High Speed Decoding of Serial Concatenated Codes

Globecom 2006 CTH02-1

Doré Jean-Baptiste, Hamon Marie-Hélène and Pénard Pierre
{jeanbaptiste.dore;mhelene.hamon;pierre.penard}@orange-ftgroup.com

Outline

- Introduction
- S-SCP codes
  - Introduction
  - Decoding Strategy
- Joint strategy for decoding and code design
  - Memory contention
  - Towards a pipeline decoding
- Conclusion

Introduction

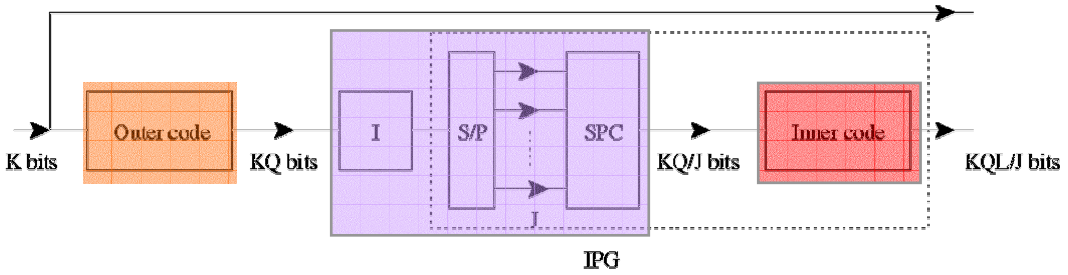
- Natural approach to design advanced coding schemes:
  - First design a code (LDPC codes, Turbo and Turbo like codes..):
    - For required system performance: threshold, minimal distance…
  - Find an efficient hardware architecture for such a code
    - In many case, such a constructed code has very little chance to be suited for hardware implementation…
- "Architecture driven" approach
  - For advanced coding scheme: First introduced for Turbo-Codes Interleaver design
    - Avoid memory contention when decoders are parallelized
  - Methodology extended for Turbo-like and LDPC codes
    - Increase throughput…
  - Joint code design and encoder/decoder architecture

Outline

- Introduction
- S-SCP codes
  - Introduction
  - Decoding Strategy
- Joint decoding strategy and code design
  - Memory contention
  - Towards a pipeline decoding
- Conclusion

S-SCP codes : Systematic with Serially Concatenated Parity

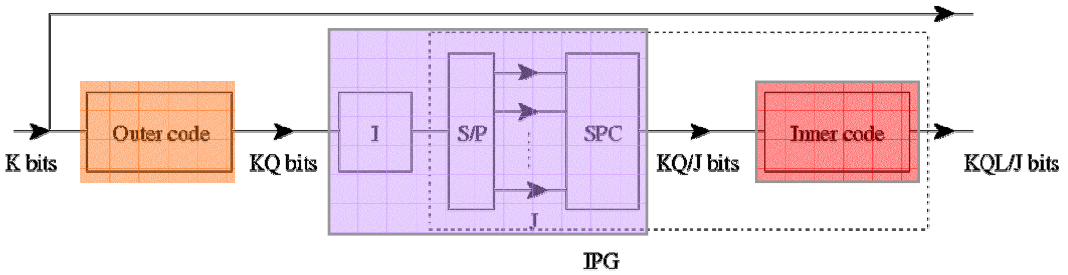
- S-SCP codes description [1] :
  - The structure can be viewed either as:
    - Serial concatenation of three codes (ARA like codes)
    - Punctured irregular LDPC codes



    - We consider a particular class of S-SCP codes
      - Outer code is a circular convolutional code: 1+D
      - Inner code is an accumulator code: 1/1+D

[1] K.M Chugg et al. "A new class of turbo-like codes with universally good performance and high speed decoding ", IEEE Milcom 2005

S-SCP codes



- Parity check matrix is derived from encoding equations:

$$\mathbf{H}\underline{x}^T = \begin{bmatrix} \mathbf{G_1} & \mathbf{I} & 0 \\ 0 & \mathbf{V} & \mathbf{G_2} \end{bmatrix} \begin{bmatrix} \underline{c}^T \\ \underline{h}^T \\ \underline{p}^T \end{bmatrix} = \underline{0}^T$$

  - $\mathbf{G_1}$: dual-diagonal K**x**K matrix when **outer code is 1+D**
  - $\mathbf{G_2}$: dual-diagonal M**x**M matrix when **inner code is 1/1+D**
  - **V** : M**x**K matrix: Interleaver + SPC
    - **J** ones per row
    - **Q** ones per column

S-SCP codes

- Design of Quasi-Cyclic (QC) S-SCP codes [2]:
  - New definition of the parity check matrix **H**
    - Construction based on shifted permutation matrices $\mathbf{I}_x$

      - *$\mathbf{I}_x$ is a **(z x z)** circularly right shifted identity matrix by **(x mod z)** positions, if x is positive*
      - *if x is negative, $\mathbf{I}_x$ is defined to be a **(z x z)** zero matrix*

    - Interleaver and SPC functions are jointly characterized by matrix **V** of size (mz x nz)

$$\mathbf{V} = \begin{bmatrix} \mathbf{I}_{\delta(0,0)} & \mathbf{I}_{\delta(0,1)} & & \mathbf{I}_{\delta(0,n-1)} \\ \mathbf{I}_{\delta(1,0)} & \mathbf{I}_{\delta(1,1)} & & \mathbf{I}_{\delta(1,n-1)} \\ & & \mathbf{I}_{\delta(i,j)} & \\ \mathbf{I}_{\delta(m-1,0)} & \mathbf{I}_{\delta(m-1,1)} & & \mathbf{I}_{\delta(m-1,n-1)} \end{bmatrix}$$

    - We also define all positive coefficients through 3 integers:

$$\delta(i,j) = \left(a^{(i+1)} b^{(j+1)}\right) \bmod \hat{z}$$

[2] Dore et al. "Design and decoding of a serial concatenated code structure based on quasi-cyclic ldpc codes," *4th International Symposium on Turbo-Codes and Related Topics*, April 2006.

S-SCP codes

- Example of V matrix
  - z = 81, J = 2, Q = 2

$$\mathbf{V} = \begin{bmatrix} I_{\delta(0,0)} & -1 & -1 & I_{\delta(0,3)} & -1 & -1 \\ -1 & -1 & I_{\delta(1,2)} & I_{\delta(1,3)} & -1 & -1 \\ I_{\delta(2,0)} & -1 & I_{\delta(2,2)} & -1 & -1 & -1 \\ -1 & I_{\delta(3,1)} & -1 & -1 & I_{\delta(3,4)} & -1 \\ -1 & -1 & -1 & -1 & I_{\delta(4,4)} & I_{\delta(4,5)} \\ -1 & I_{\delta(5,1)} & -1 & -1 & -1 & I_{\delta(5,5)} \end{bmatrix}$$

  - When a = 2 and b = 5

$$\mathbf{V} = \begin{bmatrix} I_{10} & -1 & -1 & I_{35} & -1 & -1 \\ -1 & -1 & I_{14} & I_{76} & -1 & -1 \\ I_{40} & -1 & I_{28} & -1 & -1 & -1 \\ -1 & I_{76} & -1 & -1 & I_{23} & -1 \\ -1 & -1 & -1 & -1 & I_{46} & I_{68} \\ -1 & I_{61} & -1 & -1 & -1 & I_{55} \end{bmatrix}$$

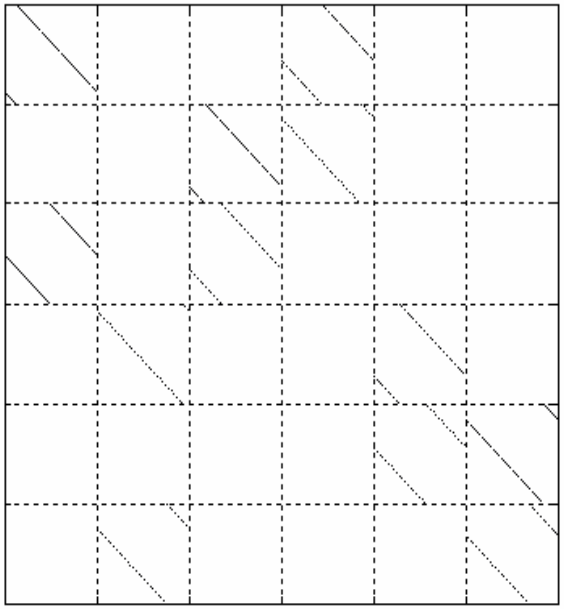


Illustration of Matrix **V**
(Black pixels correspond to position of a '1' in the matrix)

