Back-to-Back Butterfly Network, an Adaptive Permutation Network for New Communication Standards

Hassan Harb and Cyrille Chavet
UBS, Lab-STICC, Lorient France

Objective

- Efficient method to find out a solution for a given permutation using Back-to-Back Butterfly Network

Applications

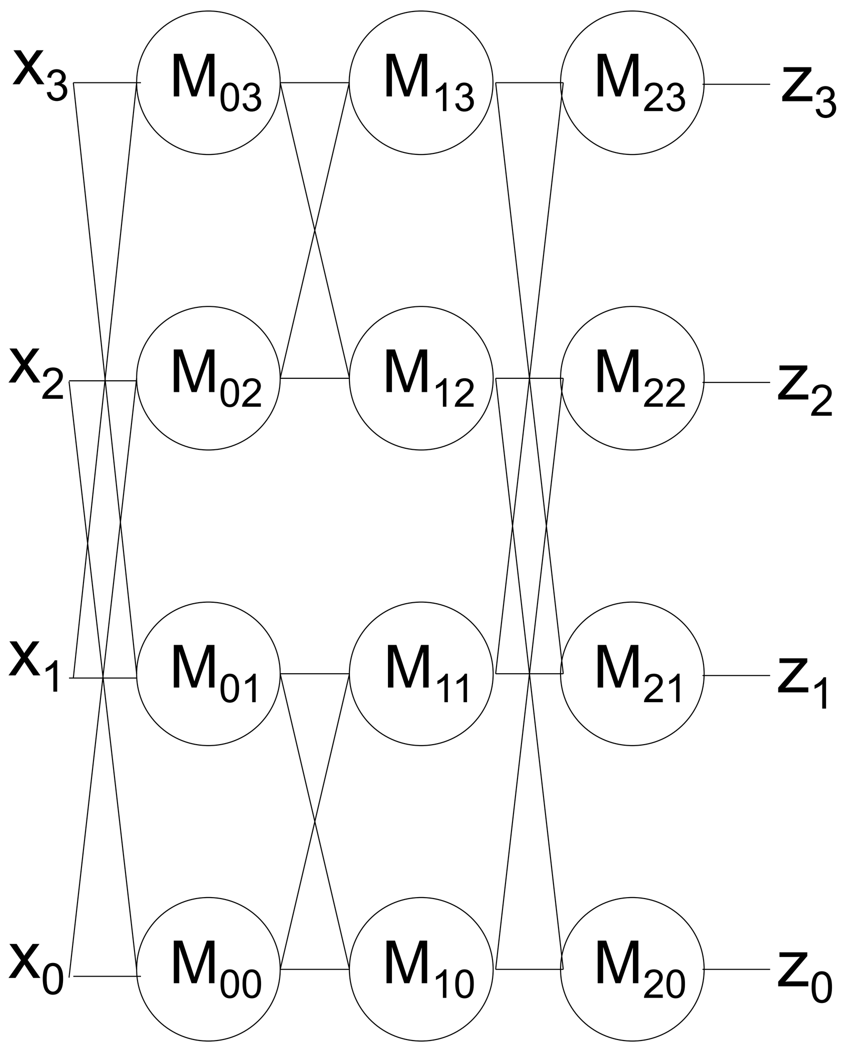
- LDPC and NB-LDPC codes
- 5G LDPC codes

Methodology

- Algorithmic Optimization
- Solver tools modeling

Back-to-Back Butterfly Network

Example with ***N=4*** inputs *(2*log₂(N)-1 stages)*:

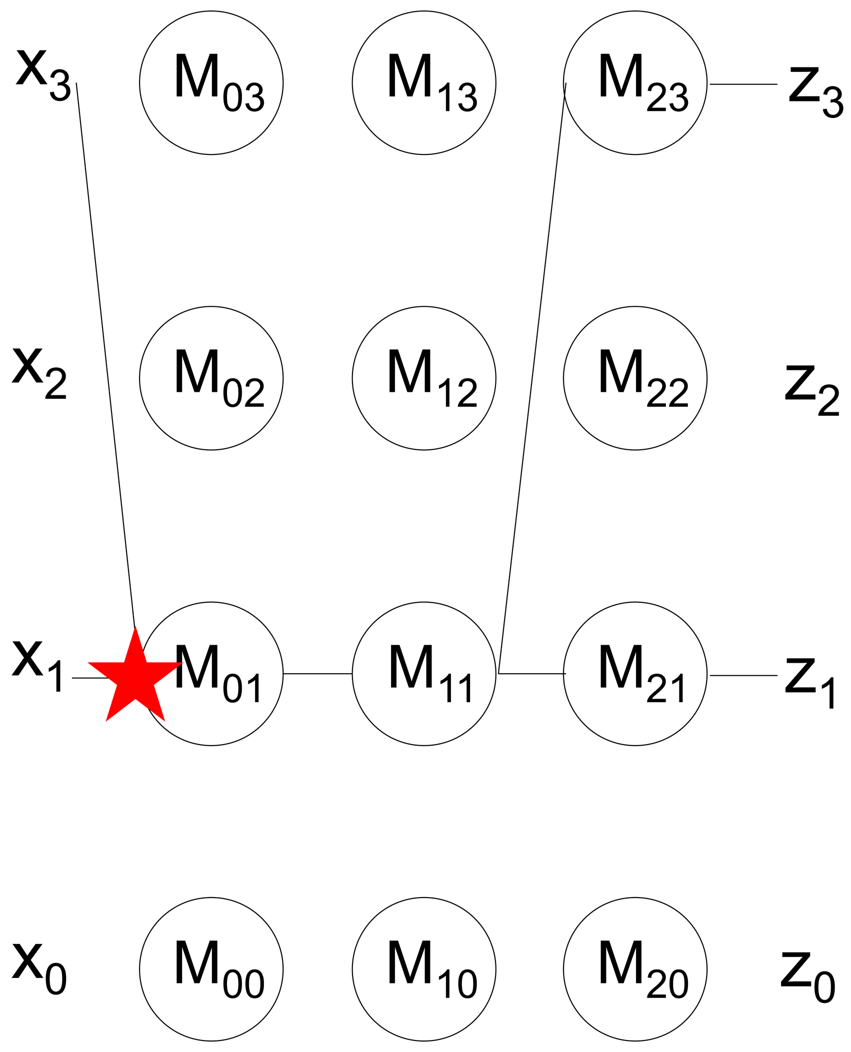


M_{ij} *- Multiplexer*

All the ***4!*** *= 24* possible permutations can be performed using this ***4x4*** *Back-to-Back* Butterfly Network

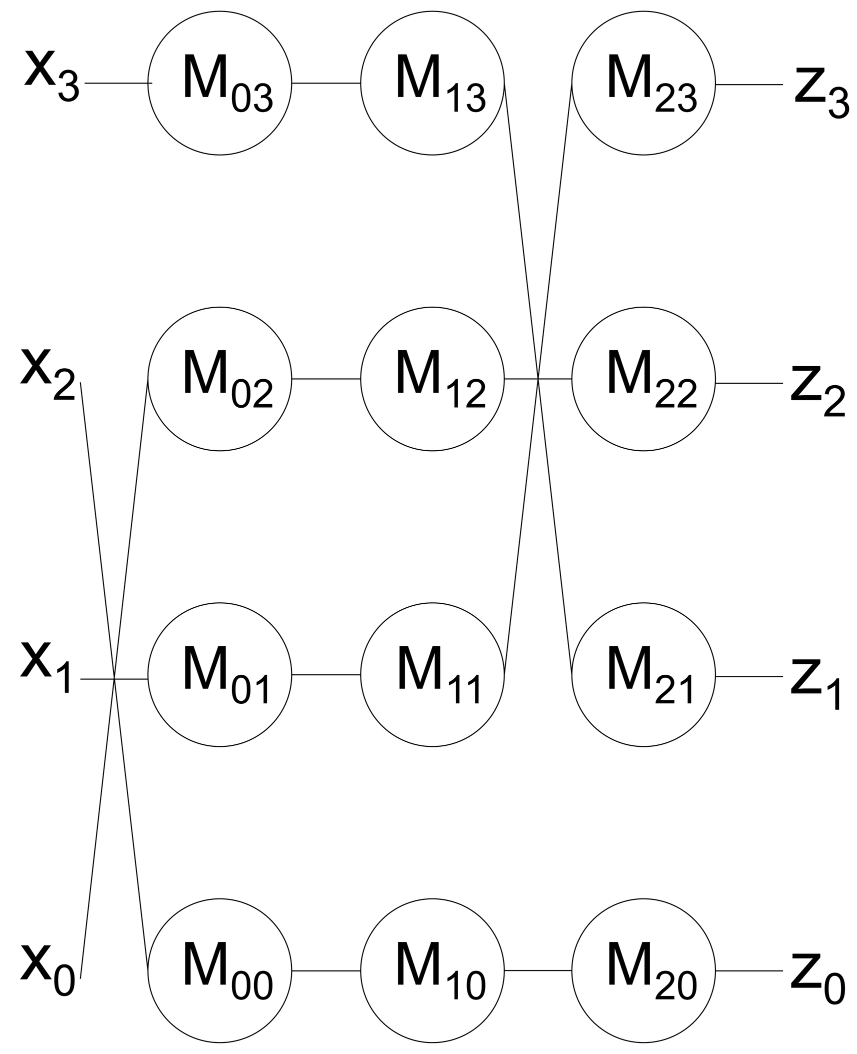
Permutation Example

Conflict possibility:



$x_3 \rightarrow z_1$, $x_2 \rightarrow z_0$, $x_1 \rightarrow z_3$, $x_0 \rightarrow z_2$

Conflict-free:



$x_3 \rightarrow z_1$, $x_2 \rightarrow z_0$, $x_1 \rightarrow z_3$, $x_0 \rightarrow z_2$

Collect all the possible paths

$x_3 \rightarrow z_1$	3 → 3 → 1
	1 → 1 → 1
$x_2 \rightarrow z_0$	2 → 2 → 0
	0 → 0 → 0
$x_1 \rightarrow z_3$	1 → 1 → 3
	3 → 3 → 3
$x_0 \rightarrow z_2$	0 → 0 → 2
	2 → 2 → 2

Collect all possible paths that transfer
- x_3 to z_1
- x_2 to z_0
- x_1 to z_3
- x_0 to z_2

Every M_{ij} (*i=0,1,2* and *j=0,1,2,3*) is represented by its index *j*.

The number of possible paths for each case is equal to the number of inputs *N=4*.
From every set of paths, one non-conflicting path should be selected.

Mitigation of the number of collected paths

$x_3 \rightarrow z_1$	3 → 3 → 1
	1 → 1 → 1
$x_2 \rightarrow z_0$	2 → 2 → 0
	0 → 0 → 0
$x_1 \rightarrow z_3$	1 → 1 → 3
	3 → 3 → 3
$x_0 \rightarrow z_2$	0 → 0 → 2
	2 → 2 → 2

1- The idea is to fix one path among ***N=4*** paths associated to $x_3 \rightarrow z_1$ (in this example ***3→3→1***).
2- The paths with common multiplexer(s) with the fixed one are removed (to avoid conflicts). For example, for $x_1 \rightarrow z_3$, the path ***3→3→3*** is removed since the second multiplexer is in common with the fixed one.
3- The remaining set of paths are given to the solver *Gecode* [1] in order to select conflict-free paths.

Solution with Gecode

1. Define a matrix of size 3x3

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix}$$

2. Define domains for each row

Every row is associated to one transition:
- first row $\{a_{00}, a_{01}, a_{02}\}$ is associated to transition $x_2 \rightarrow z_0$
- second row $\{a_{10}, a_{11}, a_{12}\}$ is associated to transition $x_1 \rightarrow z_3$
- third row $\{a_{20}, a_{21}, a_{22}\}$ is associated to transition $x_0 \rightarrow z_2$

Domains: $\{a_{00}, a_{01}, a_{02}\} \in \{\{2,2,0\}, \{0,0,0\}, \{0,1,0\}\}$, $\{a_{10}, a_{11}, a_{12}\} \in \{\{1,1,3\}, \{1,0,3\}\}$ and $\{a_{20}, a_{21}, a_{22}\} \in \{\{0,0,2\}, \{0,1,2\}, \{2,2,2\}\}$

3. Define constraints on each column

In order to prevent any conflict in terms of multiplexer, the elements of each column should be disjointly different

More precisely, $a_{00} \neq a_{10} \neq a_{20}$, $a_{01} \neq a_{11} \neq a_{21}$ and $a_{02} \neq a_{12} \neq a_{22}$

4. Launch the constraint solver tool

After defining the matrix, the domains and the constraints. The constraint solver *Gecode* is ready to be launched.
One possible solution is:

$\{a_{00}, a_{01}, a_{02}\} = \{0, 0, 0\}$
$\{a_{10}, a_{11}, a_{12}\} = \{1, 1, 3\}$
$\{a_{20}, a_{21}, a_{22}\} = \{2, 2, 2\}$

Complexity Analysis

N	Algorithm	Number of MUXs	Parallelism
80	Our model	1664	Unlimited
	[2]	945	Not possible
	[3]	640	Limited
384	Our model	8704	Unlimited
	[2]	6273	Not possible
	[3]	3840	Limited

Even though the existing circular-shift rotation networks [2] and [3] require less number of Multiplexers (MUXs) when compared to our model, **they cannot handle more than one set of elements with different lengths and different circular-shift rotation values.**
Our model is able to handle such case which is a key point for high throughput rate architecture.

[1] https://www.gecode.org
[2] S. L. X. Chen and V. Akella, "Qsn : A simple circular-shift network forreconfigurable quasi-cyclic ldpc decoders", IEEE Trans. on Circuits andSystems II: Express Briefs, vol. 57, no. 10, pp. 782–786, Oct 2010.
[3] E. Boutillon and H. Harb, "Extended barrel-shifter for versatile qc-ldpcdecoders", IEEE Wireless Communications Letters, pp. 1–1, 2020.

IEEE ICASSP Conference, Barcelona, Spain, May 4-8 2020