Quelques aspects de la réalisation matérielle d'un décodeur *LDPC*

<u>François VERDIER</u>
David DECLERCQ
Jean-Marc Philippe

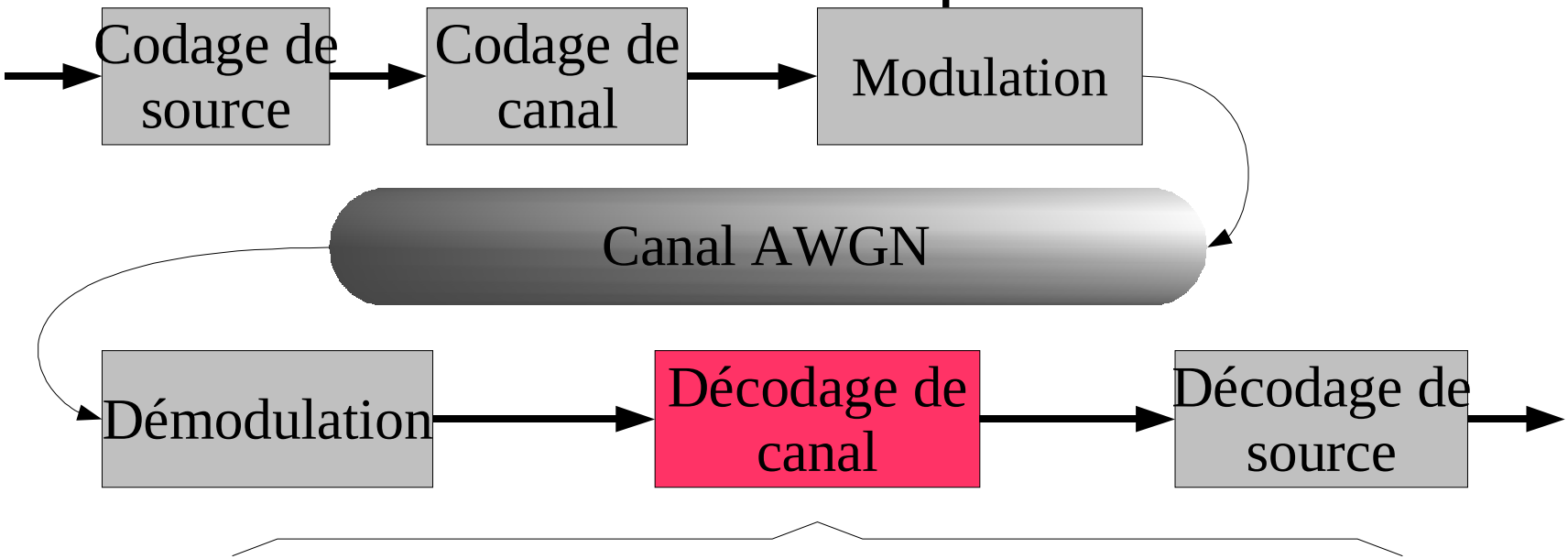
« Equipes Traitement des Images et du Signal »
ETIS
UMR CNRS 8051

http://www.etis.ensea.fr

Quelques aspects de la réalisation matérielle d'un décodeur *LDPC*

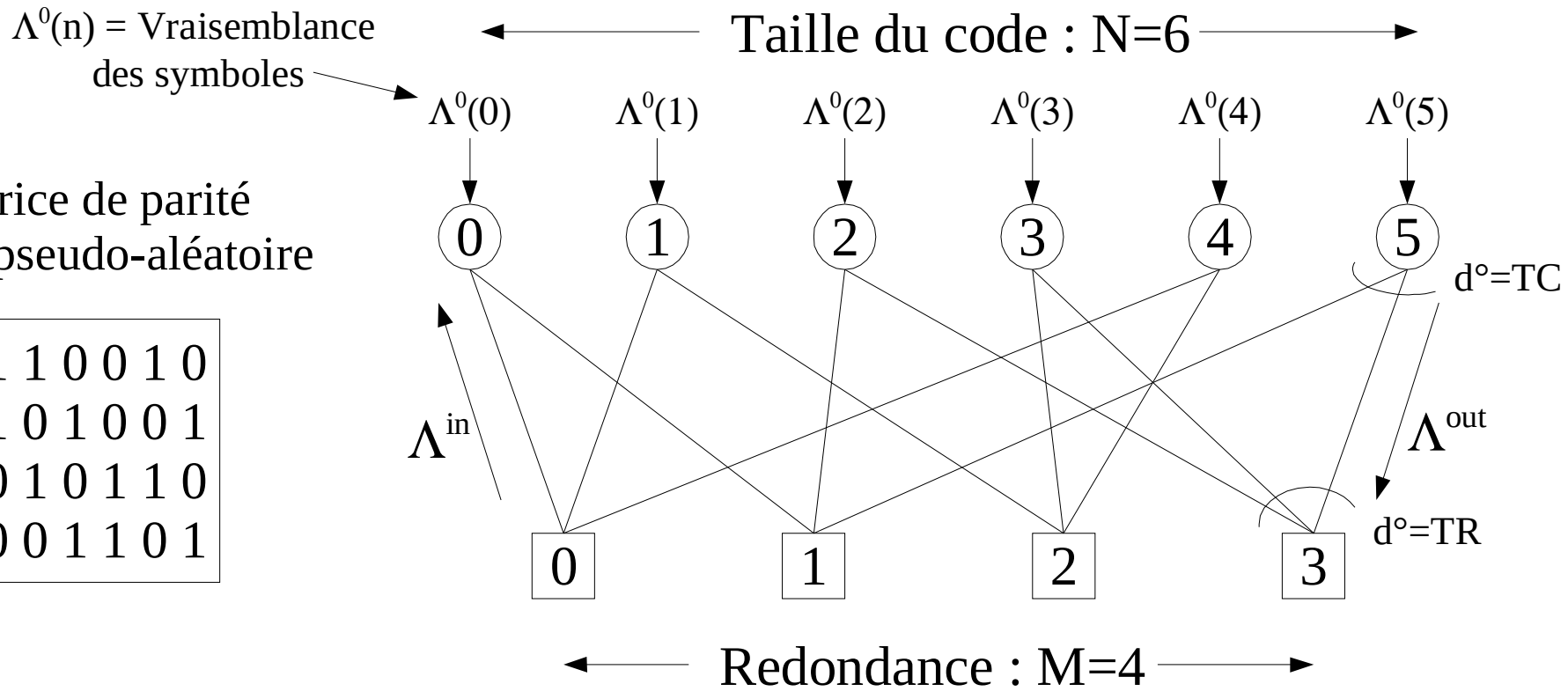
1. Les codes *LDPC*
2. La classe des codes congruents réguliers
3. Architecture et parallélisation du décodeur
4. Conséquences du calcul en précision finie