S-SCP codes

- New definition of inner and outer code
  - Outer code definition from circularly shifted identity matrices
    - Circular 1+D convolutional code

$$G_1 = \begin{bmatrix} I & & & I_1 \\ I & I & & \\ & I & & \\ & & I & \\ 0 & & I & I \end{bmatrix}$$

  - Inner code definition
    - Accumulator code (1/1+D)

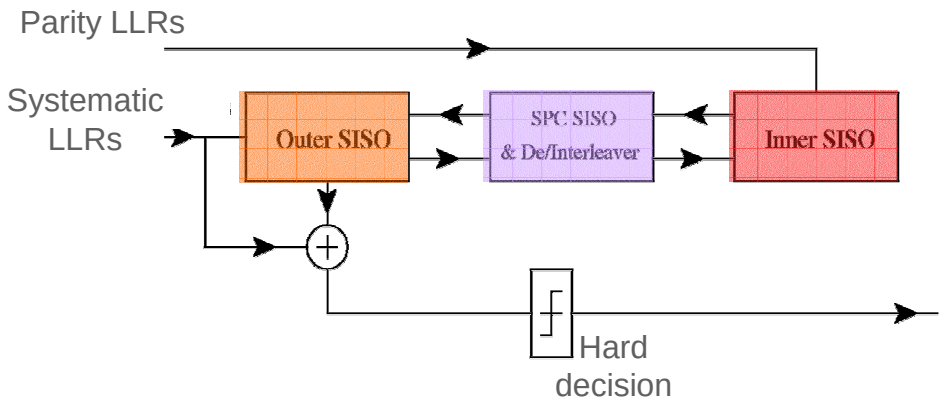
$$G_2 = \begin{bmatrix} I & & & I'_1 \\ I & I & & \\ & I & & \\ & & I & \\ 0 & & I & I \end{bmatrix}$$

    - I'_x is a non circulary shifted identity matrix by x positions

[2] Dore et al. "Design and decoding of a serial concatenated code structure based on quasi-cyclic ldpc codes," *4th International Symposium on Turbo-Codes and Related Topics*, April 2006.

S-SCP codes

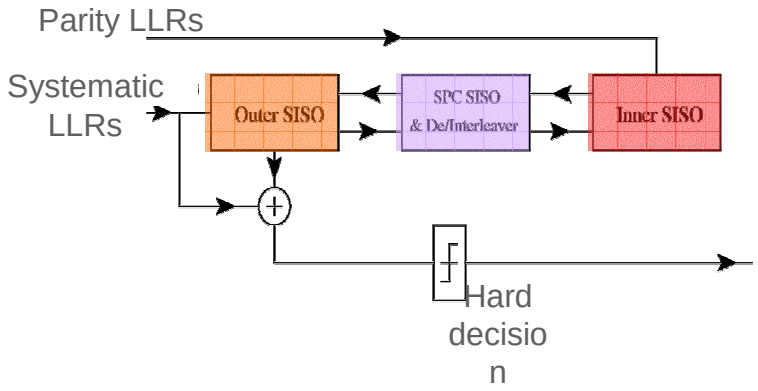
- Decoding strategy
  - Various decoding algorithms [2]:
    - BP familiy algorithms
  - Turbo like decoding algorithm:
    - Outer and Inner SISO decoder: Forward Backward Algorithm (FBA)
    - SPC decoder: can be viewed as a LDPC decoder
      - 2 steps, Inward and Outward

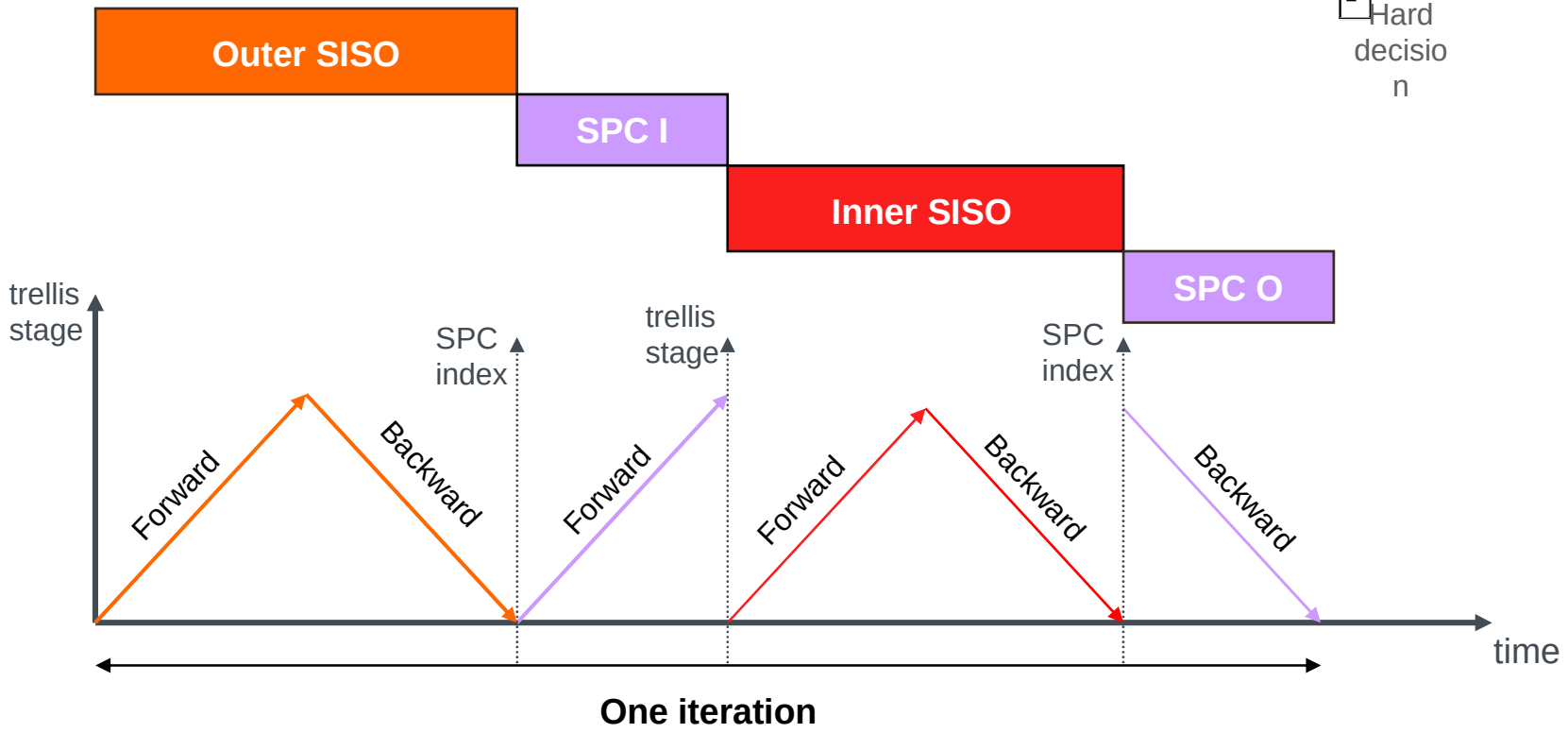


[2] Dore et al. "Design and decoding of a serial concatenated code structure based on quasi-cyclic ldpc codes," *4th International Symposium on Turbo-Codes and Related Topics*, April 2006.

