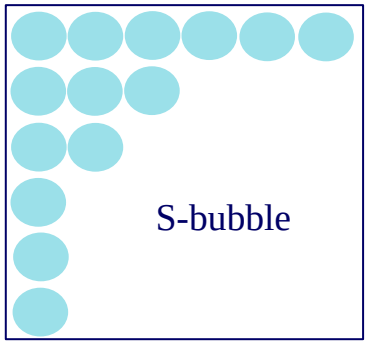
[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

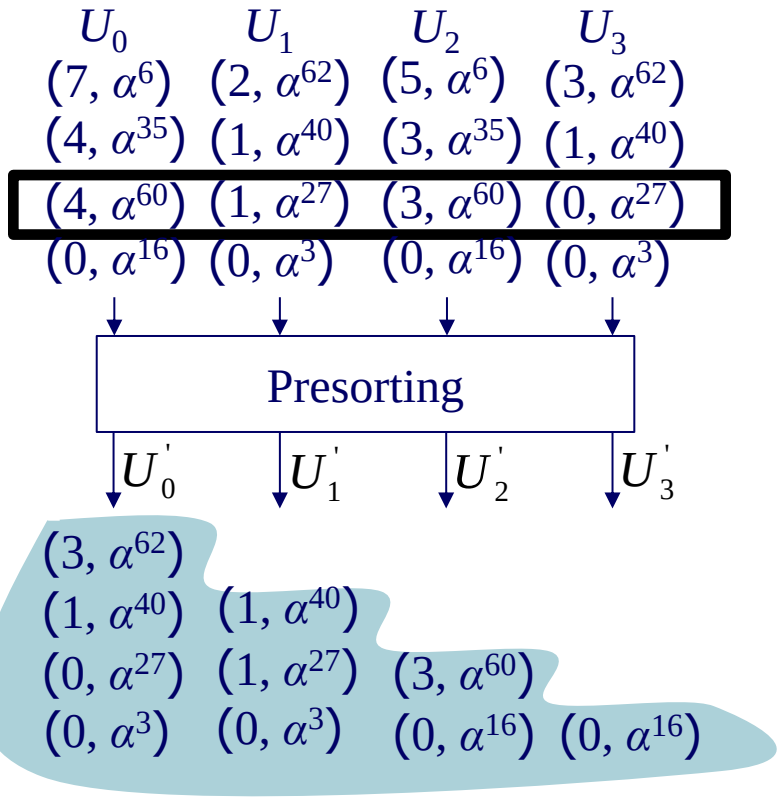


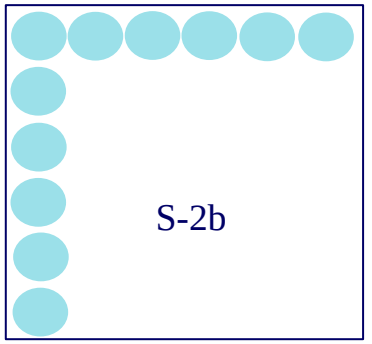


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.
[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

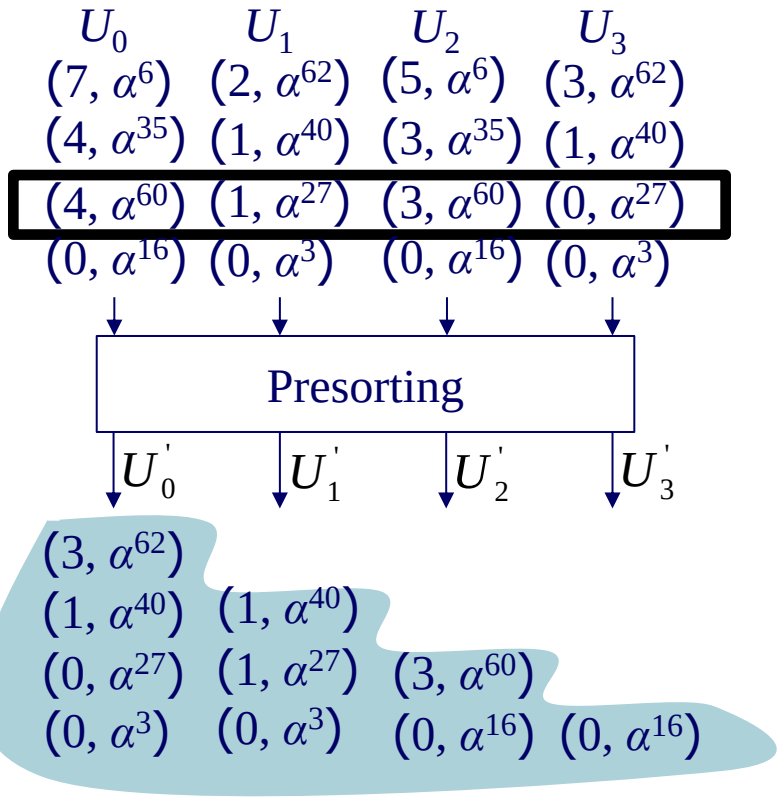


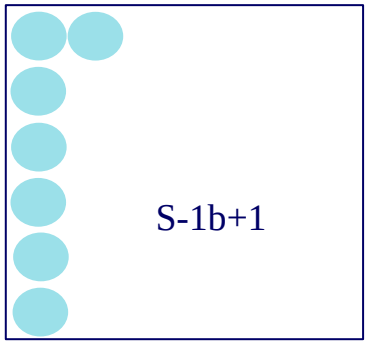


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.
[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)



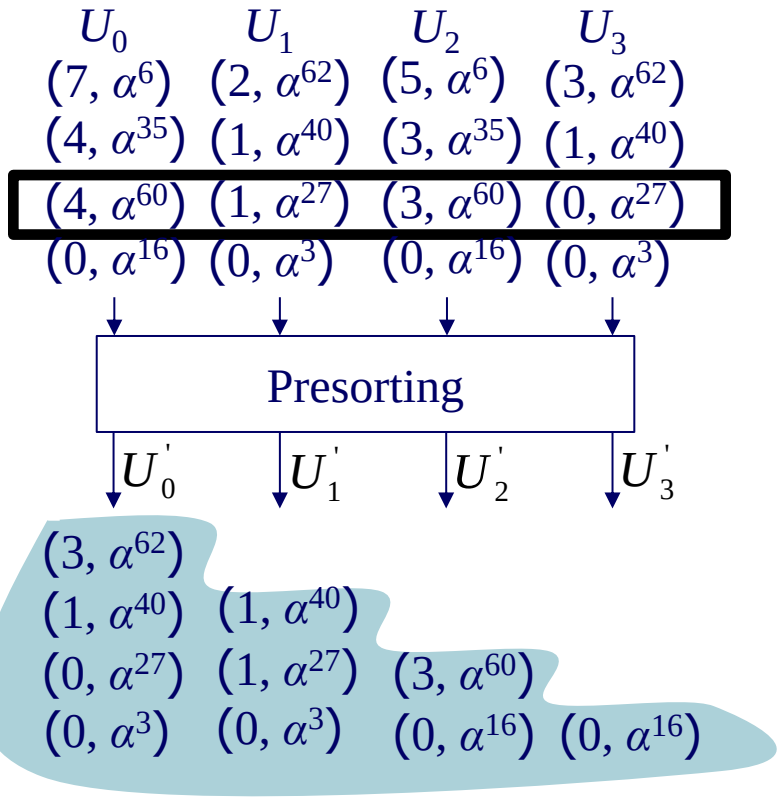


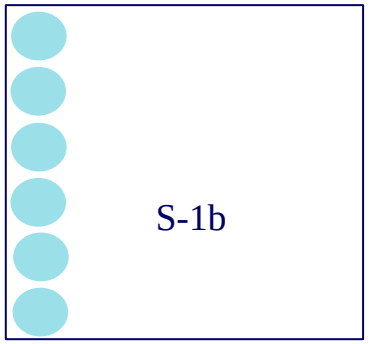
[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

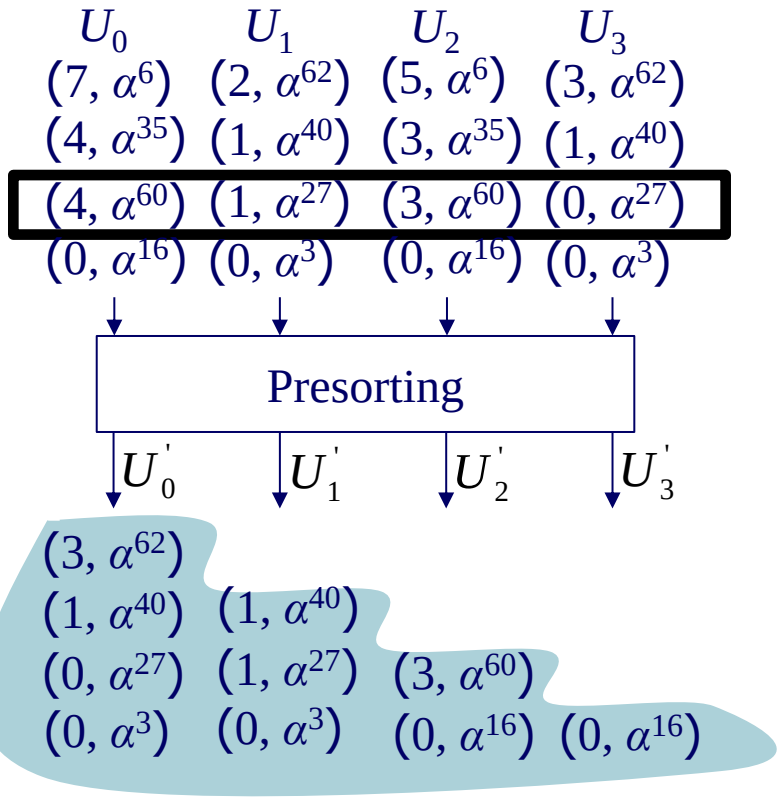


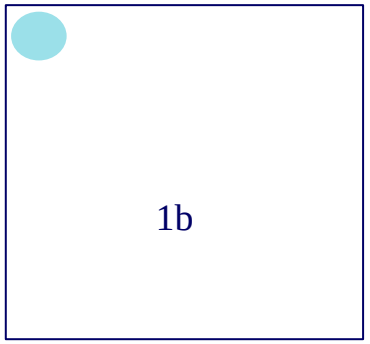


[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.
[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: State-of-the-art: Presorting Algorithm

$d_c = 4$, $n_m = 4$, $\alpha \in$ GF(64)