1. Les codes *LDPC* dans les transmissions radio-numériques



- Codes « blocs » à vérification de parité de grande taille (10^4 à 10^6)
- Canaux fortement bruités (RSB = qq dB)
- Débits importants (> Msymb./sec.)
- Rendements variables

Représentation graphique d'un code *LDPC*



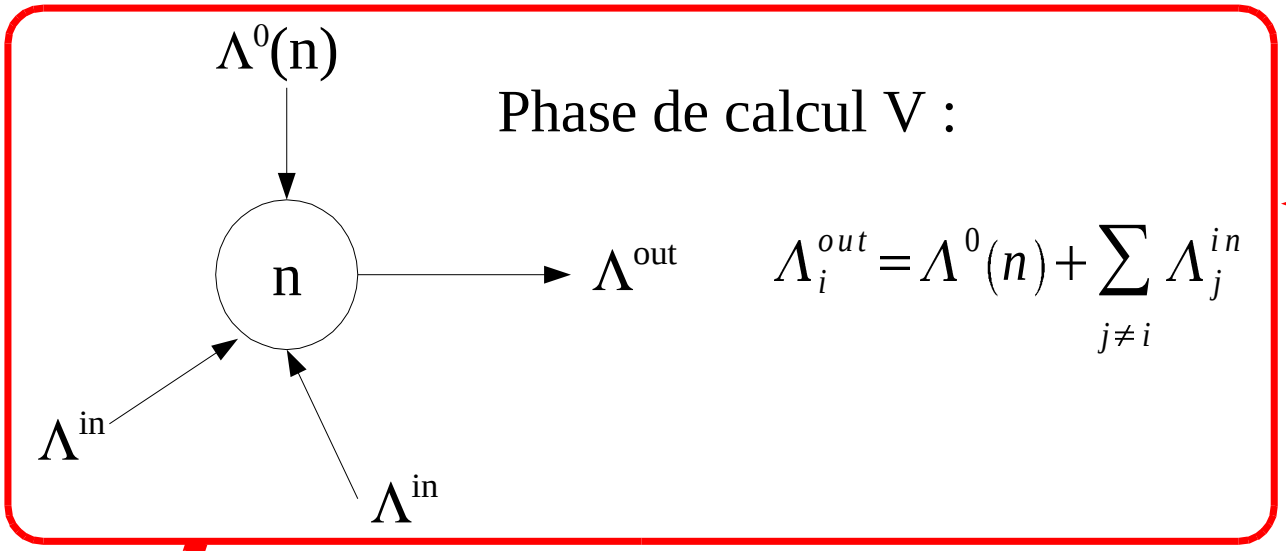
Matrice de parité creuse pseudo-aléatoire

$$H = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Λ^{in} et Λ^{out} : messages échangés sur les branches

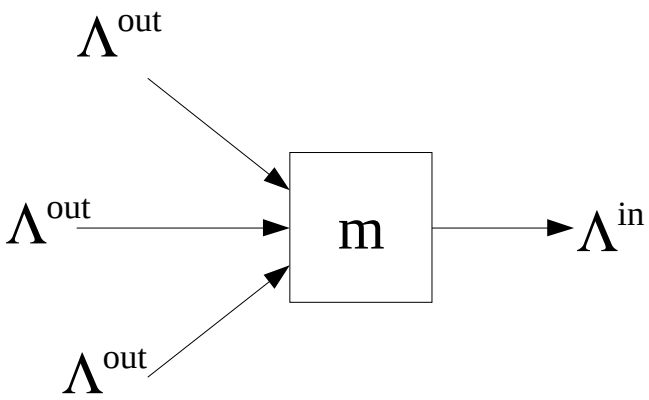
Code LDPC régulier (6,4,2,3) de rendement ½

Algorithme de décodage (Log-BP)



Phase de calcul V :

$$\Lambda_i^{out} = \Lambda^0(n) + \sum_{j \neq i} \Lambda_j^{in}$$



Phase de calcul C :

$$\left|\Lambda_j^{in}\right| = \mathrm{g}\left(\sum_{j \neq i} \mathrm{f}\left(\left|\Lambda_i^{out}\right|\right)\right)$$

$$\mathrm{f}(x) = \log\left(\tanh\frac{x}{2}\right)$$

$$\mathrm{g}(x) = 2 \times atanh\left(\mathrm{e}^x\right)$$

$$sign\left(\Lambda_j^{in}\right) = \prod_{j \neq i} sign\left(\Lambda_i^{out}\right)$$