S-SCP codes

- Decoder scheduling
  - Graphical representation



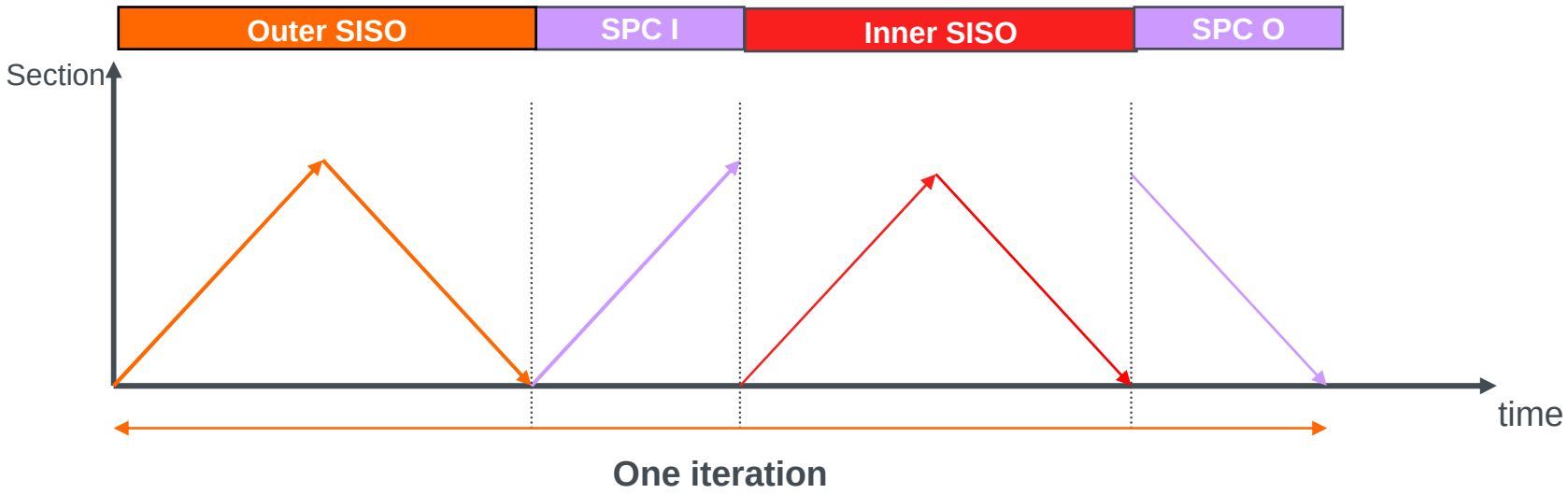


Outline

- Introduction
- S-SCP codes
  - Introduction
  - Decoding Strategy
- Joint decoding strategy and code design
  - Memory contention
  - Towards a pipeline decoding
- Conclusion

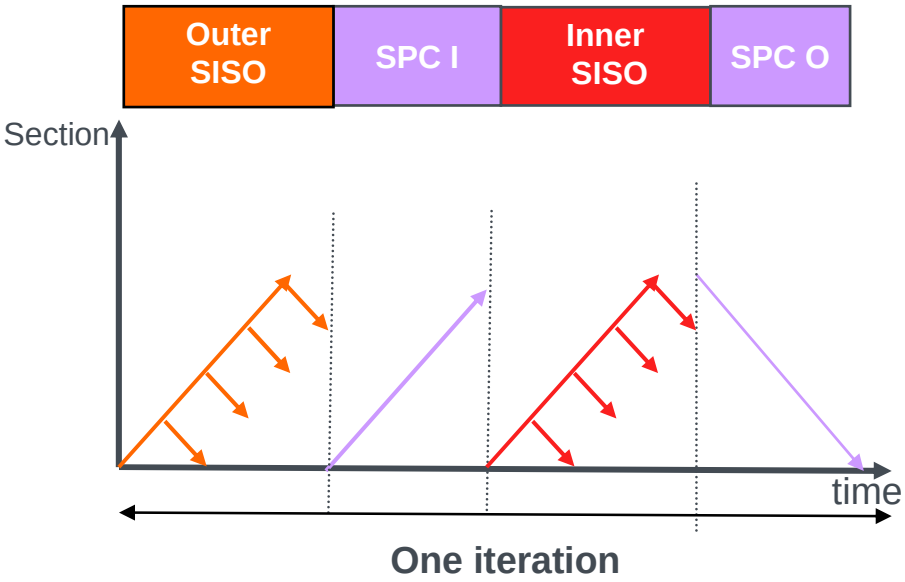
Joint decoding strategy and code design

- "Brute force" architecture
  - Serial Scheduling
    - Low data rate
  - High memory requirements
    - All forward and backward metrics must be stored…
  - No particular design rules on the code



Joint decoding strategy and code design

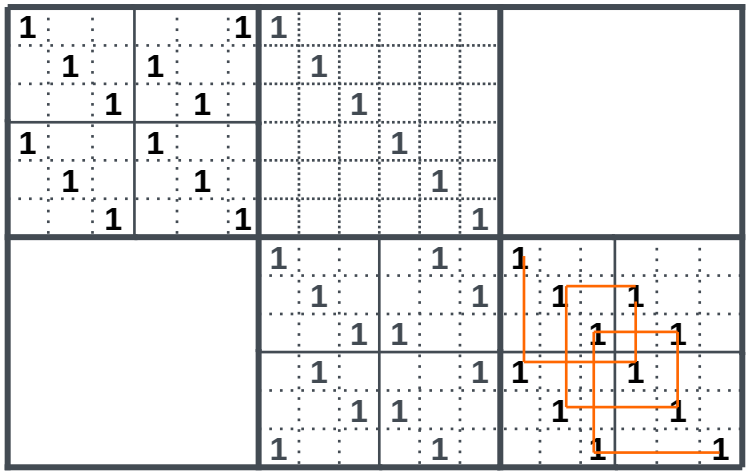
- "Brute force" architecture with sliding windows decoding
  - Serial Scheduling
    - Low data rate
  - Outer and Inner SISO SW decoding
    - Pipeline decoding
    - Latency is **reduced**
    - Memory requirements are **reduced**
  - No particular design rules

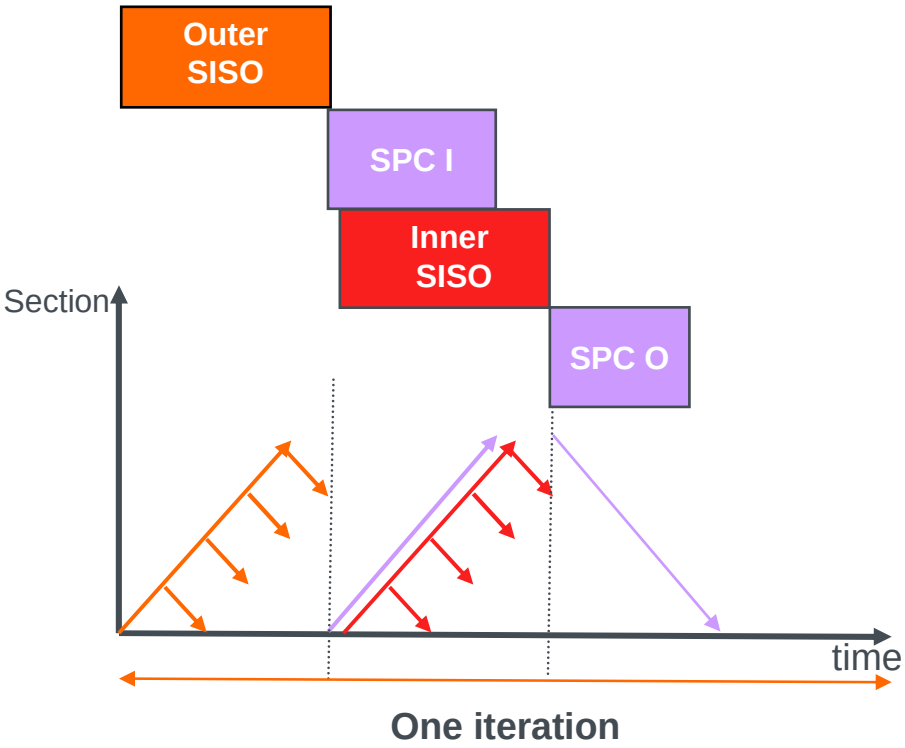


Joint decoding strategy and code design

- Pipeline architecture (I)
  - Serial scheduling
    - Pipeline between SPC Inward and Inner SISO
  - No particular design rules
    - Scheduling of parity equations to check