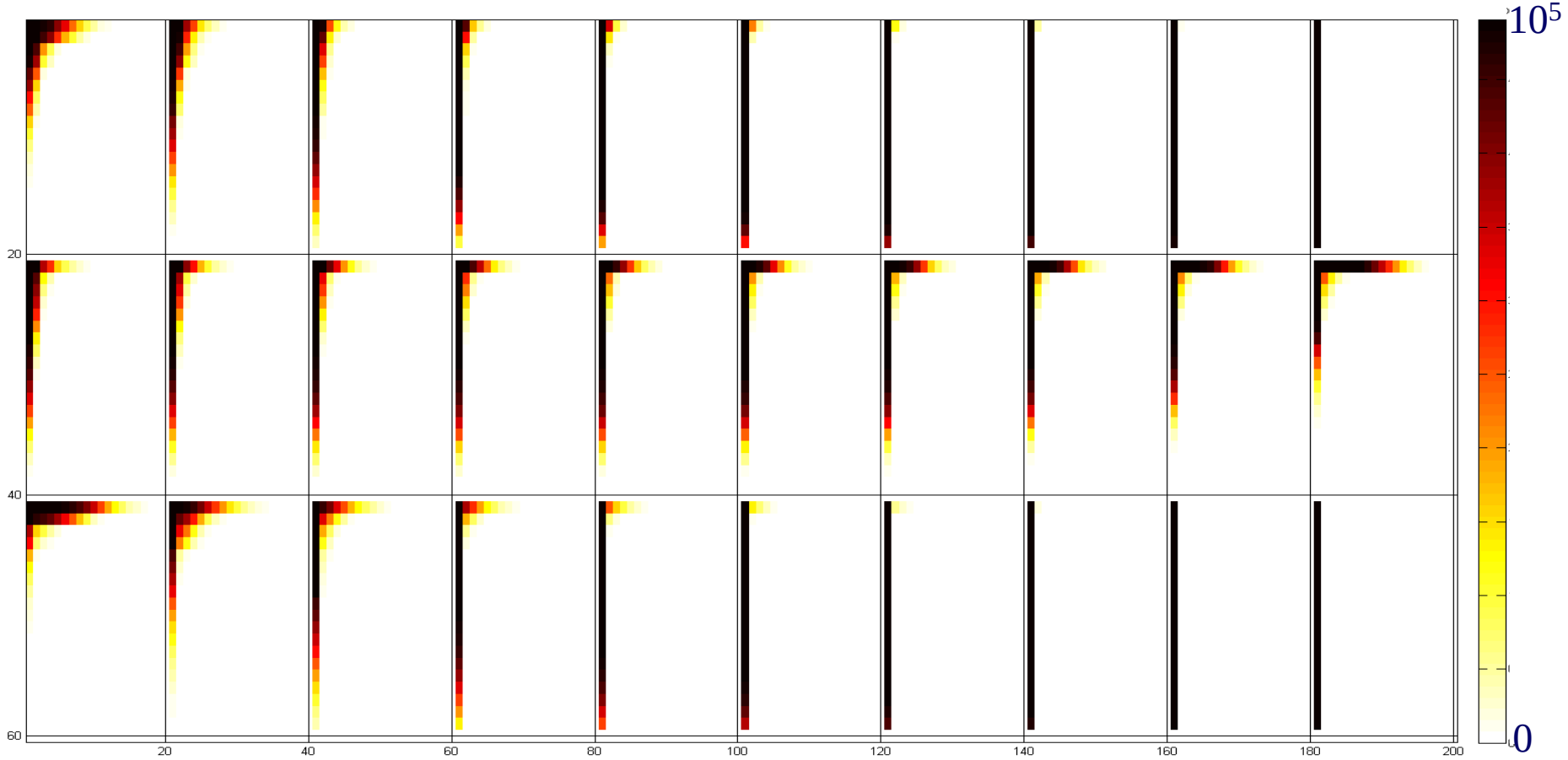




[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.
[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: Proposed: FB-CN with Presorting

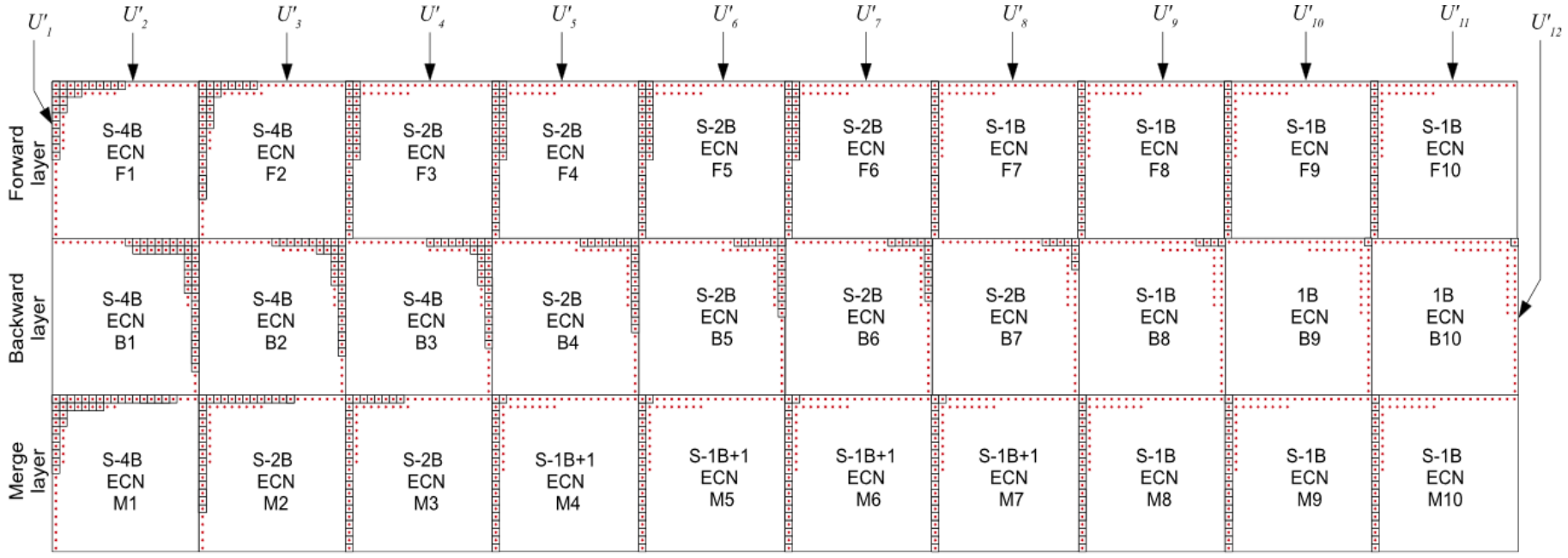
d_c = 12 [14]



[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: Proposed: FB-CN with Presorting

d_c = 12 [14]



FB-CN: 1680 bubbles (red bubbles)

S-FB: 648 bubbles (square bubbles)

[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

Lab-STICC

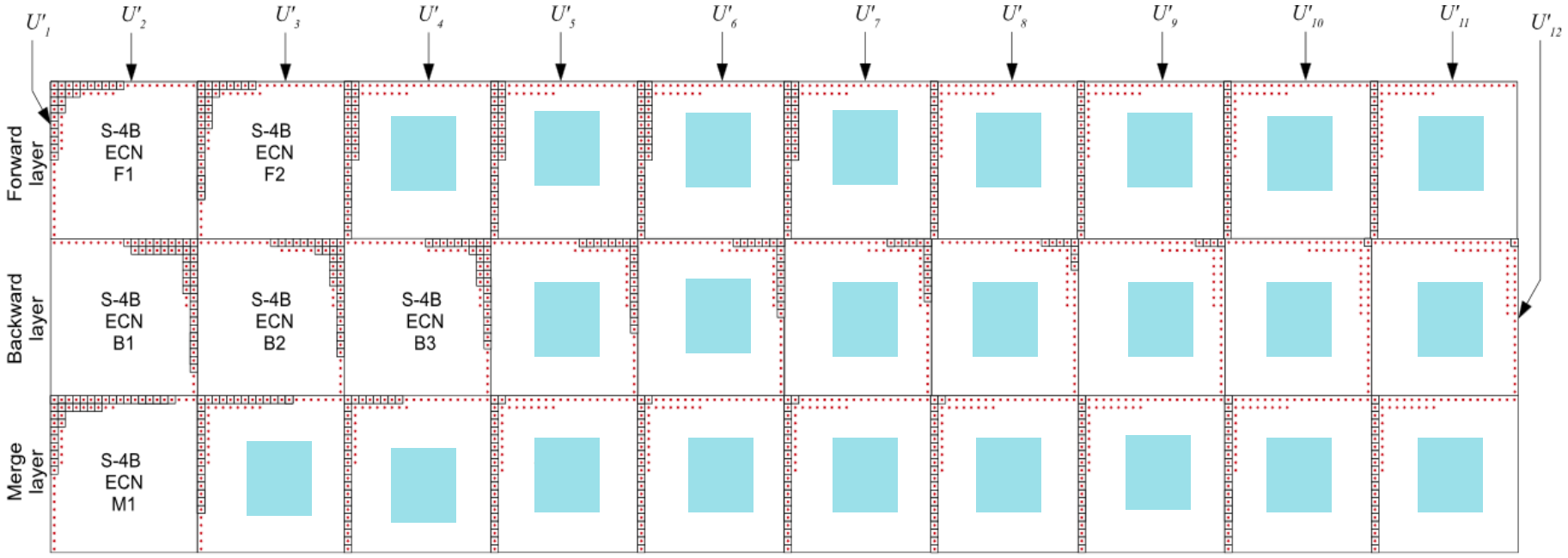
cnrs

SB->EF

CN and Presorting: Proposed: FB-CN with Presorting

d_c = 12 [14]

26 ECNs ≡ 86.6 %



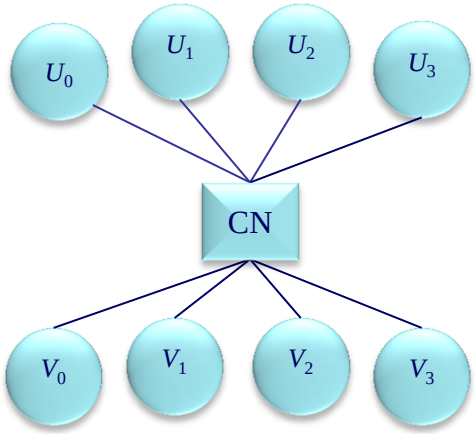
FB-CN: 1680 bubbles (red bubbles)

S-FB: 648 bubbles (square bubbles)

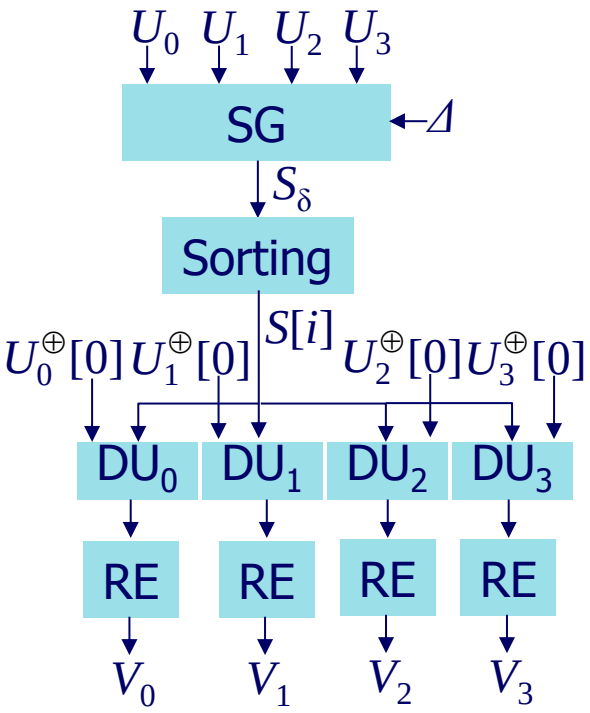
[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

CN and Presorting: Proposed: Extended Forward CN (EF-CN)

$d_c = 4$



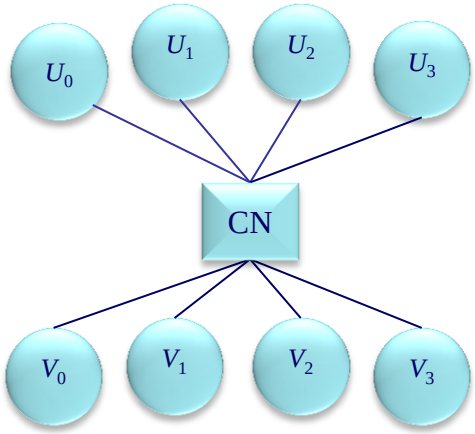
SB-CN [17]

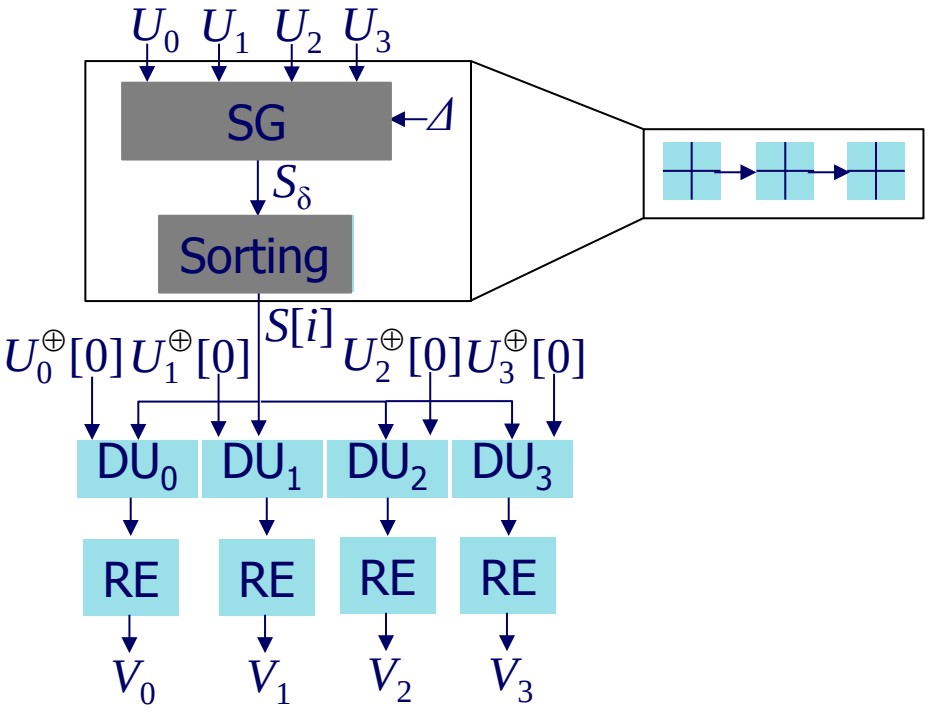


[17] P Schl¨afer, N Wehn, M Alles, T Lehnigk-Emden, E Boutillon. "*Syndrome based check node processing of high order NB-LDPC decoders*". International Conference on Telecommunications, Apr 2015, Sydney, Australia.