Problème d'Adéquation Algorithme / Architecture

Matrice de décodage pseudo-aléatoire → **Codage ? Stockage ?**

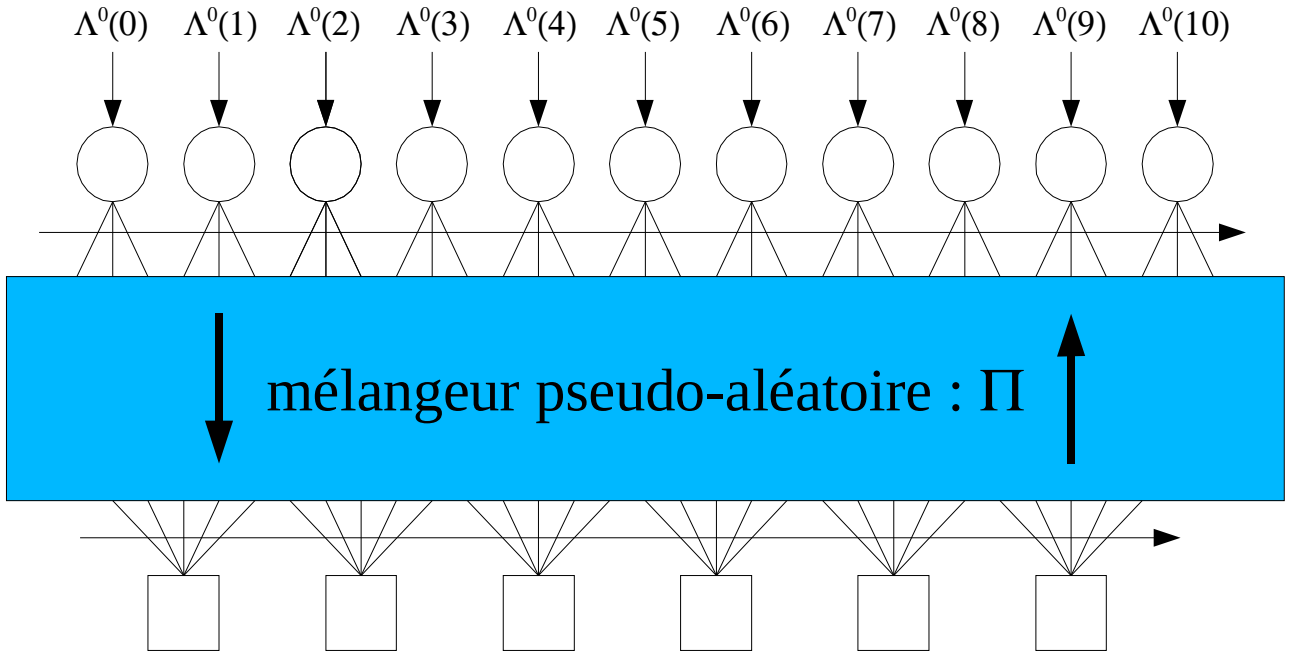
Codes de grande taille (10^4 à 10^6) ↔ **Grande quantité de mémoire**

Taille de code variable → **Parallélisation / scalabilité ?**

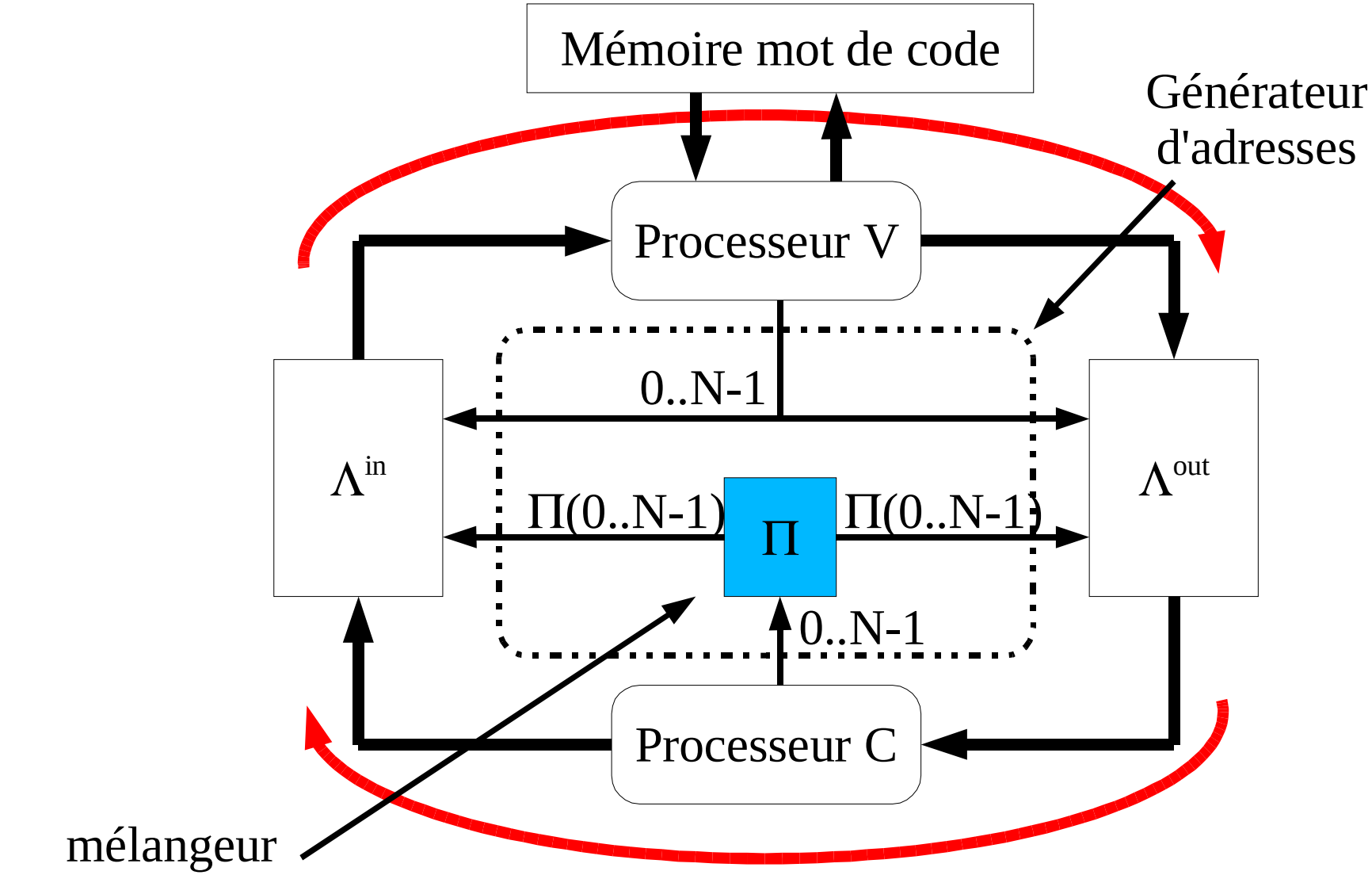
Impact architectural du code pseudo-aléatoire