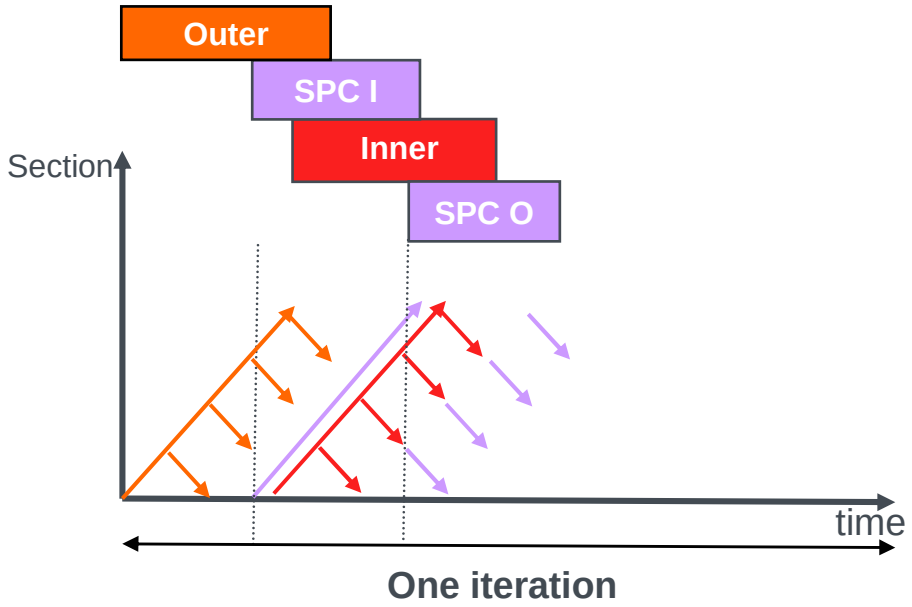
$$\mathbf{H} = \begin{bmatrix} \mathbf{G}_1 & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{V} & \mathbf{G}_2 \end{bmatrix}$$





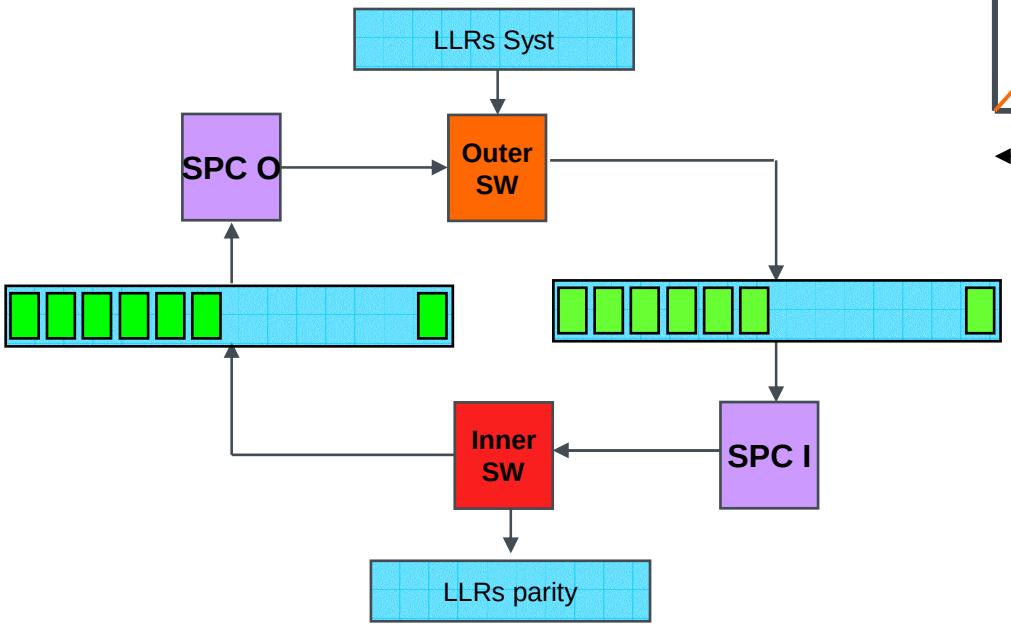
Joint decoding strategy and code design

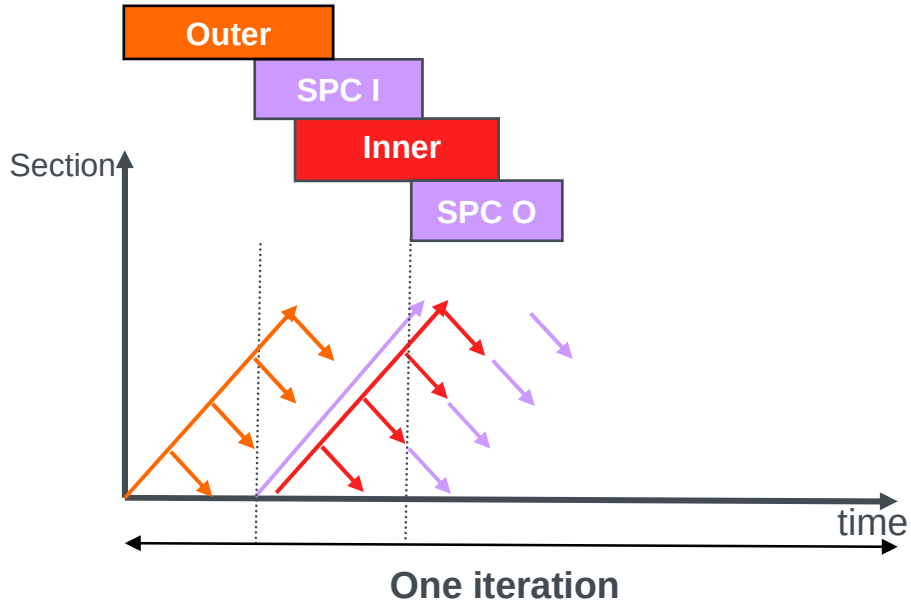
- Pipeline architecture (II)
  - Serial scheduling
    - Pipeline between Outer SISO ,SPC Inward, Inner SISO and SPC Outward
  - Throughput is increased
    - Without hardware duplication
    - Without additional border effects due to parallelization
  - Design rules required for **efficient pipeline decoding**



Joint decoding strategy and code design

- Pipeline decoding
  - Proposed architecture
    - Memory banks organization
  - Each windows decoding information are stored in a memory bank
    - Window of size m (N = (m+n) z)





- Design rules required
  - Avoid memory conflicts
  - under hypothesis:
    - Single port memory

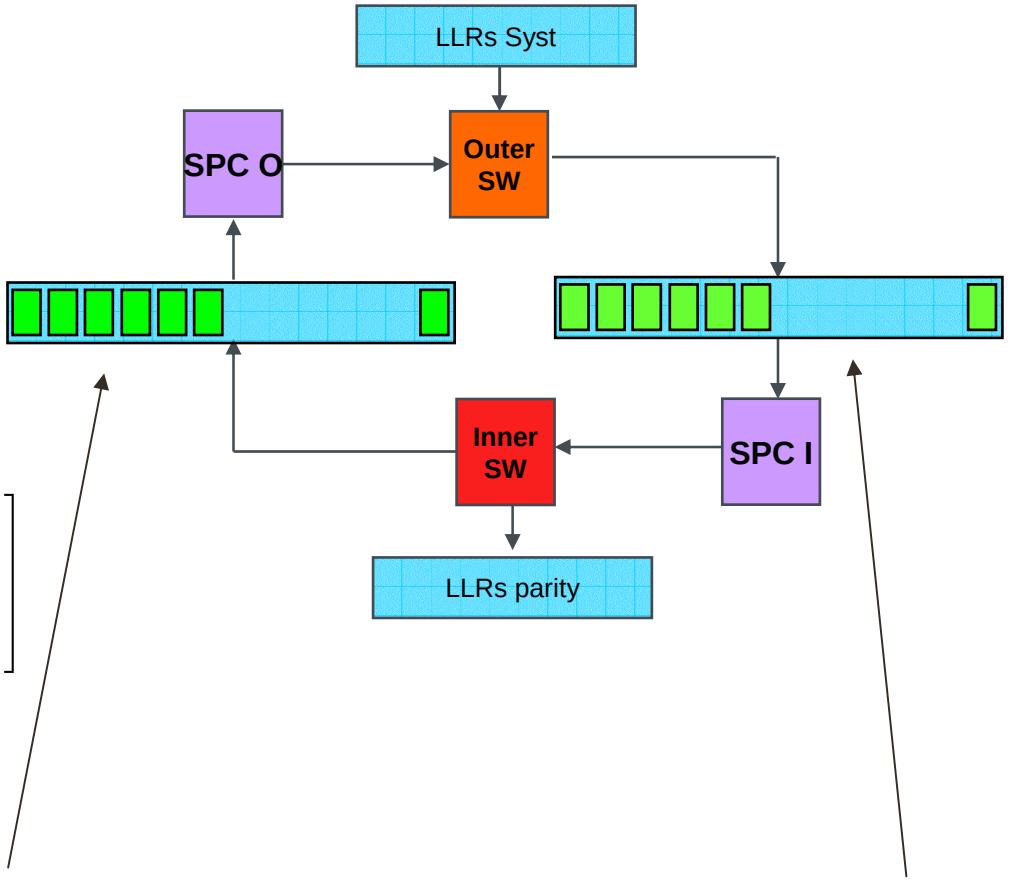
Joint decoding strategy and code design

- Pipeline architecture
  - Avoiding memory conflicts

$$\mathbf{H} = \begin{bmatrix} \mathbf{G}_1 & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{V} & \mathbf{G}_2 \end{bmatrix}$$