CN and Presorting: Proposed: Extended Forward CN (EF-CN)

$d_c = 4$

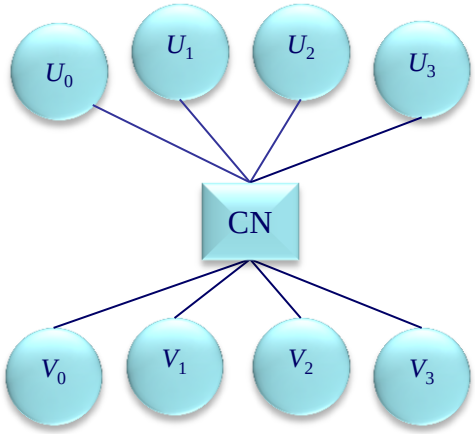




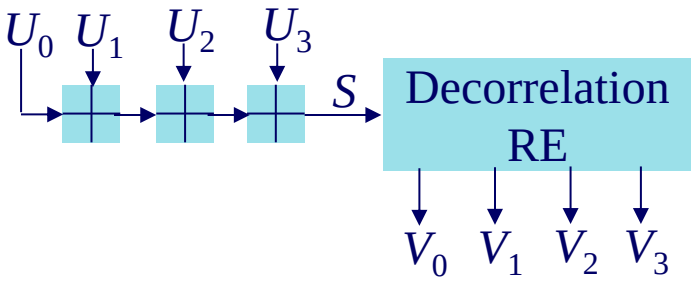
[18] Cédric Marchand, Emmanuel Boutillon, Hassan Harb, Laura Conde-Canencia and Ali Al Ghouwayel, "Hybrid Check Node Architectures for NB-LDPC Decoders", Accepted in IEEE Transactions on Circuits And Systems-I, August 2018.

CN and Presorting: Proposed: EF-CN and Hybrid CN (H-CN)

$d_c = 4$



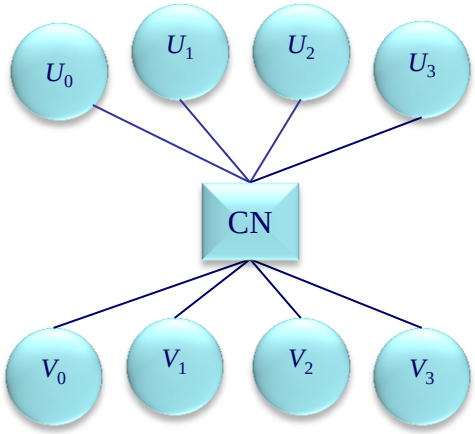
Extended Forward CN (EF-CN)



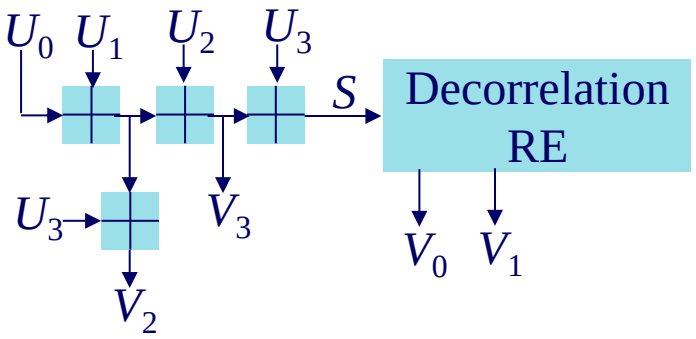
[18] Cédric Marchand, Emmanuel Boutillon, Hassan Harb, Laura Conde-Canencia and Ali Al Ghouwayel, "Hybrid Check Node Architectures for NB-LDPC Decoders", Accepted in IEEE Transactions on Circuits And Systems-I, August 2018.

CN and Presorting: Proposed: EF-CN and Hybrid CN (H-CN)

$d_c = 4$

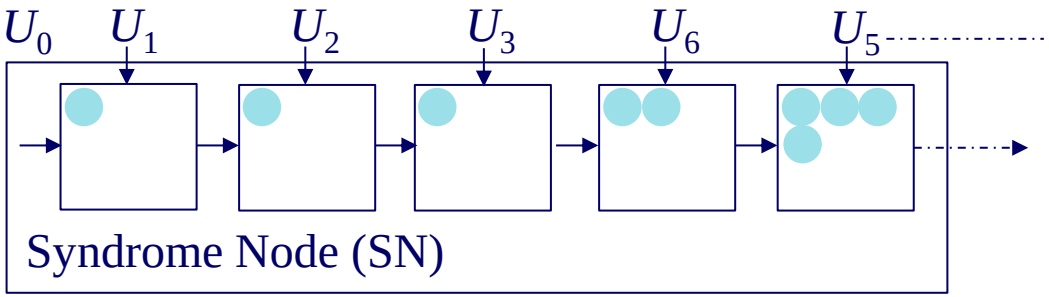


Hybrid CN (H-CN)



[18] Cédric Marchand, Emmanuel Boutillon, Hassan Harb, Laura Conde-Canencia and Ali Al Ghouwayel, "Hybrid Check Node Architectures for NB-LDPC Decoders", Accepted in IEEE Transactions on Circuits And Systems-I, August 2018.

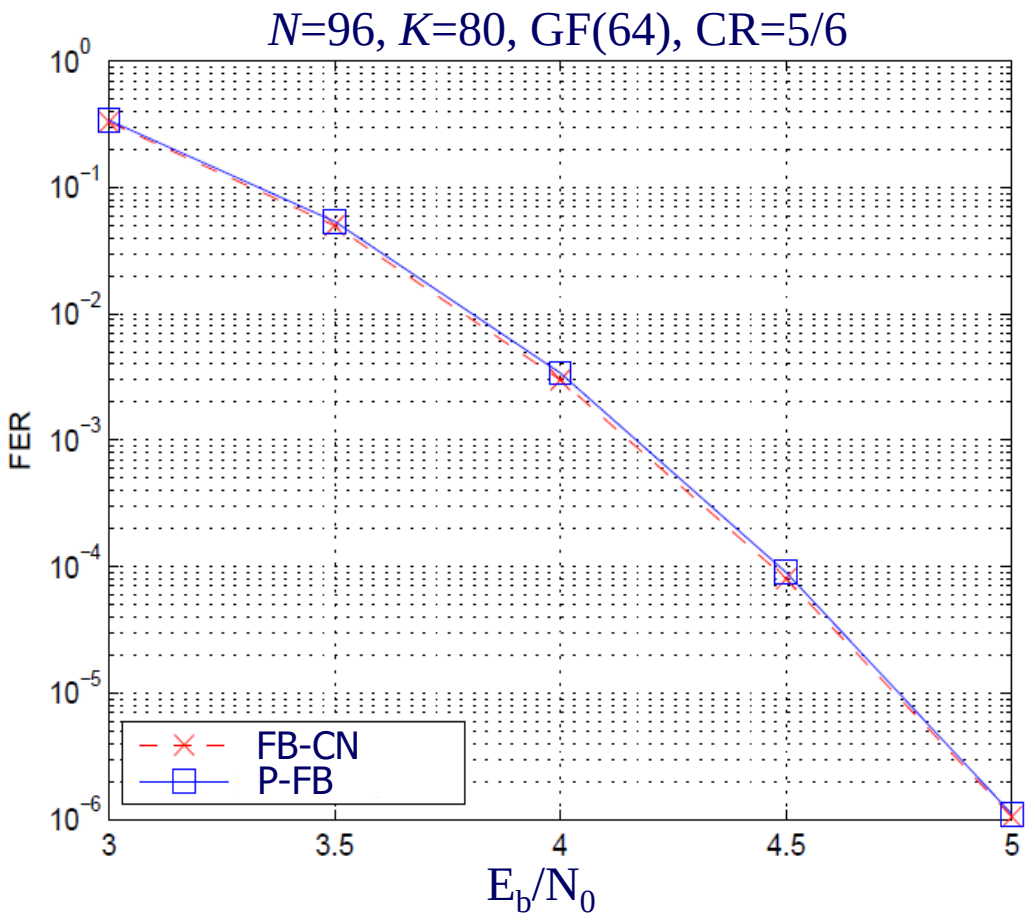
CN and Presorting: Proposed: H-CN with Presorting



Latency reduction

[18] Cédric Marchand, Emmanuel Boutillon, Hassan Harb, Laura Conde-Canencia and Ali Al Ghouwayel, "Hybrid Check Node Architectures for NB-LDPC Decoders", Accepted in IEEE Transactions on Circuits And Systems-I, August 2018.

CN and Presorting: Performance and Synthesis Analysis



[15] J. O. Lacruz, F. Garcia-Herrero, M. J. Canet, and J. Valls. "*Reduced-Complexity Non-Binary LDPC Decoder for High-Order Galois Fields Based on Trellis Min-Max Algorithm*". IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 24, no. 8, pp. 2643-2653, Aug 2016.

CN and Presorting: Performance and Synthesis Analysis

Post synthesis results on a Xilinx Virtex 6 FPGA

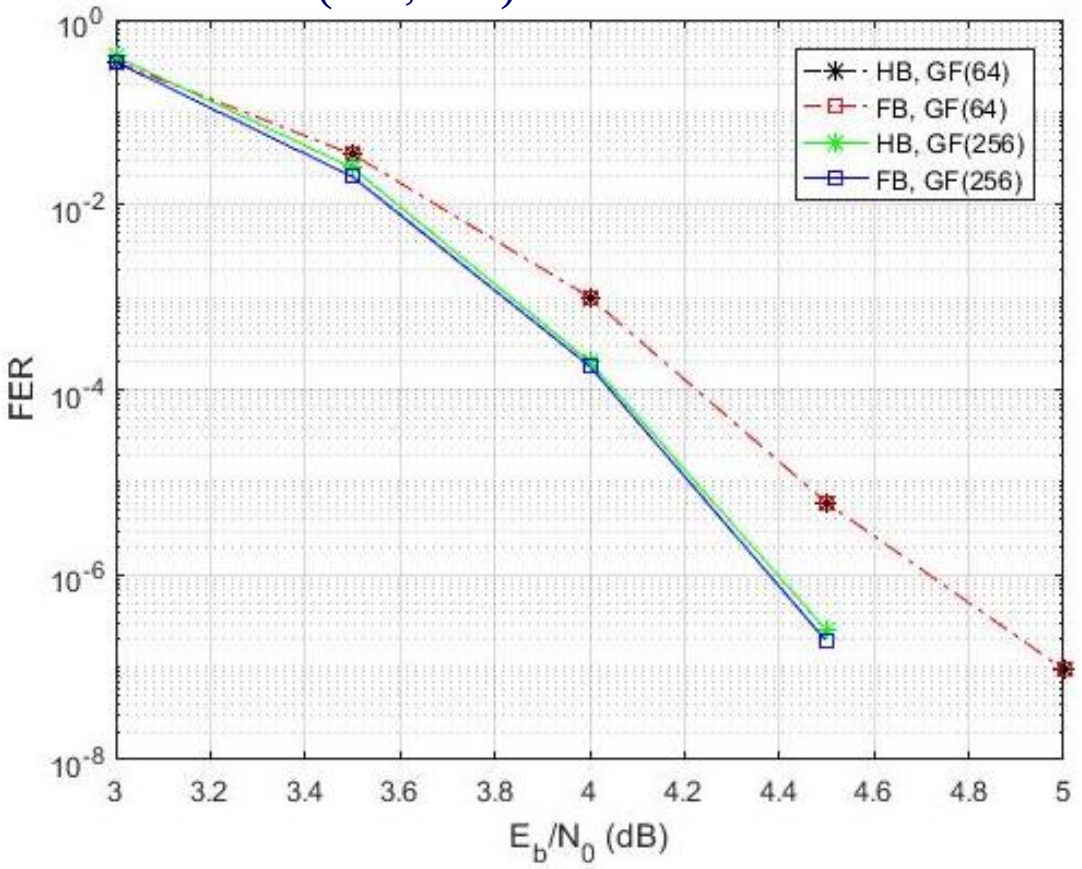
ECN	OS	F (MHz)
1B	7	714
S-1B	17	714
S-1B+1	35	349
S-2B	82	334
S-4B	138	269