Les messages Λ^{in} et Λ^{out} sont stockés en mémoire
Les branches sont parcourues séquentiellement dans l'ordre naturel

Parcours des branches / parcours des adresses ?



Solution proposée :

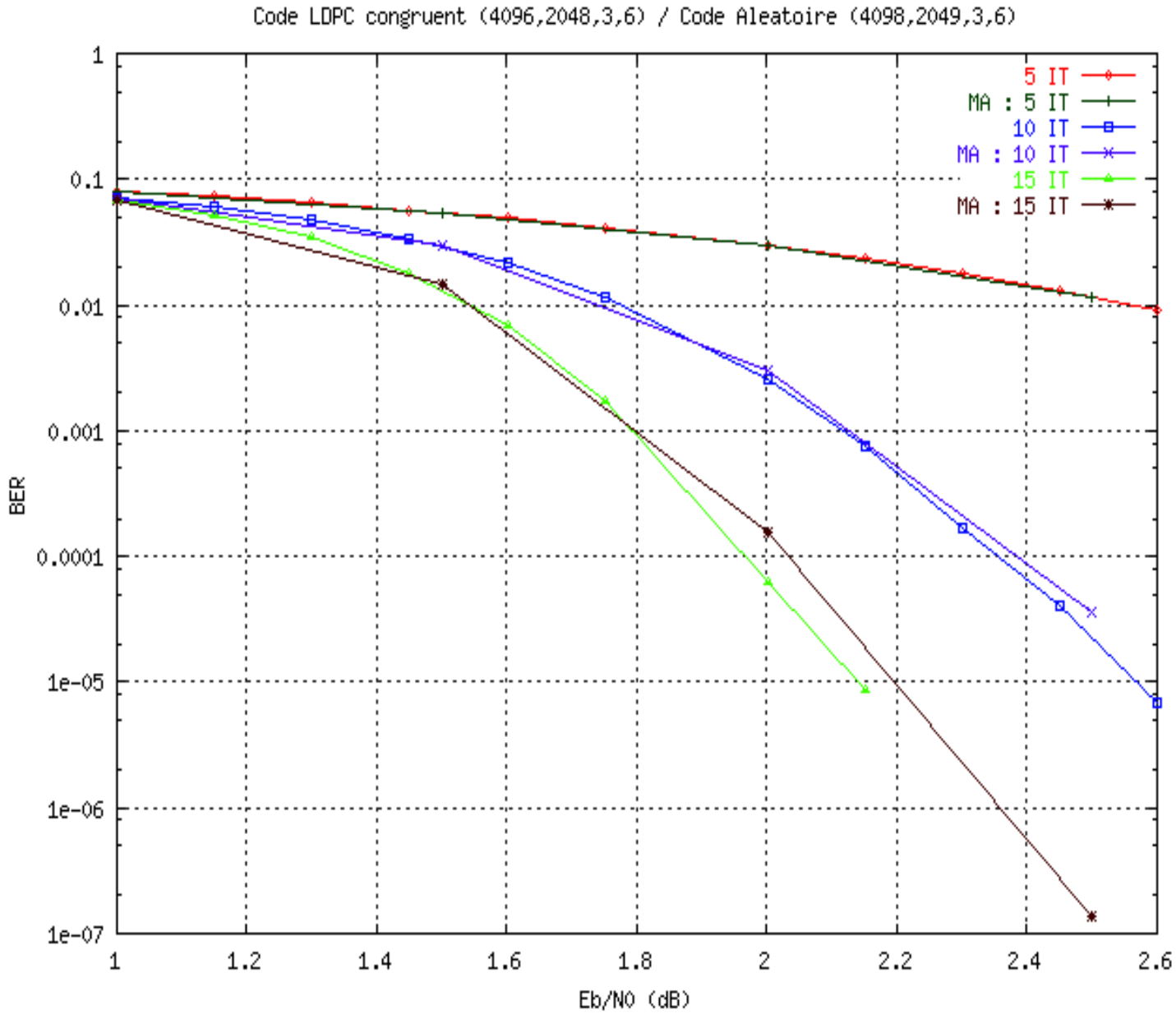


2. Une classe de codes facilement implantables : les codes congruents

- Le caractère pseudo-aléatoire de la matrice de décodage est incompatible avec un stockage ou un calcul simple des adresses des messages
- ➔ Adresses calculées par un générateur congruent et récursif [Prabhakar2002]
  - ➔ Pas de stockage de la matrice
  - ➔ Paramétrage simple du générateur congruent