  - Rules on matrix **V** design

$$\mathbf{V} = \begin{bmatrix} \mathbf{I}_{\delta(0,0)} & \mathbf{I}_{\delta(0,1)} & & \mathbf{I}_{\delta(0,n-1)} \\ \mathbf{I}_{\delta(1,0)} & \mathbf{I}_{\delta(1,1)} & & \mathbf{I}_{\delta(1,n-1)} \\ & & \mathbf{I}_{\delta(i,j)} & \\ \mathbf{I}_{\delta(m-1,0)} & \mathbf{I}_{\delta(m-1,1)} & & \mathbf{I}_{\delta(m-1,n-1)} \end{bmatrix}$$

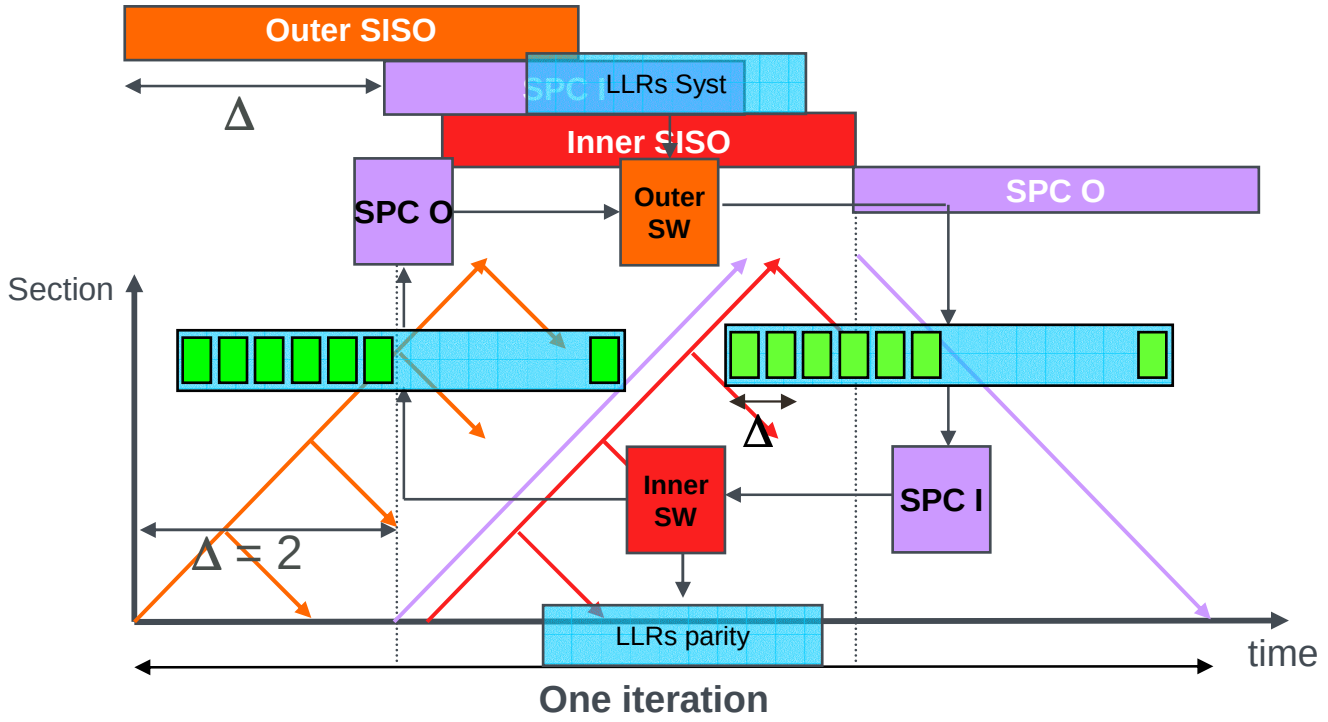


$$\delta(i,j) \neq \delta(k,j), \ \forall j, \ \forall i \neq k$$

$$\delta(i,j) \neq \delta(i,k), \ \forall i, \ \forall j \neq k$$

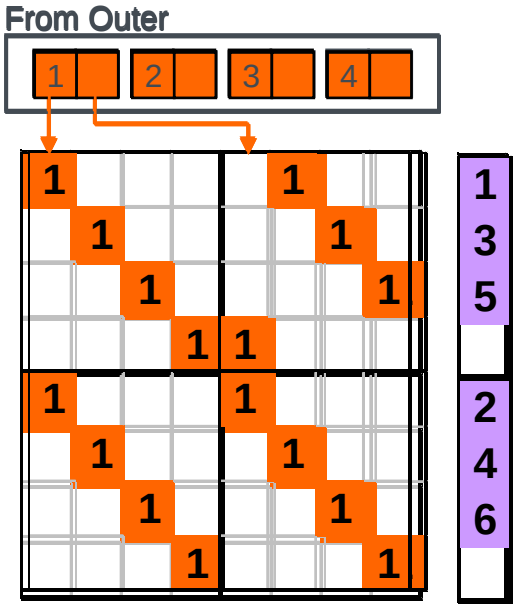
Joint decoding strategy and code design

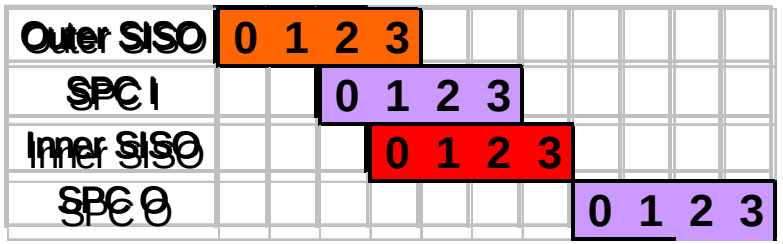
- Pipeline decoding
  - Idea: Find design rules on matrix **V** to characterize pipeline decoding
  - Introduction of the overlapping parameter Δ
    - A SPC Inward window starts when Δ Outer SISO windows have been decoded



Joint decoding strategy and code design

- Rules on matrix **V** to characterize pipeline decoding
  - First Hypothesis:
    - Outer code is not a circular code





$$\mathbf{H} = \begin{bmatrix} G_1 & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & V & G_2 \end{bmatrix} \qquad \mathbf{V} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