		OS				
d_c	Case	Sorter	Switch	CN	Total	Gain
6	FB-CN	-	-	1617	1617	5%
	S-FB	50	93	1268	1532	
8	FB-CN	-	-	2481	2481	17%
	S-FB	77	142	1701	2061	
12	FB-CN	-	-	4666	4666	43%
	S-FB	160	283	1858	2653	
20	FB-CN	-	-	6519	6519	54%
	S-FB	386	495	1232	2955	

P_{clk}: Critical path
CL: Cycle Latency
$T_{CN} = F_{clk}$/CL(layer): Number of computed CN per second
AE=T_{CN}/Area: Area Efficiency
EE=T_{CN}/Power: Energy Efficiency

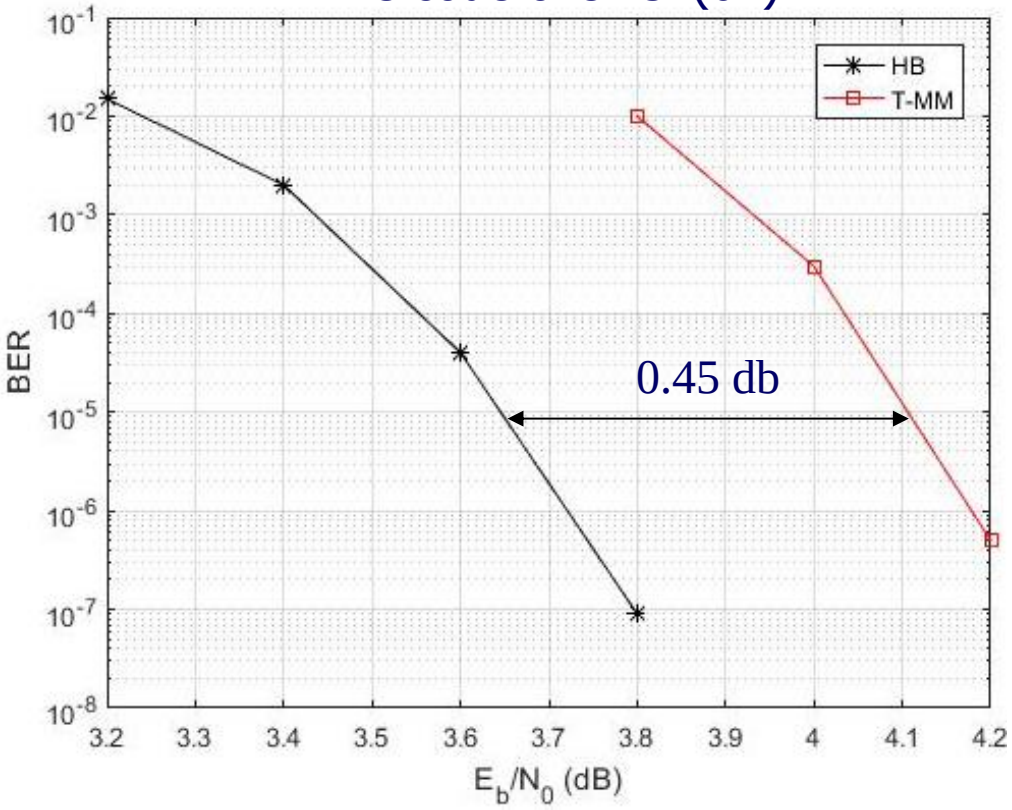
CN and Presorting: Performance and Synthesis Analysis

(144, 120) NB-LDPC code



CN and Presorting: Performance and Synthesis Analysis

BER performance for a (1536, 1344) NB-LDPC code over GF(64)



T-MM: Trellis Min-MAX algorithm [15]

[15] J. O. Lacruz, F. Garcia-Herrero, M. J. Canet, and J. Valls. "*Reduced-Complexity Non-Binary LDPC Decoder for High-Order Galois Fields Based on Trellis Min-Max Algorithm*". IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 24, no. 8, pp. 2643-2653, Aug 2016.

CN and Presorting: Performance and Synthesis Analysis

Post-synthesis results for CN architectures on 28 nm FD-SOI technology

d_c=12

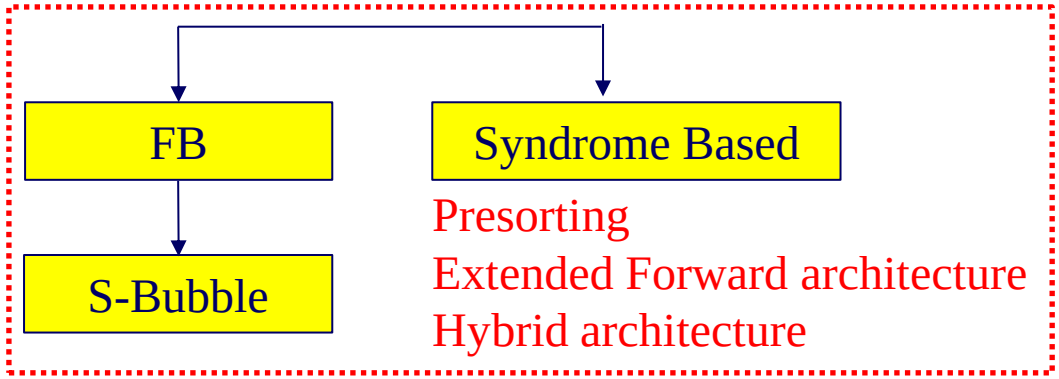
CN GF(64)	Area (mm^2)	Power (mW)	P_{clk} (ns)	CL(CN) (cycles)
FB-CN	0.140	94	1.02	22
H-CN Presorted	0.0227	14.9	1.03	15

Factor Gain 6.16

CN GF(256)	Area (mm^2)	Power (mW)	P_{clk} (ns)	CL(CN) (cycles)
FB-CN	0.328	210	1.14	22
H-CN Presorted	0.0753	45	1.2	16

Factor Gain 4.35

CN and Presorting: Conclusion



[14] H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.

[18] Cédric Marchand, Emmanuel Boutillon, Hassan Harb, Laura Conde-Canencia and Ali Al Ghouwayel, "Hybrid Check Node Architectures for NB-LDPC Decoders", Accepted in IEEE Transactions on Circuits And Systems-I, August 2018.

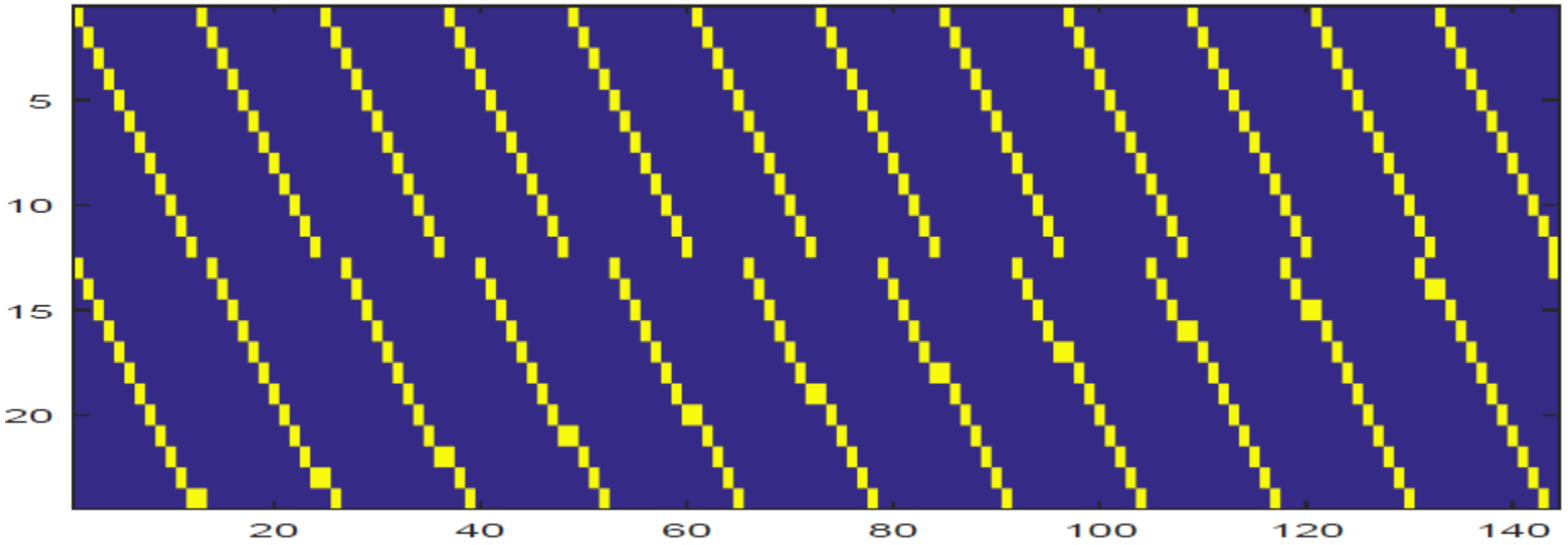
Outline

- Introduction
- NB-LDPC Codes
- EMS Algorithm and Architectures
- Proposed Parallel and Pipelined NB-LDPC Decoder Architecture
- Extra Related Works
- Conclusion, Perspectives and Publications