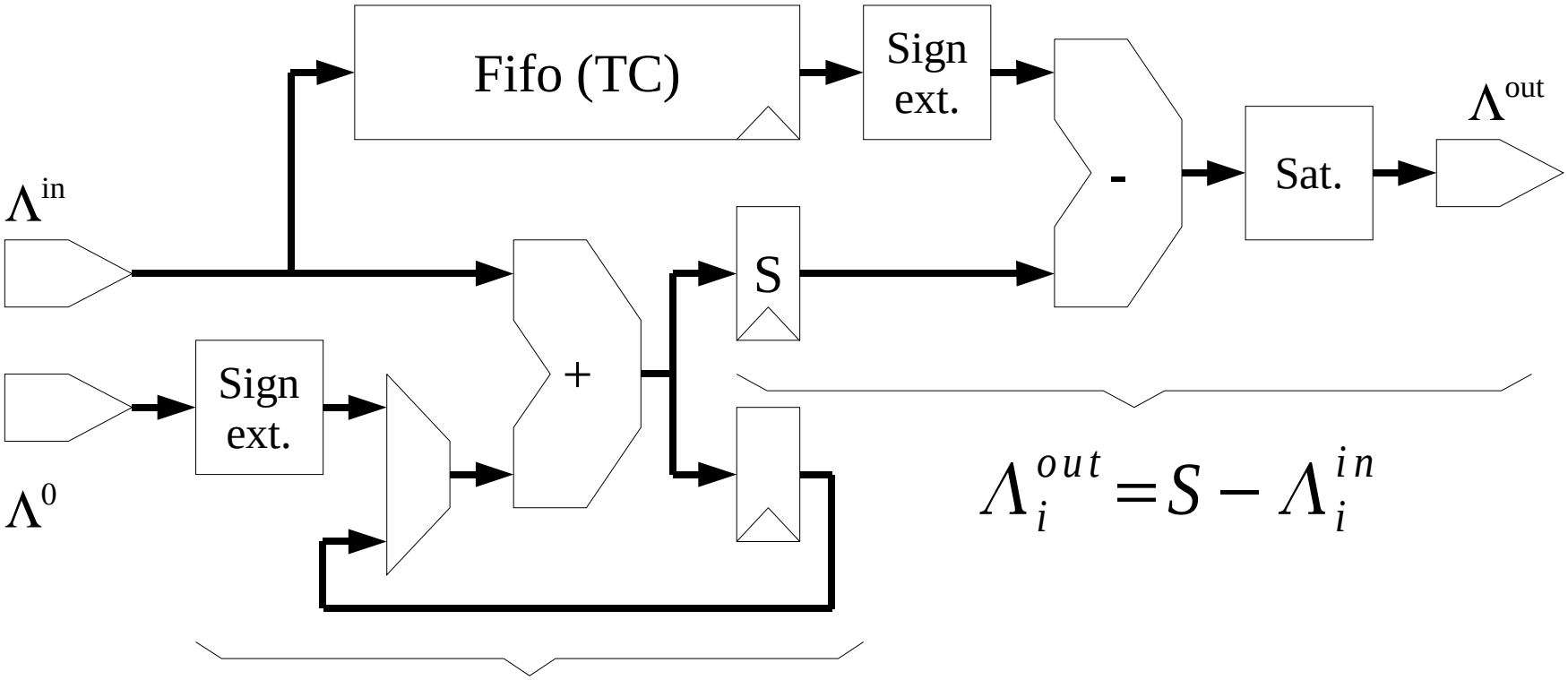
$$\Pi_{n+1} = \left(a \times \Pi_n + b \right) mod \left(N_b \right)$$

Performances des codes congruents réguliers :



3. Architecture du décodeur

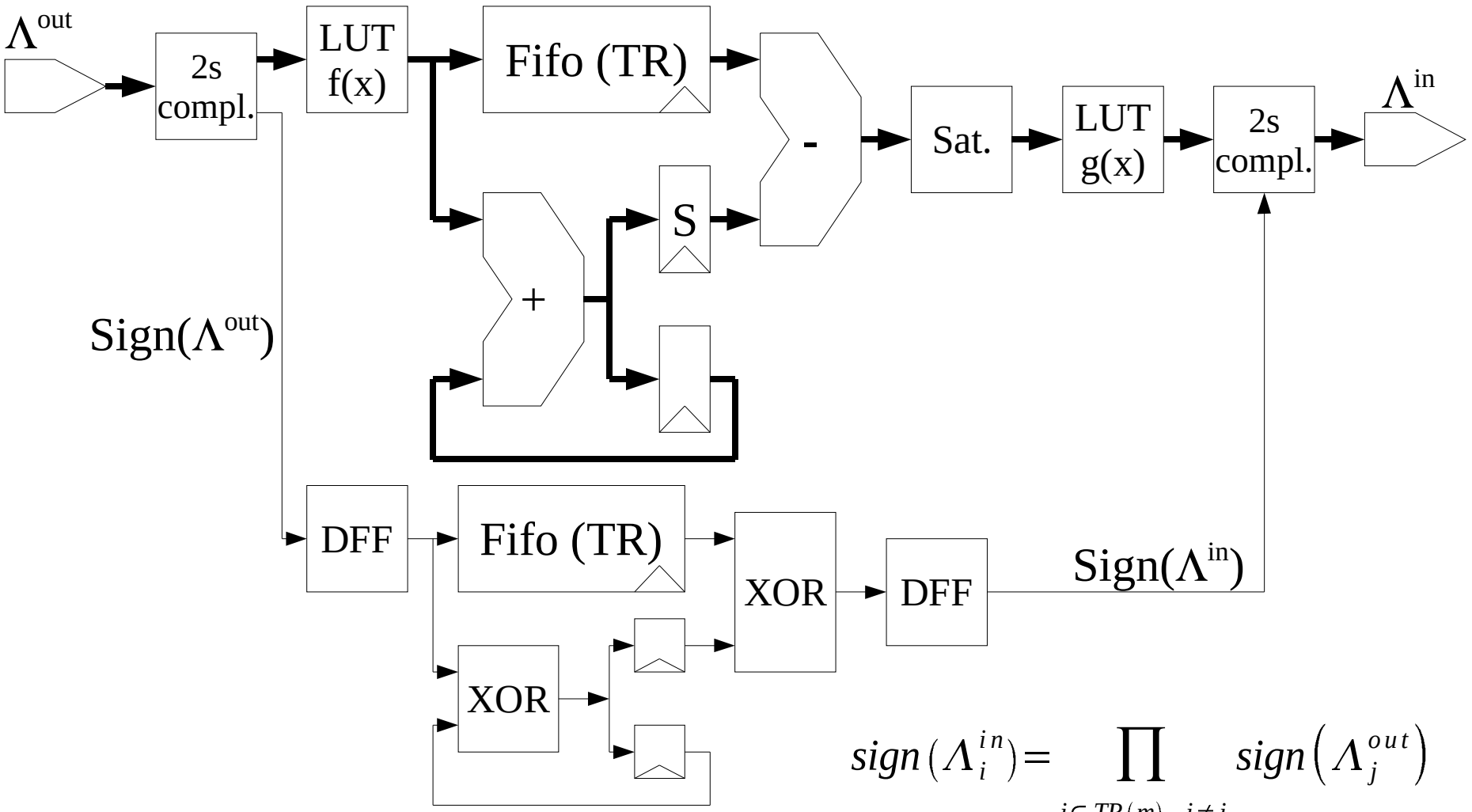
3.1) Implantation pipeline du processeur V