- Relation between **V** coefficients and Δ

$$\Delta = 1 + \max\left(\delta(i,j)\right), \ \forall \delta(i,j) \geq 0$$

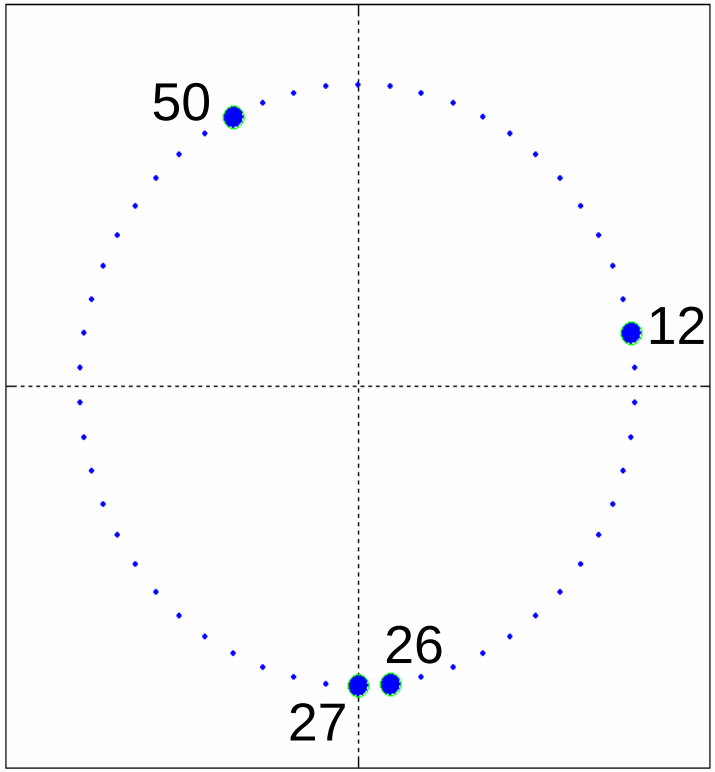
- Remark:

$$\Delta \leq 1 + \hat{z}$$

$$\delta(i,j) = \left(a^{(i+1)} b^{(j+1)}\right) \bmod \hat{z}$$

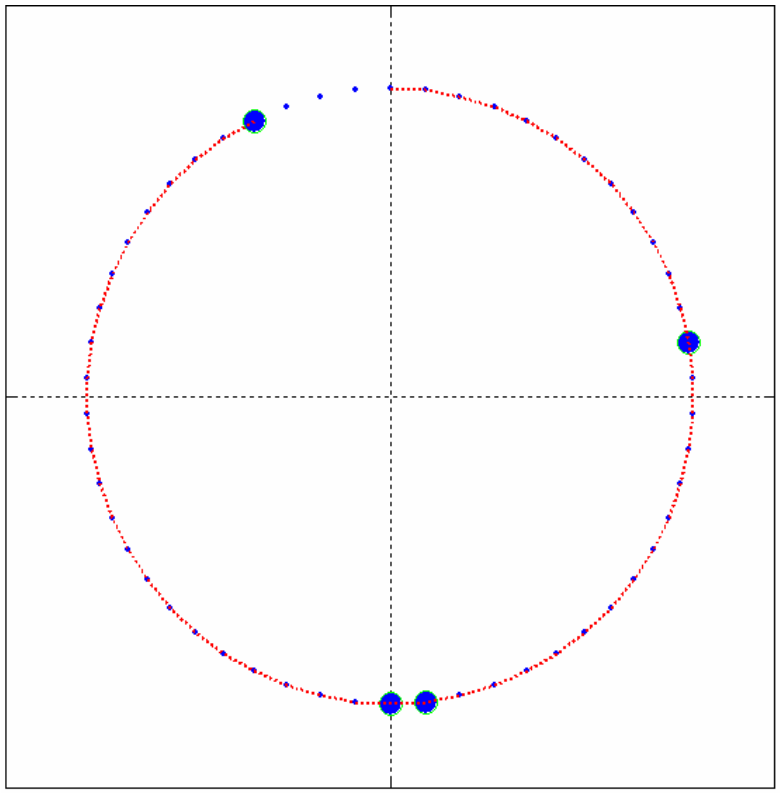
Joint decoding strategy and code design

- Rules on matrix **V** to characterize pipeline decoding
  - Hypothesis:
    - Outer code is not a circular code
  - Graphical representation
    - z = 54



$$\mathbf{V} = \begin{bmatrix} 26 & 12 \\ 50 & 27 \end{bmatrix}$$

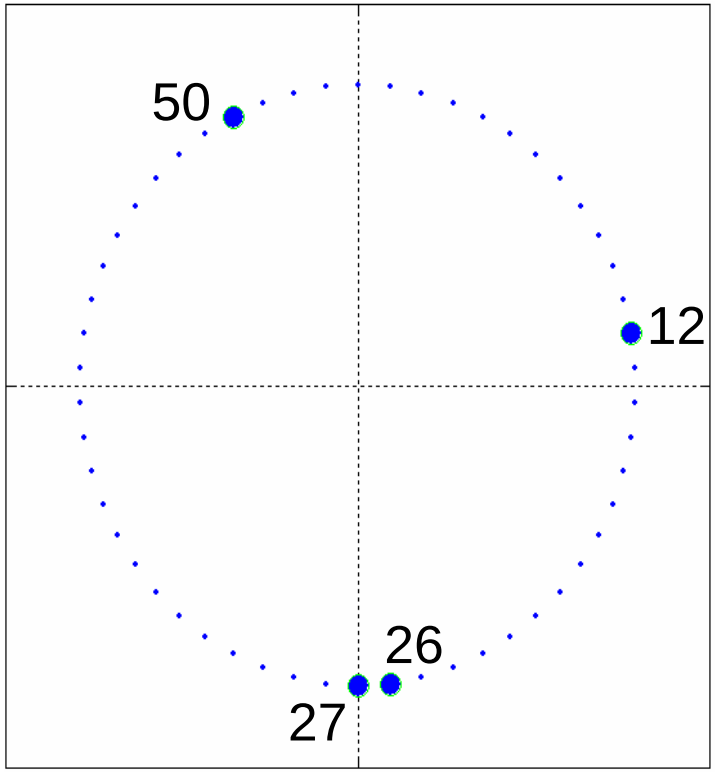
$$\Delta = 51$$



Joint decoding strategy and code design

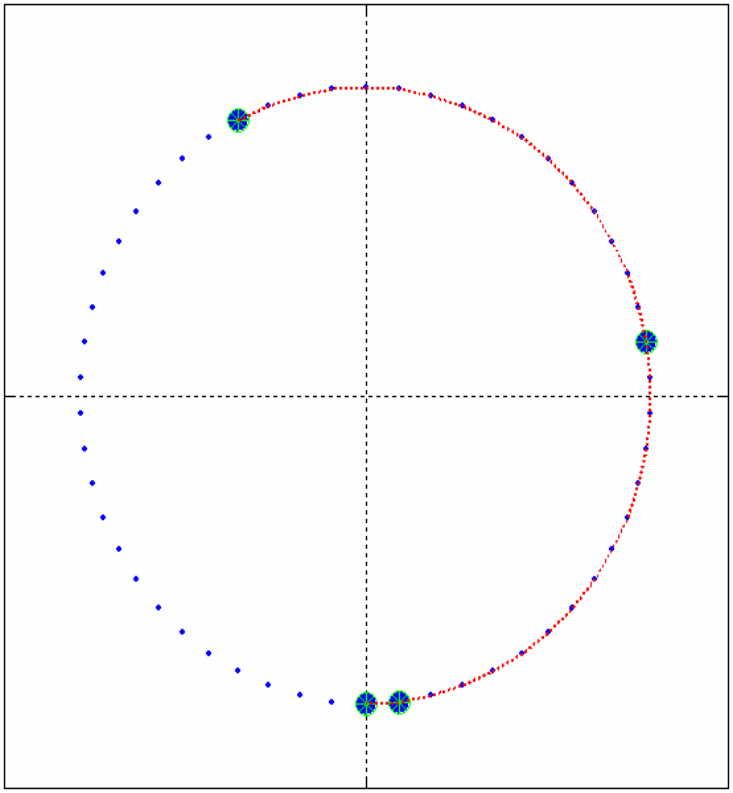
- Rules on matrix **V** to characterize pipeline decoding
  - Hypothesis:
    - Outer code is **a circular code**
  - Graphical representation
    - z = 54