Used PCM

Base matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \end{bmatrix}$$



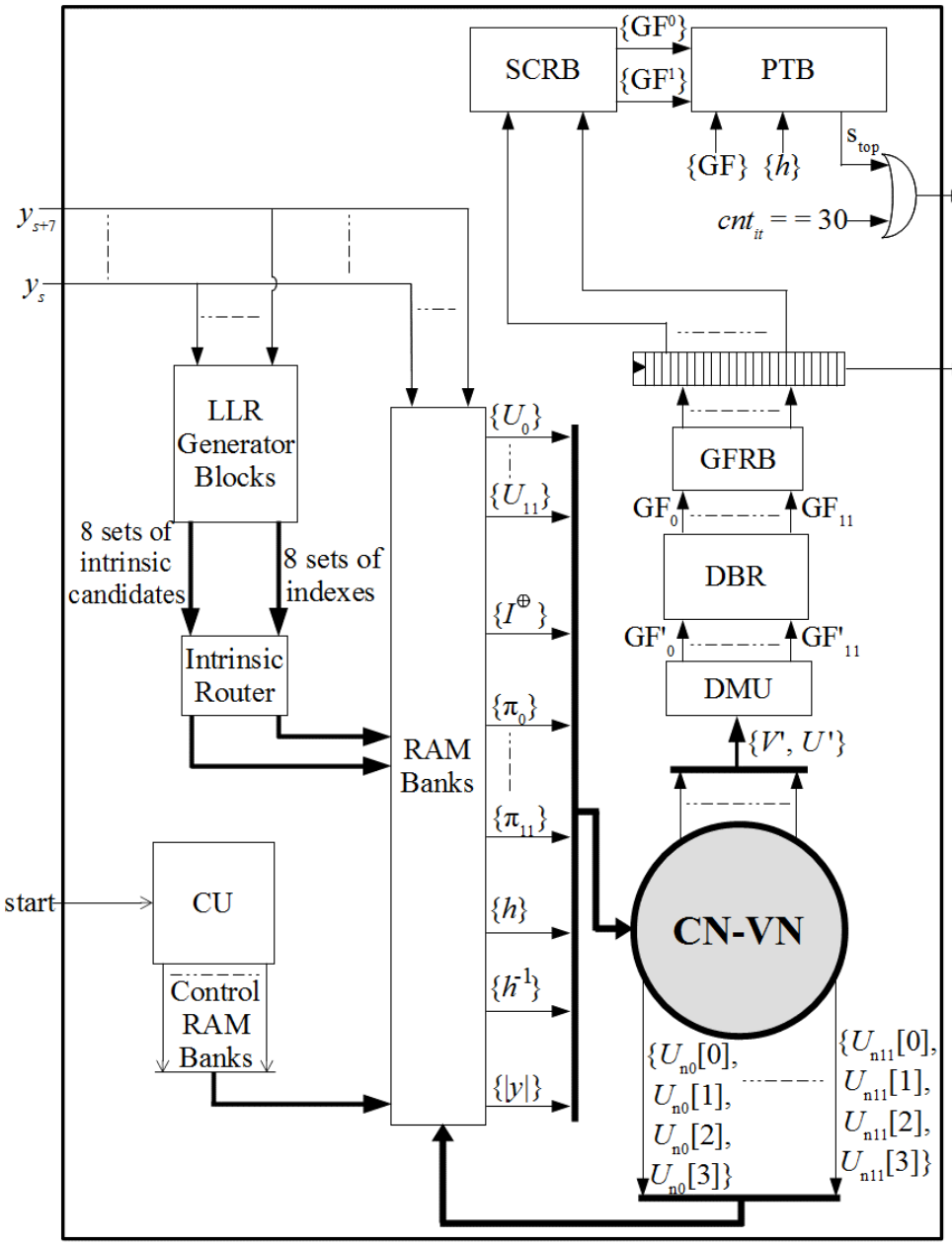
Topology of H

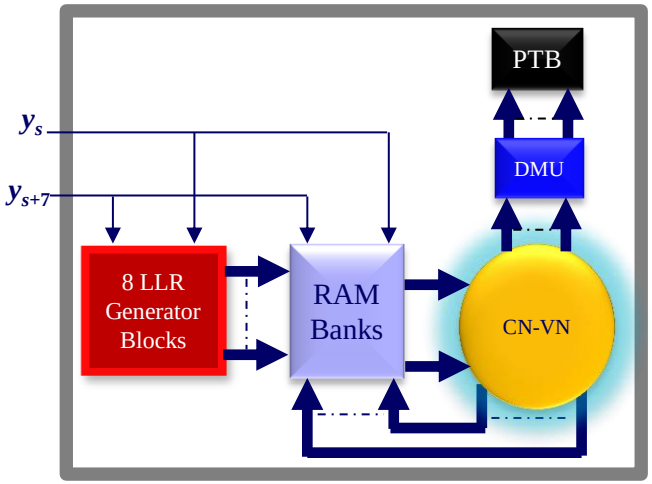
$N = 144$, $M = 24$, $d_c = 12$, $d_v = 2$, CR=5/6.

2 Layers.

http://www-labsticc.univ-ubs.fr/nb_ldpc/

Global Decoder Architecture





CU: Control Unit

DMU: Decision Making Unit

DBR: Decision Block Reordering

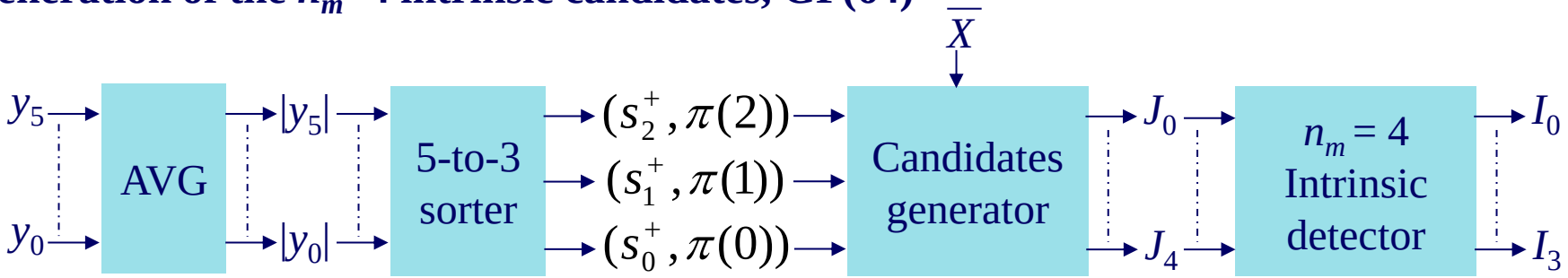
GFRB: GF Routing Block

SCRB: Stopping Criteria Router Block

PTB: Parity Test Block

Parallel Pipelined LLR Generator Architecture

Generation of the n_m=4 intrinsic candidates, GF(64)



LLR generator	n_m	OS	F (MHz)	Periodicity (CCs)		Factor gain efficiency	Factor gain throughput
				Proposed	[10]		
Proposed	12	516	402	1	8	4	15.3
	4	167	556		1	2.15	2.65
[10]	All cases	137	210				

Efficiency (MHz/OS) = F/(OSxPeriodicity)

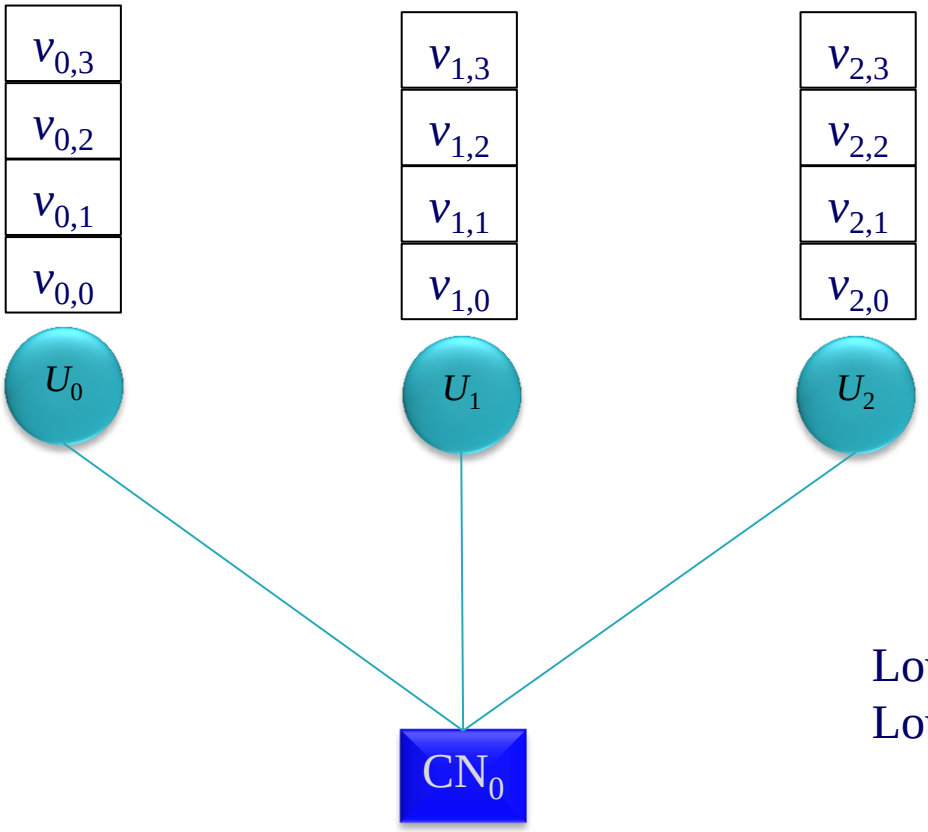
Throughput (Msymbols/s) = (F x n_m)/Periodicity

OS: Occupied slices
F: Frequency in MHz
CCs: Clock Cycles
$\underline{Y}$: Observed Symbol
X: Hard Decision on Y

[10] A.A. Ghouwayel and E. Boutillon. 2011.

Serial Processing

GF(64), $n_{m\_in} = 4$, $n_{m\_out} = 20$



Low Throughput
Low area consumption

n_{m_in}: Length of each CN input vector
n_{m_out}: Length of each CN output vector

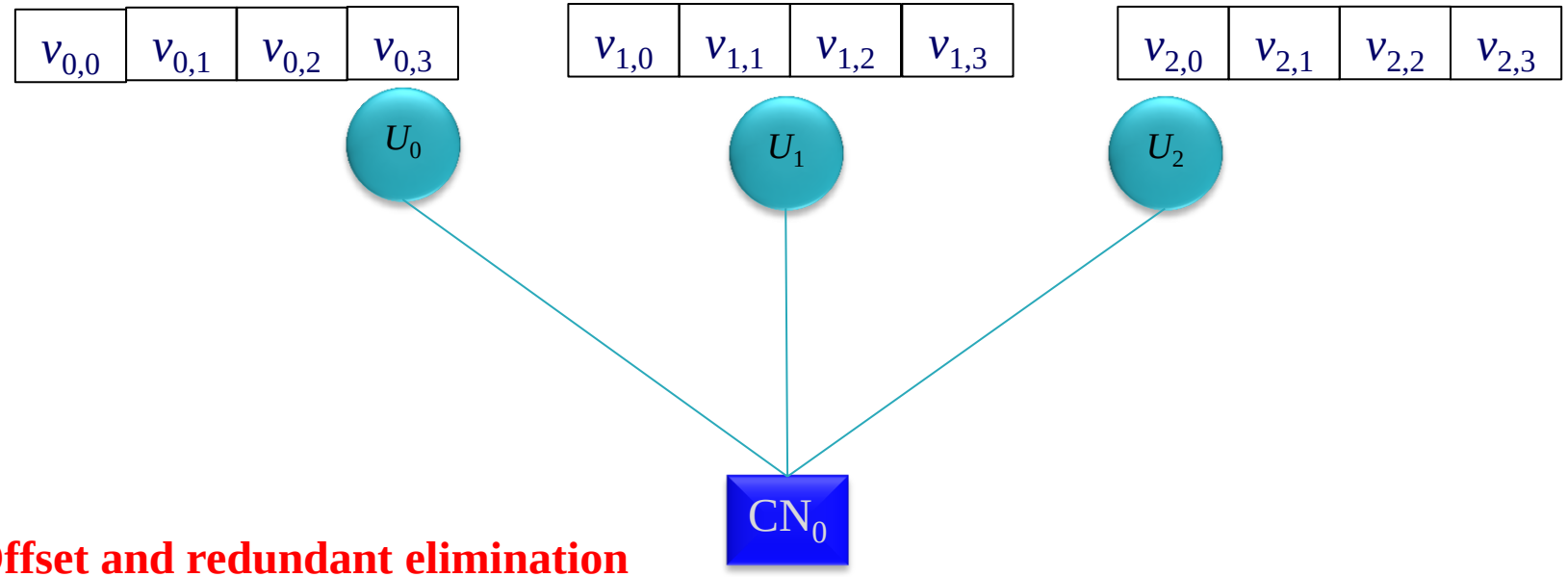
Parallel Processing

GF(64), $n_{m\_in} = 4$, $n_{m\_out} = 20$

Presorting [13]

Number of processed symbols in the CN << $d_c x n_m$

$d_c x n_{m\_in}$

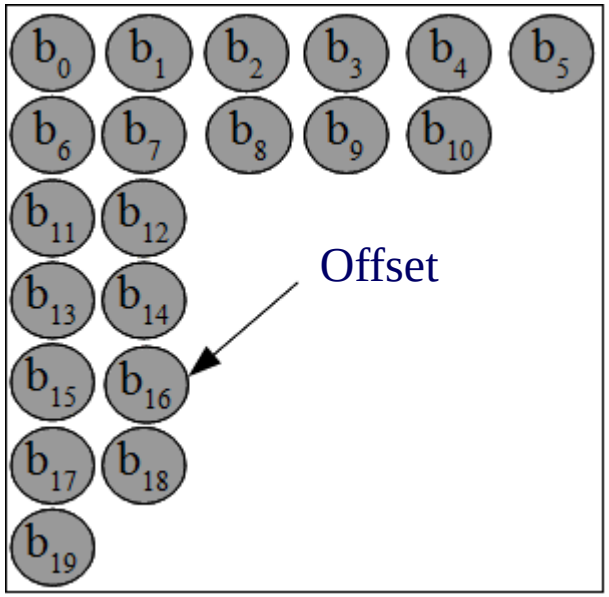


Offset and redundant elimination

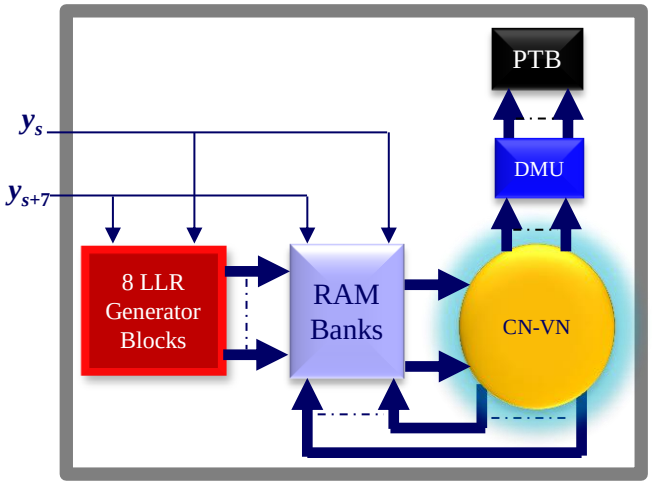
n_{m_in}: Length of each CN input vector
n_{m_out}: Length of each CN output vector

[13] C. Marchand and E. Boutillon. "NB-LDPC check node with pre-sorted input. In 9th International Symposium on Turbo Codes & Iterative Information Processing, September 2016.