3.2) Implantation pipeline du processeur C

$$S = \sum_{i \in TR(m)} \mathrm{f}\left(\frac{\left|\Lambda_i^{out}\right|}{2}\right)$$

$$\left|\Lambda_i^{in}\right| = \mathrm{g}\left(S - \mathrm{f}\left(\frac{\left|\Lambda_i^{out}\right|}{2}\right)\right)$$



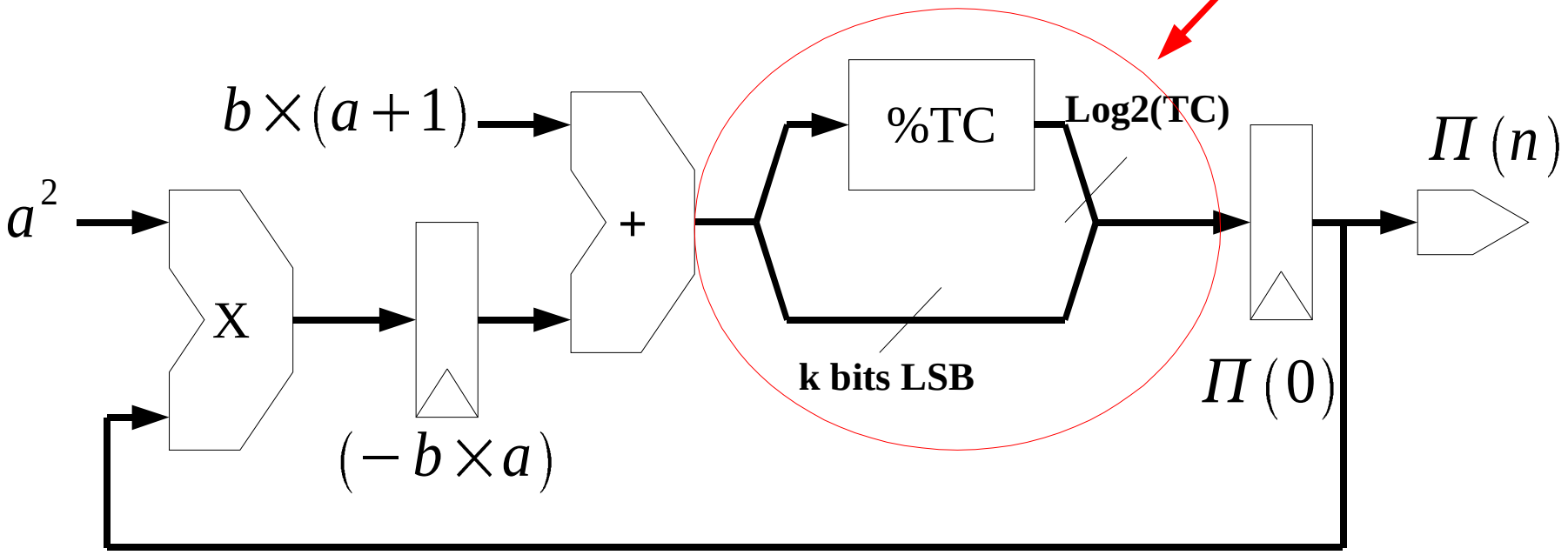
$$sign\left(\Lambda_i^{in}\right) = \prod_{j \in TR(m),\ j \neq i} sign\left(\Lambda_j^{out}\right)$$

3.3) Implantation du générateur congruent

Implantation pipeline du générateur pseudo-aléatoire récursif :