$\Delta \neq 51$



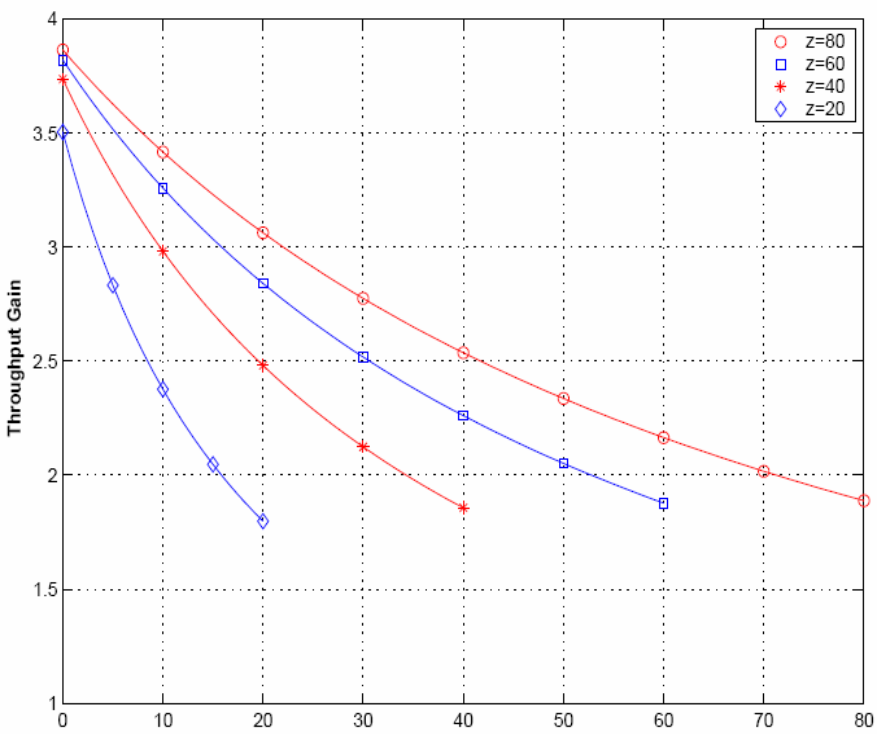
$$\mathbf{V} = \begin{bmatrix} 26 & 12 \\ 50 & 27 \end{bmatrix}$$

$\Delta = 32$



Joint decoding strategy and code design

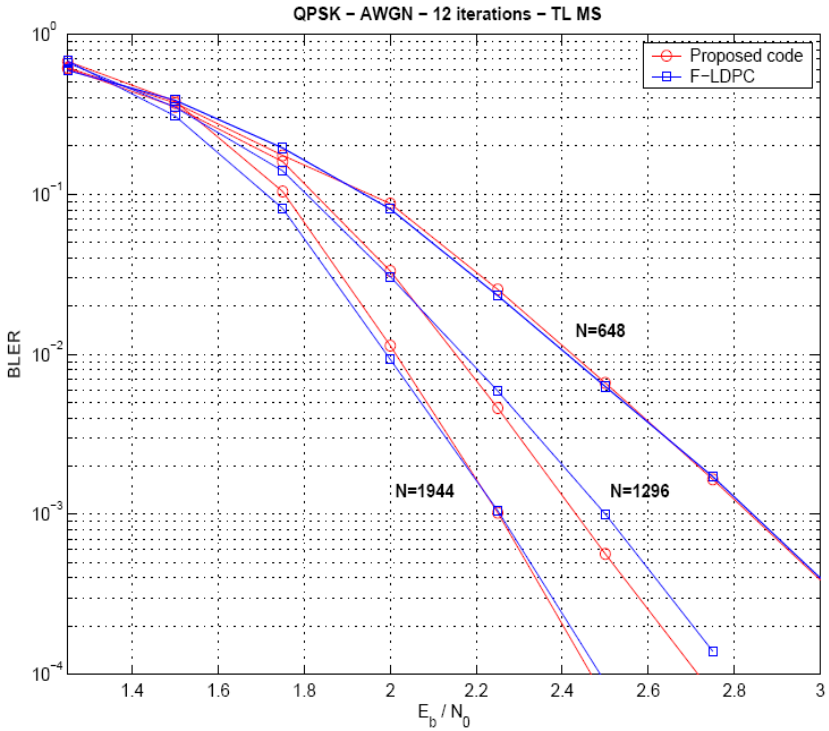
- Pipeline decoding
  - Throughput analysis
    - 12 iterations
    - J = 2, n = 12



Δ: overlapping parameter

- Design example

N	648	1296	1944
R	$1/2$	$1/2$	$1/2$
J	2	2	2
z	27	54	81
$\hat{z}$	27	53	49
Δ	23	48	49



Conclusion

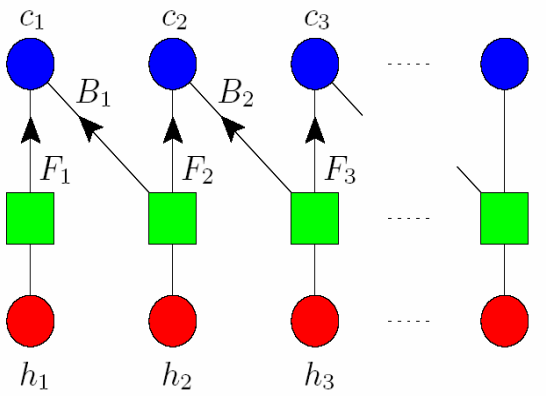
- "Architecture driven" approach
  - Parity check matrix of the code is based on shifted identity matrices
  - Simple constraints on code design (law on coefficients choice)
    - Avoid memory conflicts
    - Overlapping decoding for a particular architecture
  - Throughput analysis :
    - Improve throughput by two without hardware duplication in the example considered
- Methodology and design rules can be extended
  - For an architecture with only one memory bank
  - To enable decoder parallelization
    - Avoid memory conflicts
  - To guarantee a good convergence of the decoding algorithm
    - Rules on shift coefficients are derived for layered decoding

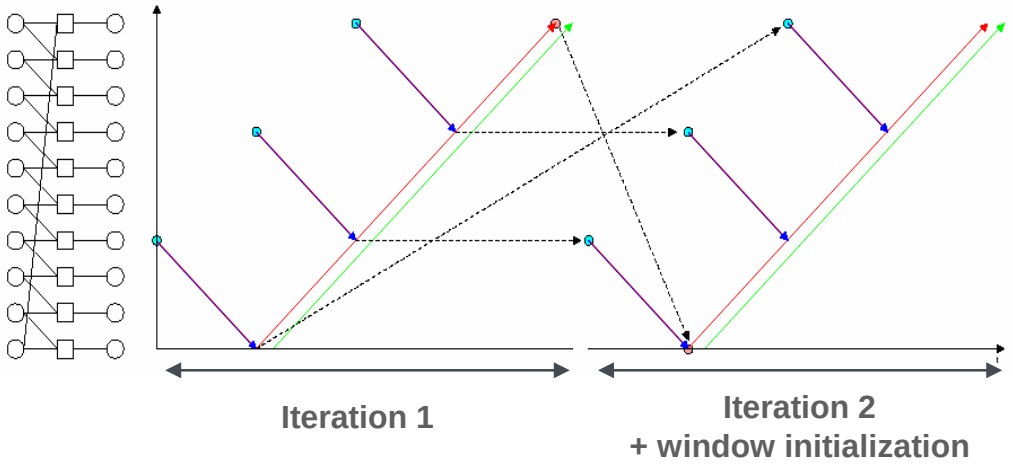
Iterative decoding of QC S-SCP

- **Outer and inner SISO decoders**
  - Forward Backward Algorithm on the graph
    - Simply described using parity check function g(.,.)