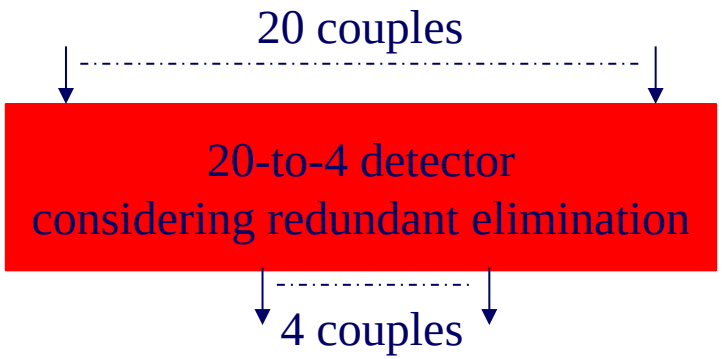
Predefined Offset

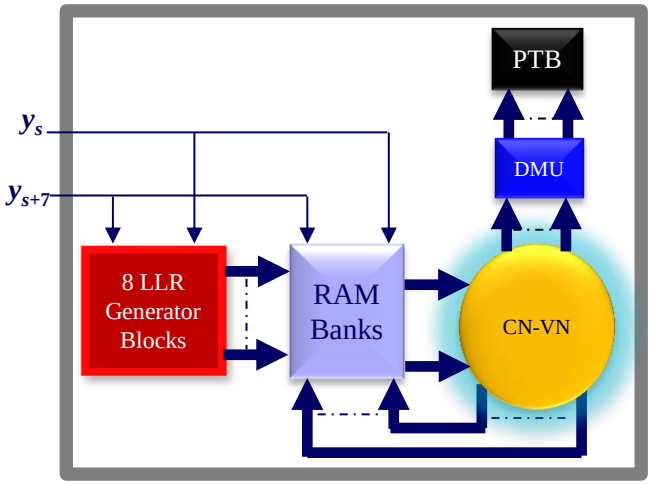


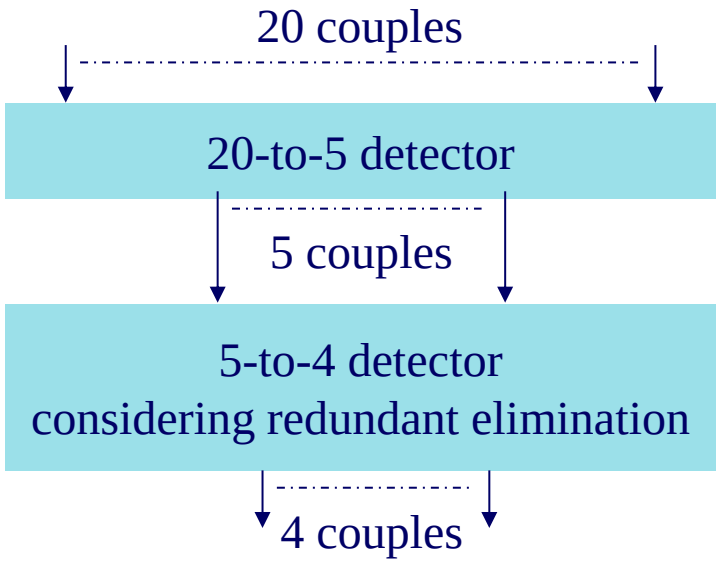
Predefined Offset



Redundant Elimination

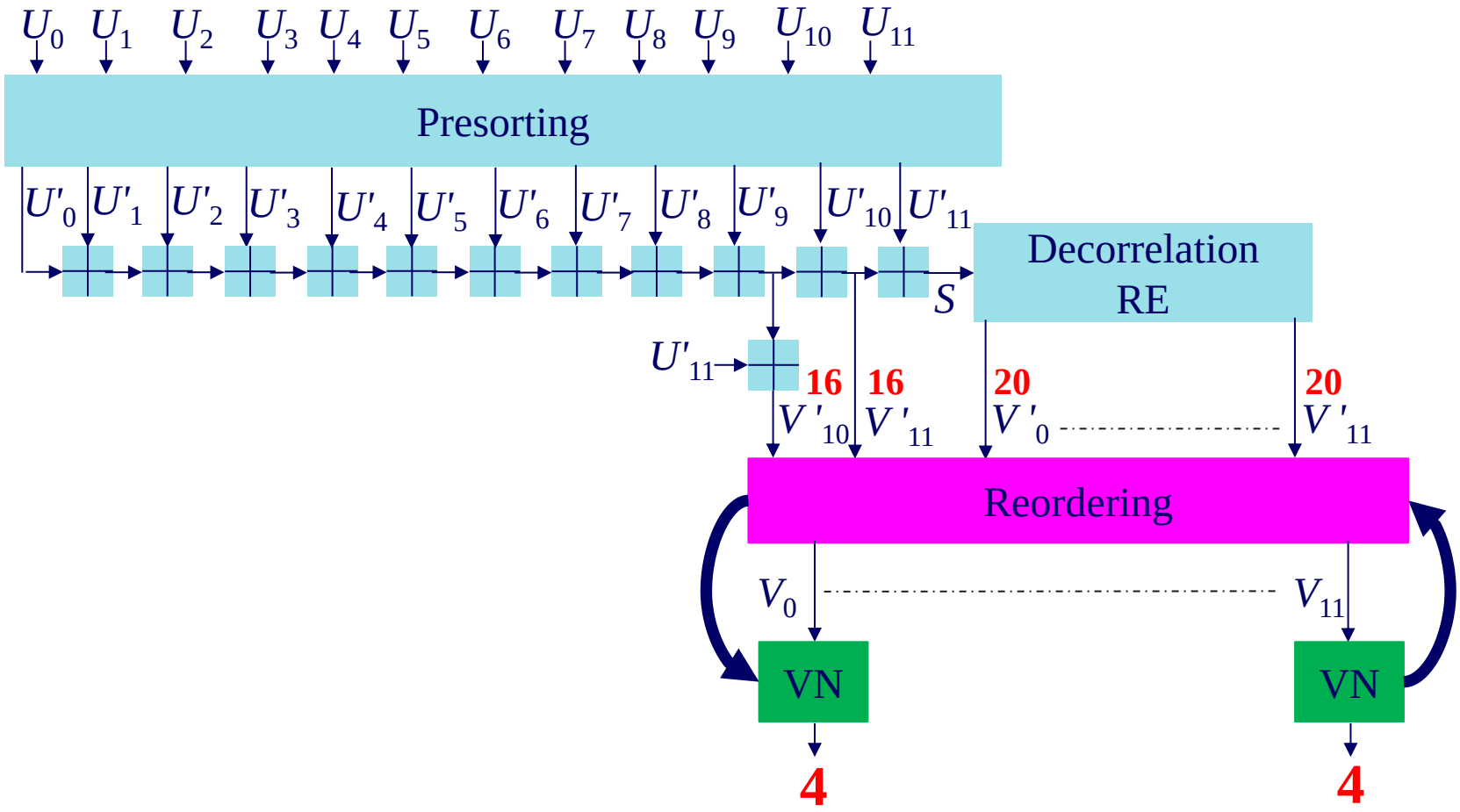






H-CN with Presorting: Parallel Approach

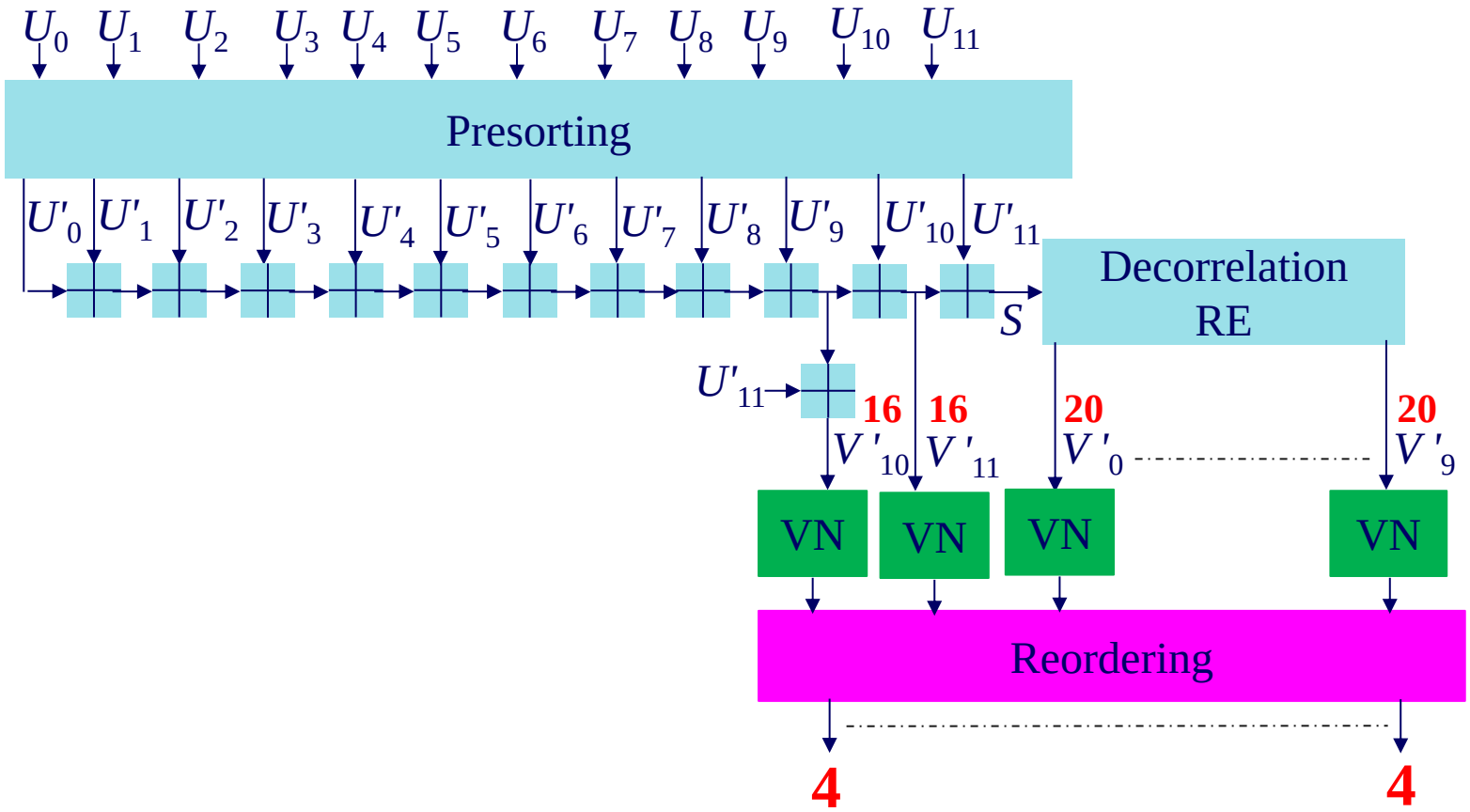
CN block, d_c = 12



[18] Cédric Marchand, Emmanuel Boutillon, Hassan Harb, Laura Conde-Canencia and Ali Al Ghouwayel, "Hybrid Check Node Architectures for NB-LDPC Decoders", Accepted in IEEE Transactions on Circuits And Systems-I, August 2018.