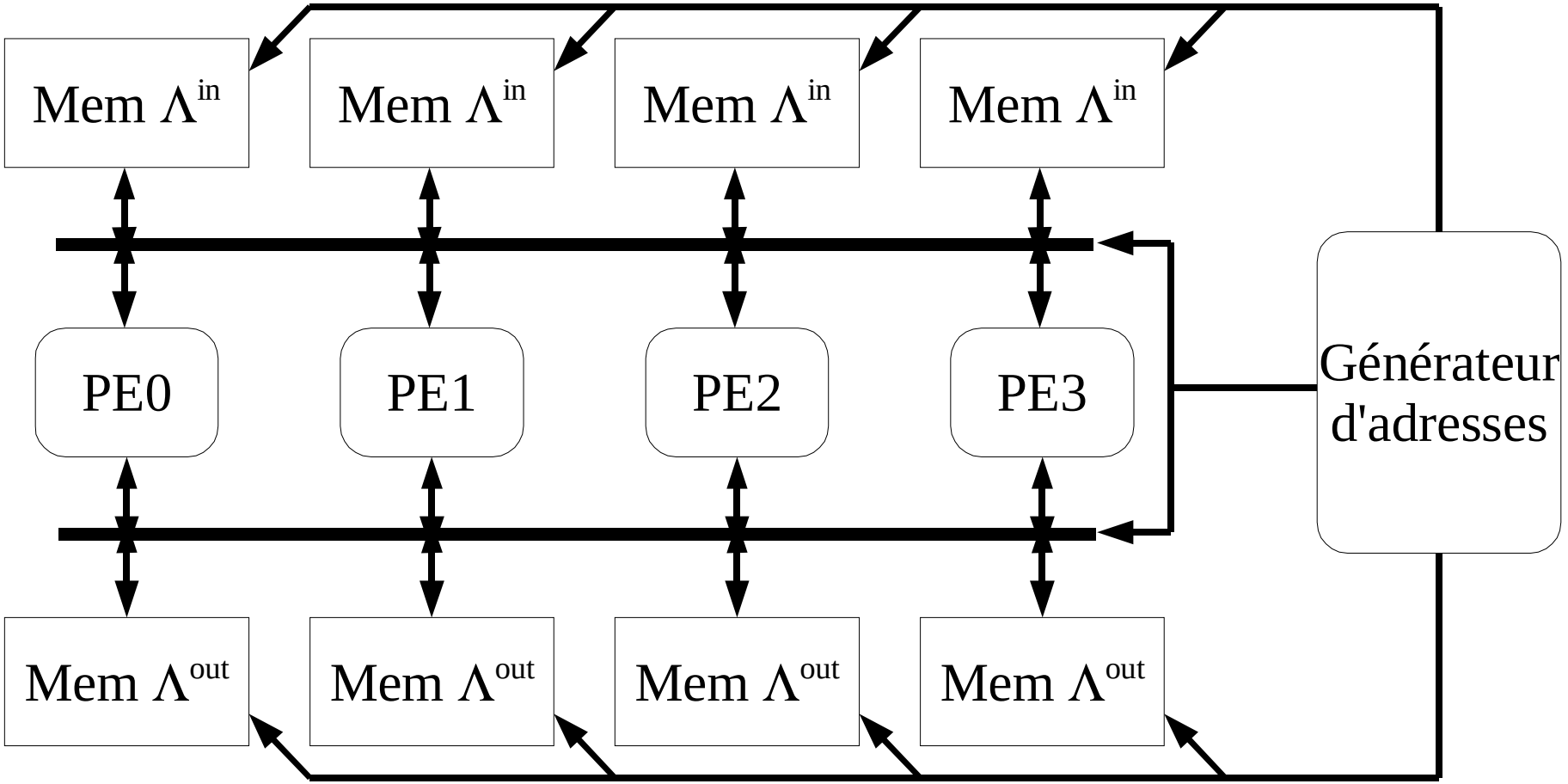
$$\Pi(n+1)=\left(a\times\Pi(n)+b\right)mod\left(N_b\right)$$ Impossible en 1 cycle à 100Mhz !

$$\Rightarrow \Pi(n+2)=\left(a^2\times\Pi(n)+b\times(a+1)\right)mod\left(TC.2^k\right)$$



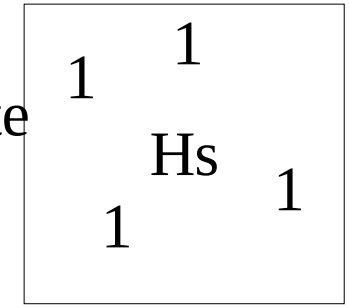
Parallélisation du décodage

Processeurs parallèles synchrones + 2 réseaux de routage

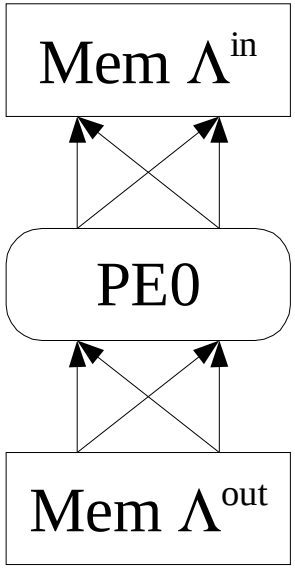


Construction du code parallèle (2PE)