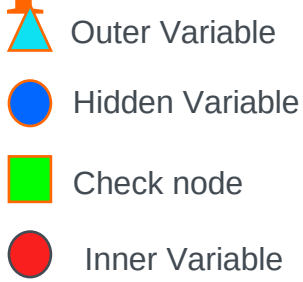
$$
\begin{aligned}
F_i &= g\left(c_{i-1} + F_{i-1}, h_i\right) \\
B_i &= g\left(c_{i+1} + B_{i+1}, h_{i+1}\right) \\
E_{c_i} &= B_i + F_i \\
E_{h_i} &= g\left(c_{i-1} + F_{i-1}, c_i + B_i\right)
\end{aligned}
$$

    - Memor liding windows

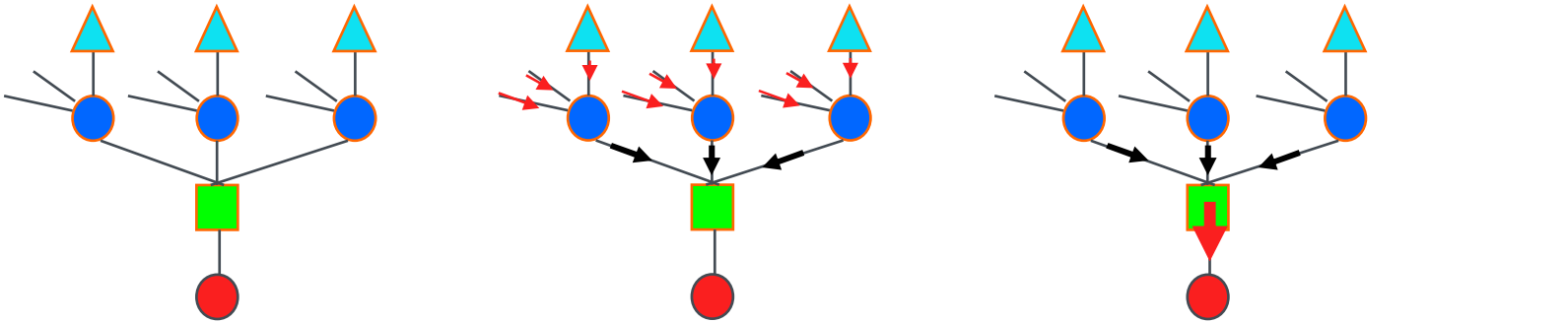




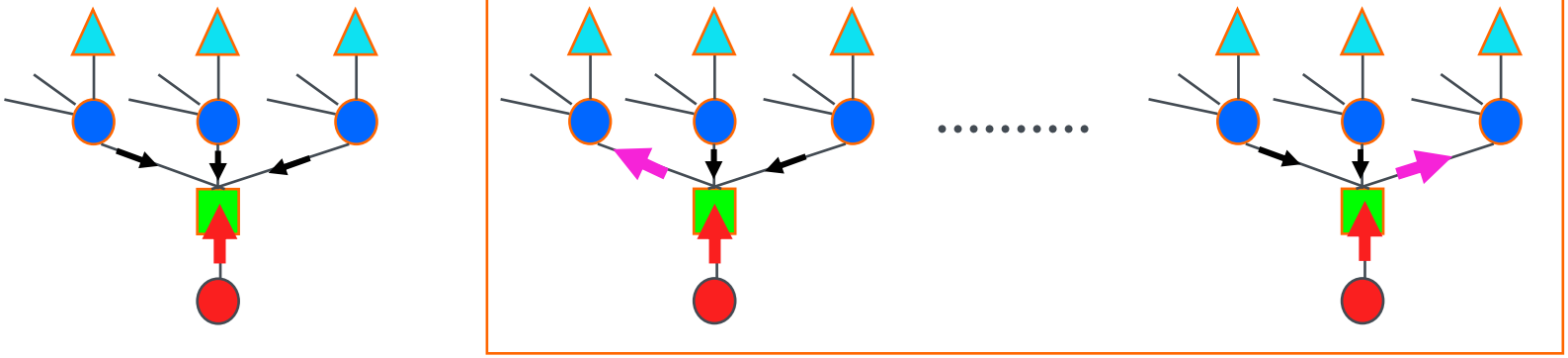
Iterative decoding of QC S-SCP



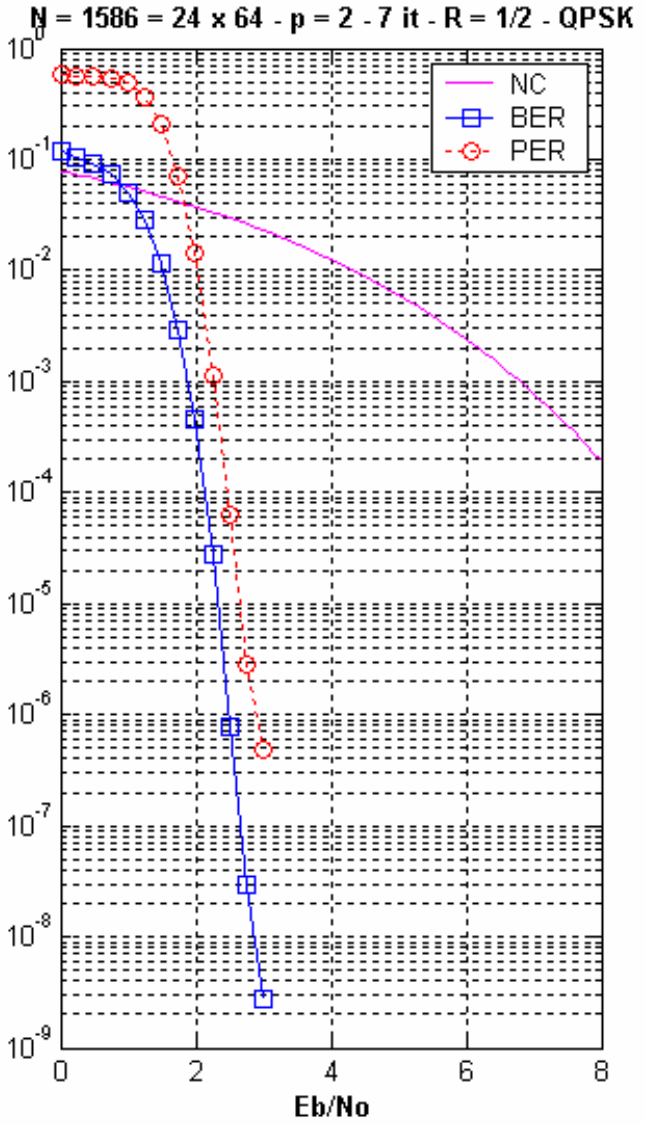
- **SPC decoders**
  - Inward process

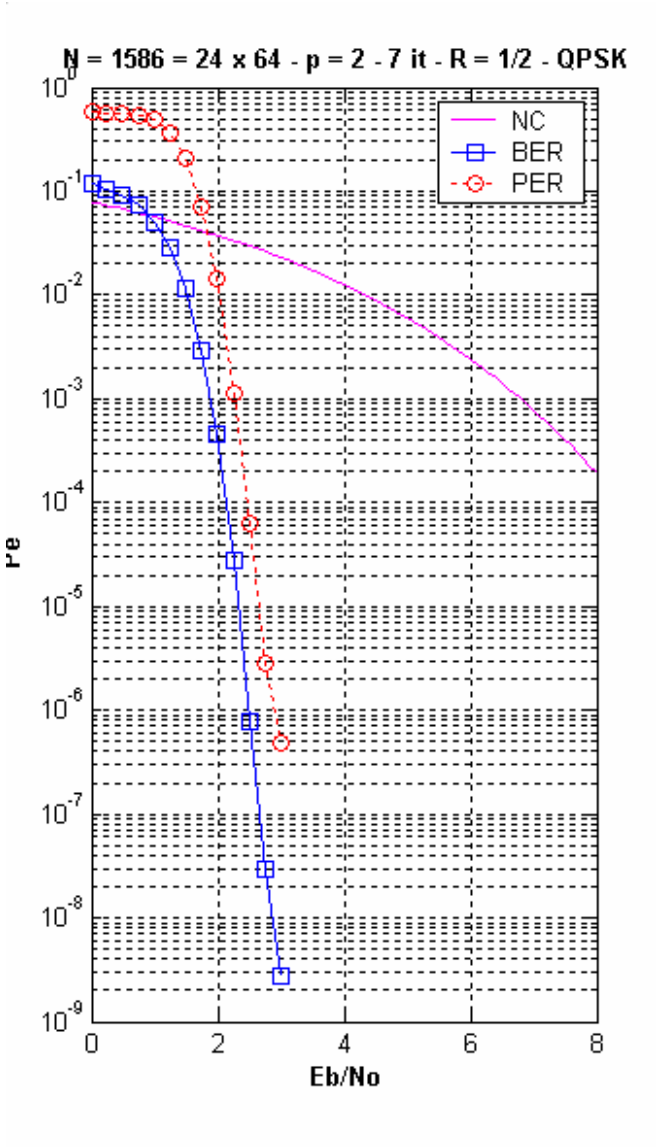


  - Outward process



Hardware plateform




N = 1586 = 24 x 64 - p = 2 - 7 it - R = 1/2 - QPSK
NC
BER
PER
Pe
Eb/No
10^0
10^-1
10^-2
10^-3
10^-4
10^-5
10^-6
10^-7
10^-8
10^-9
0
2
4
6
8