H-CN with Presorting: Parallel Approach

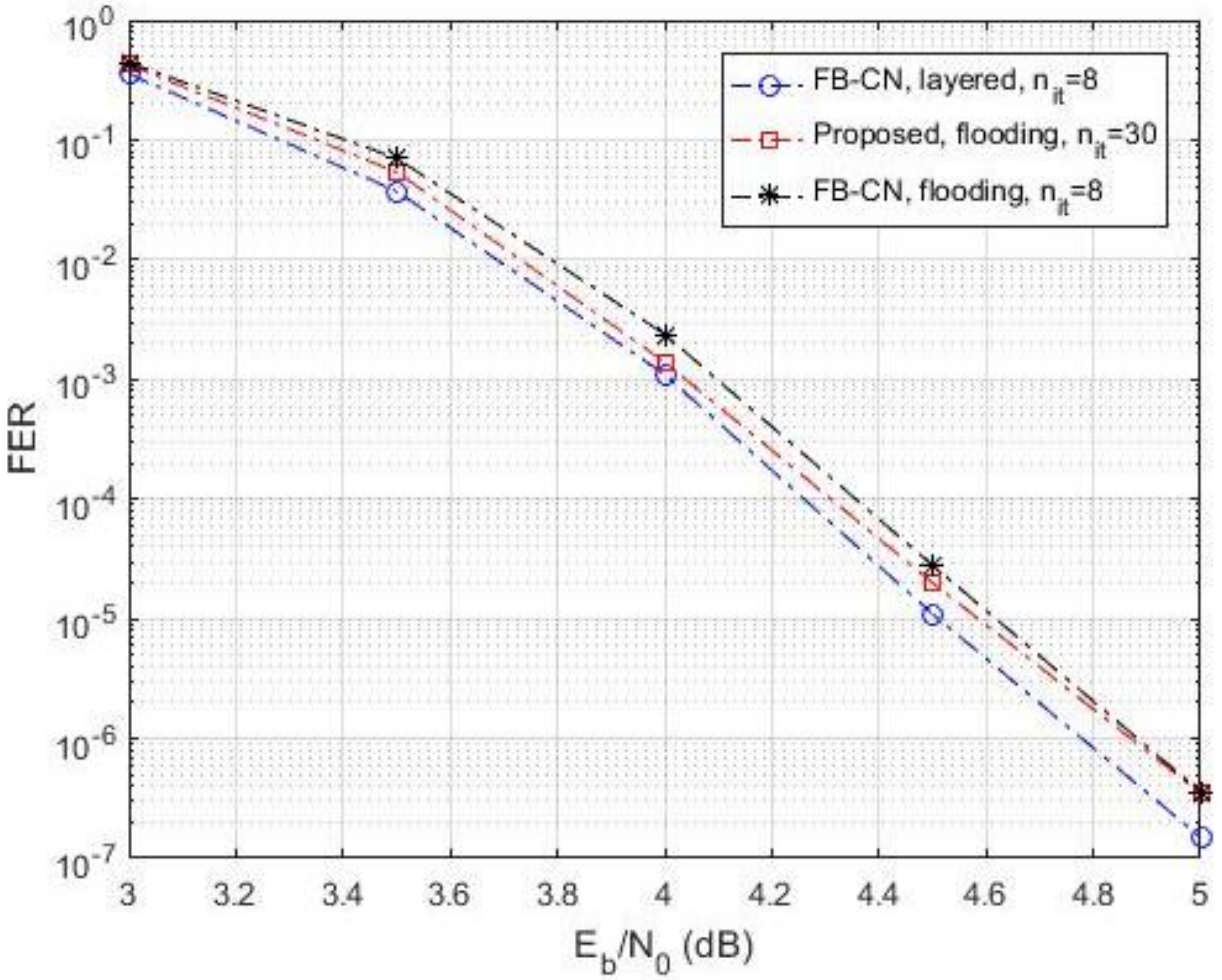
CN block, d_c = 12



Number of reordered symbols is reduced from 272 down to 48 symbols

[18] Cédric Marchand, Emmanuel Boutillon, Hassan Harb, Laura Conde-Canencia and Ali Al Ghouwayel, "Hybrid Check Node Architectures for NB-LDPC Decoders", Accepted in IEEE Transactions on Circuits And Systems-I, August 2018.

Simulation, Emulation and Throughput Results



y_i = sat((floor(Qxa_i/σ)+0.5), Q)

Q=15

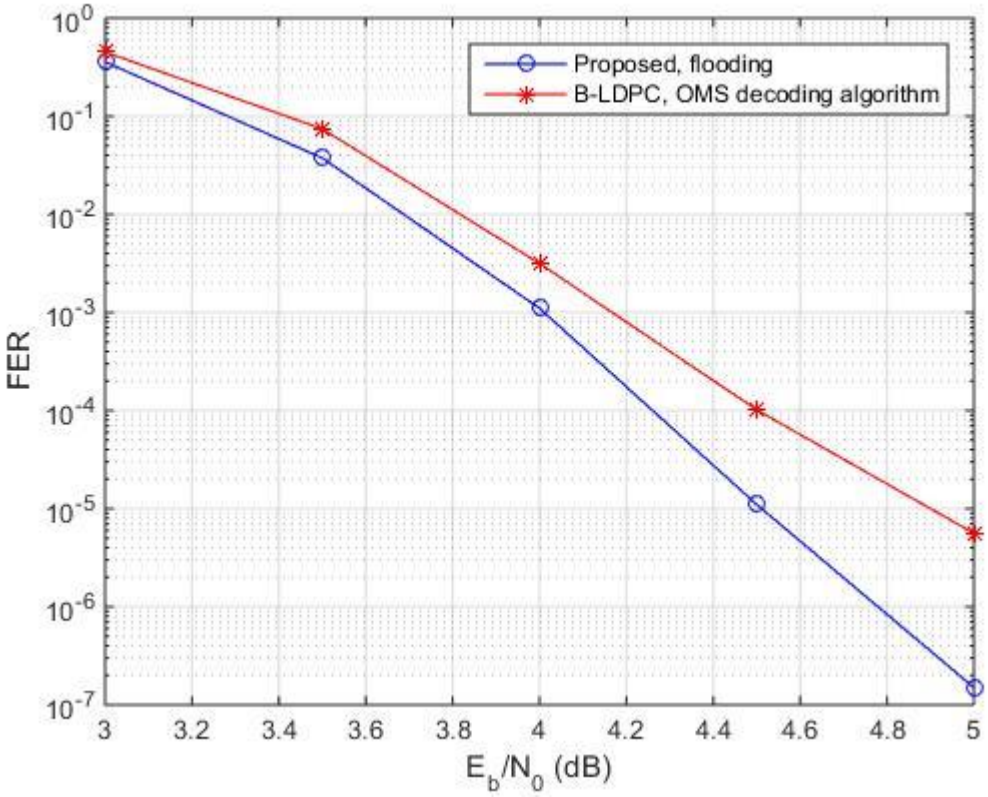
Sat(a, b) = b if a > b

-b if a < -b

a Otherwise

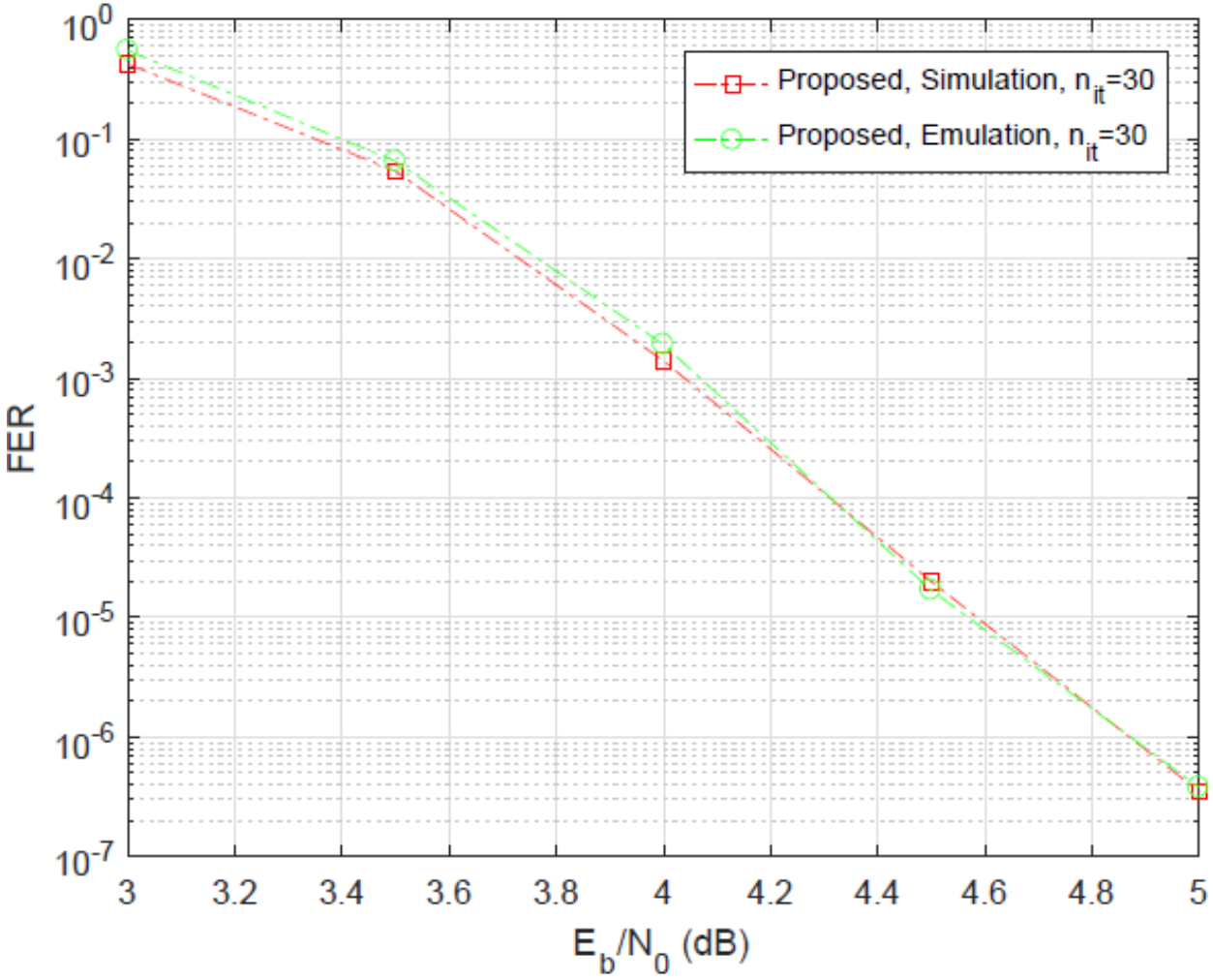
n_{it}: Number of iterations

Simulation, Emulation and Throughput Results



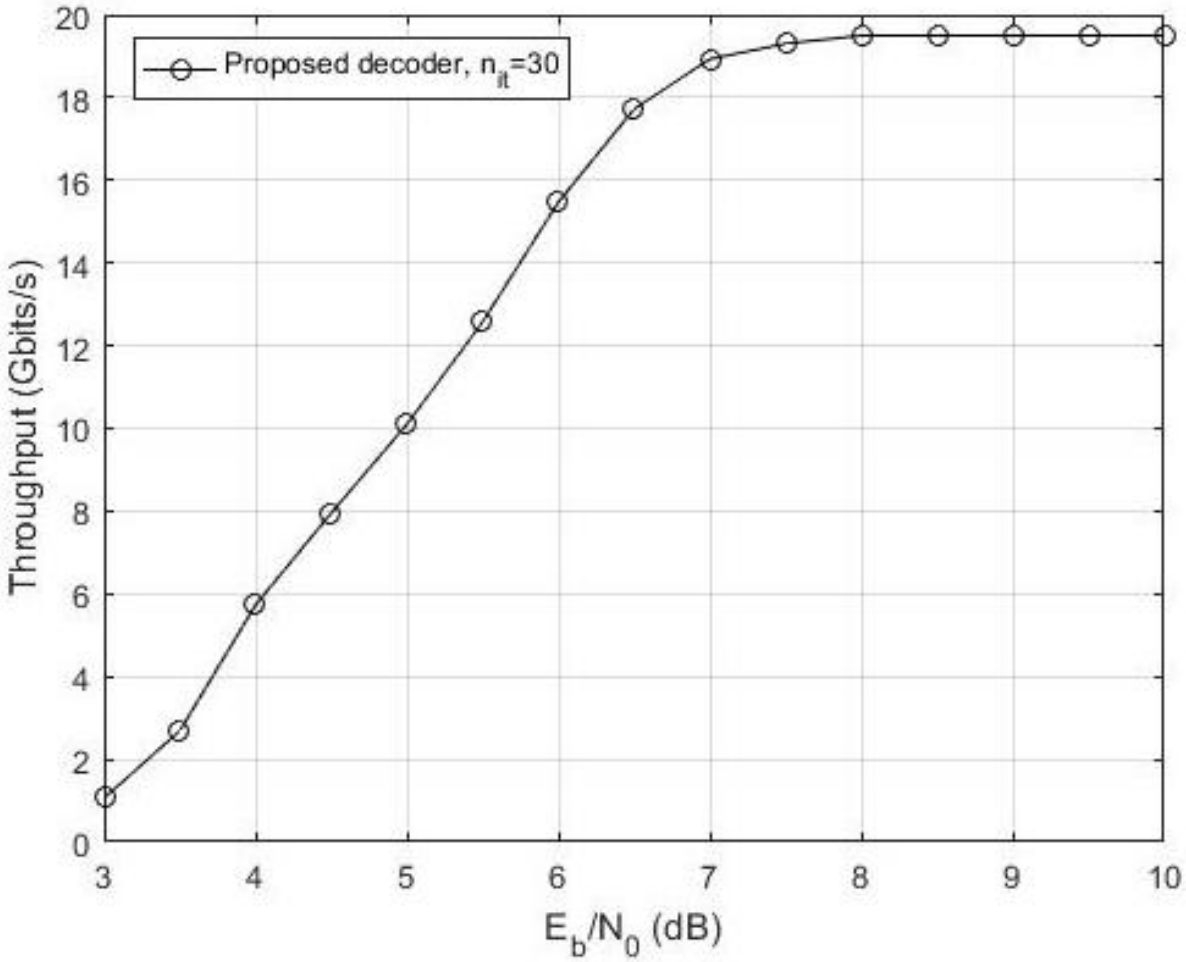
n_{it} = 30
OMS: Offset-Min Sum

Simulation, Emulation and Throughput Results



Name: Bertrand
Surname: Le Gal
Occupation: Doctor at IMS LAB in Bordeaux.

Simulation, Emulation and Throughput Results



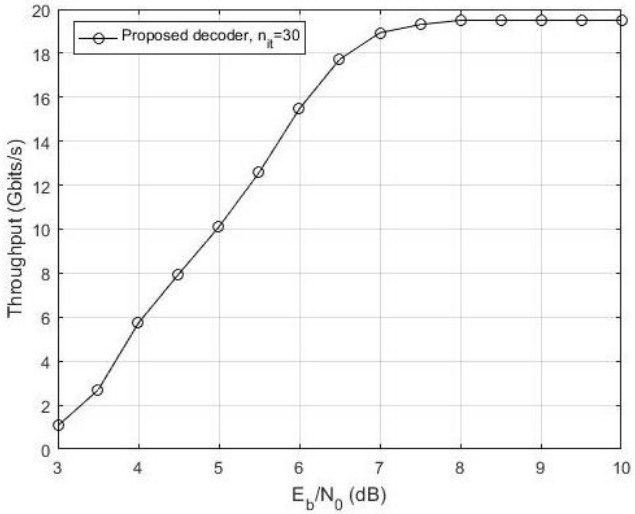
Throughput (Gbits/s)

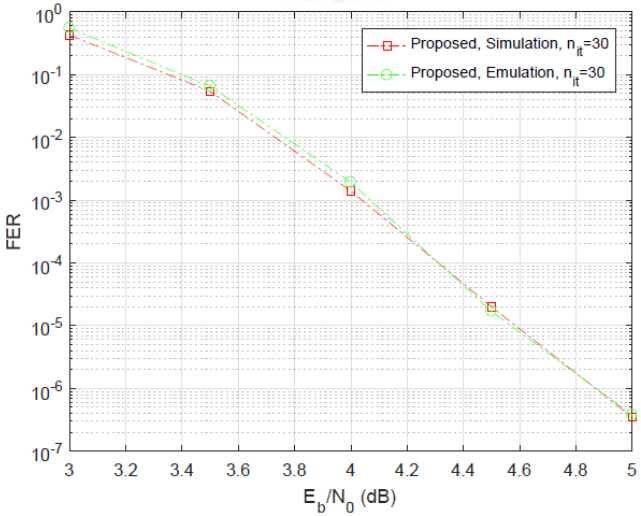
$$= \frac{\log_2(q) \times K \times F}{10^3 \times a_{it} \times M}$$

a_{it}: Average number of iterations

Synthesis Results

	[24]	[25]	[26]	Proposed
Technology	40 nm	90 nm	65 nm	28 nm
Design	Synthesis	Synthesis	Silicon	Synthesis
N (symbols)	3888	837	160	144
CR	8/9	13/15	1/2	5/6
GF	4	32	64	64
Decoding Algorithm	T-EMS	IL-MwBRB	EMS	EMS
Decoding schedule	Layered	-	Flooding	Flooding
Gate Count (NANDs)	4M	4.54M	2.78M	0.79
Frequency (MHz)	1000	207.04	700	650
Iterations	10	10	10-30	1-30
Throughput (Mb/s)	3600	21661.56	1221	1600-19500
Throughput Efficiency (Mbps/M-gate)	900	4771.27	439.2	2025-24683





[24] Erbao Li, D. Declercq, and K. Gunnam. Trellis-Based Extended Min-Sum Algorithm for Non-Binary LDPC Codes and its Hardware Structure. Communications, IEEE Transactions on, 61(7) :2600-2611, July 2013.

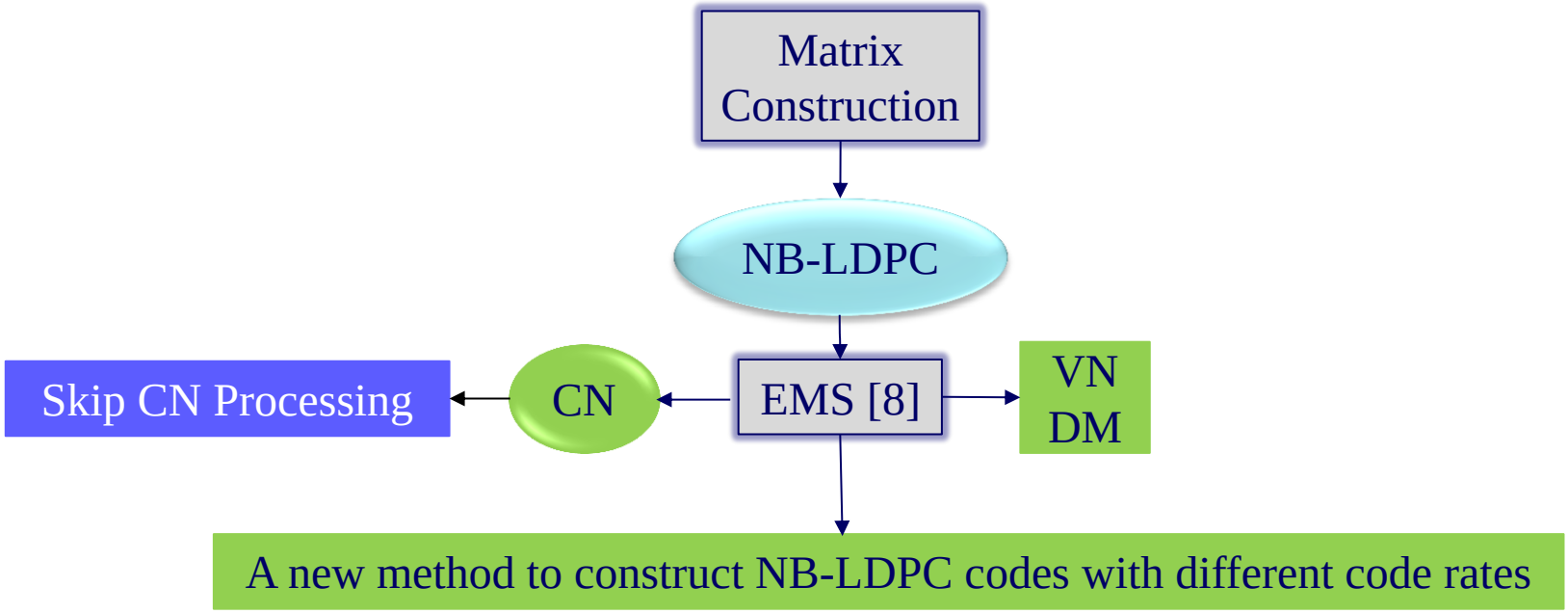
[25] J. Tian, J. Lin, and Z. Wang. A 21.66Gbps Non-Binary LDPC Decoder for High-Speed Communications. IEEE Transactions on Circuits and Systems II: Express Briefs, vol. PP, no. 99, pp. 1-1, 2017.

[26] Y. S. Park, Y. Tao, and Z. Zhang. A Fully Parallel Nonbinary LDPC Decoder With Fine-Grained Dynamic Clock Gating. IEEE Journal of Solid-State Circuits, vol. 50, no. 2, pp. 464-475, Feb 2015.

Outline

- Introduction
- NB-LDPC Codes
- EMS Algorithm and Architectures
- Proposed Parallel and Pipelined NB-LDPC Decoder Architecture
- Extra Related Works
- Conclusion, Perspectives and Publications

Extra Related Works



Proposition of a new sorting algorithm to select the two extrema values from a set of cardinality *Ns*.

Outline

- Introduction
- NB-LDPC Codes
- EMS Algorithm and Architectures
- Proposed Parallel and Pipelined NB-LDPC Decoder Architecture
- Extra Related Works
- Conclusion, Perspectives and Publications

Conclusion

NB-LDPC

Code:

- Matrix construction
- NB-LDPC codes construction

Sorter Algorithm

Hardware design:

- CN:
  - Extended Forward
  - Hybrid
  - Presorted Forward Backward
  - Presorted Extended Forward
  - Presorted Hybrid
  - Skip processing CNs
- Parallel pipelined NB-LDPC decoder
- Variable Node and Decision Making blocks

PhD Objective met ✓.

Low complexity and high throughput NB-LDPC decoder has been developed

Perspectives

- Optimization of parallel Hybrid architecture for all value of d_c and GF(q).
- Automatic generation of the hardware architecture from a given NB-LDPC matrix.
- Merging of the H-CN and the T-EMS algorithm.
- Applying the acquired knowledge of the NB-LDPC codes on the Turbo codes.

Publications
(Accepted, submitted and in preparation)

1. H. Harb, C. Marchand, A. A. Ghouwayel, L. Conde-Canencia, and E. Boutillon, "Pre-sorted forward-backward NB-LDPC check node architecture," in IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2016, pp. 142-147, Dallas, USA.
2. Titouan Gendron, Hassan Harb, Alban Derrien, Cédric Marchand, Laura Conde-Canencia, Bertand Le Gal and Emmanuel Boutillon, "Demo: Construction of good Non-Binary Low Density Parity Check codes", Demo night at SIPS'2017, Lorient, France, Oct. 2017.
3. C. Marchand, H. Harb, E. Boutillon, A. Al Ghouwayel, and L. Conde-Canencia, "Extended-forward architecture for simplied check node processing in NB-LDPC decoders," in IEEE International Workshop on Signal Processing Systems (SiPS), October. 2017, Lorient, France.
4. Cédric Marchand, Emmanuel Boutillon, Hassan Harb, Laura Conde-Canencia and Ali Al Ghouwayel, "Hybrid Check Node Architectures for NB-LDPC Decoders", in IEEE Transactions on Circuits And Systems-I, August 2018.
5. Hassan Harb, Emmanuel Boutillon, Bertrand Le Gal, "Real-time evaluation of NBLDPC codes thanks to HLS-based hardware emulation", Demo night at DASIP'2018, Porto, Portugal, Oct. 2018.
6. Ali Al-Ghouwayel, Member, IEEE, Hassan Harb and Emmanuel Boutillon, Senior Member, IEEE, "First-Then-Second Extrema Selection,". Submitted.
7. Hassan Harb, Ali Al Ghouwayel, Cédric Marchand, Laura Conde-Canencia, Emmanuel Boutillon, "Throughput Rocket EMS NB-LDPC Decoder Based On A Parallel And Pipelined Architecture,". In preparation.
8. Hassan Harb, Ali Al Ghouwayel and Emmanuel Boutillon, "Parallel pipelined LLR generator,". In preparation.

THANK YOU

Emmanuel

Laura

Cedric

Ali Al Ghouwayel

Ali Alaeddine

Hassan