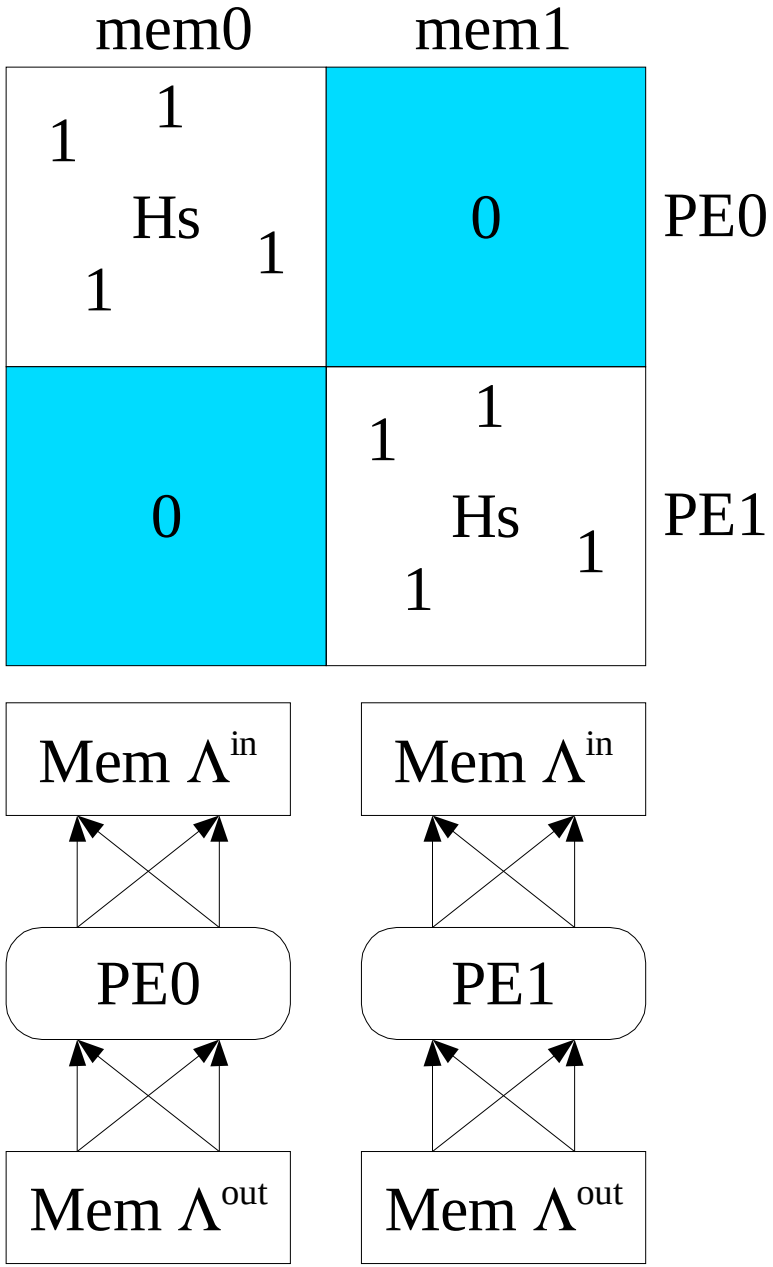
matrice structurante congruente



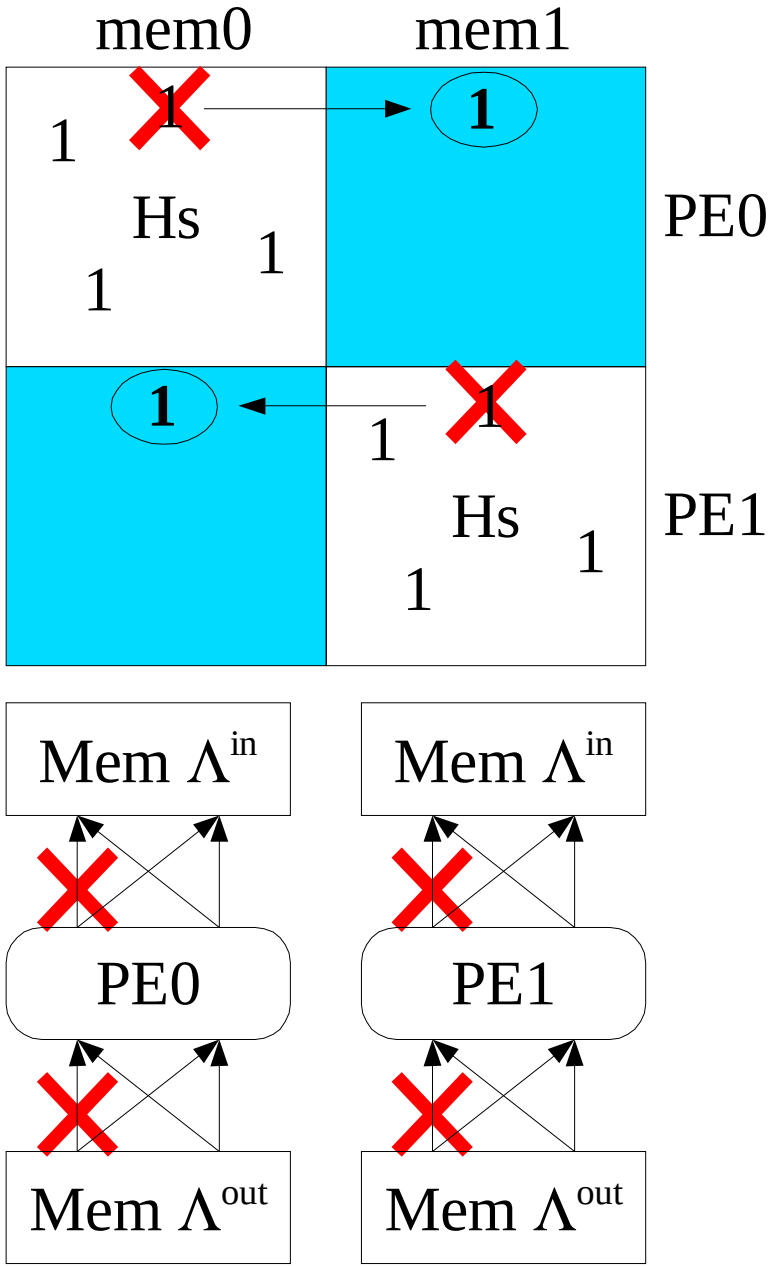
phase de calcul du processeur C



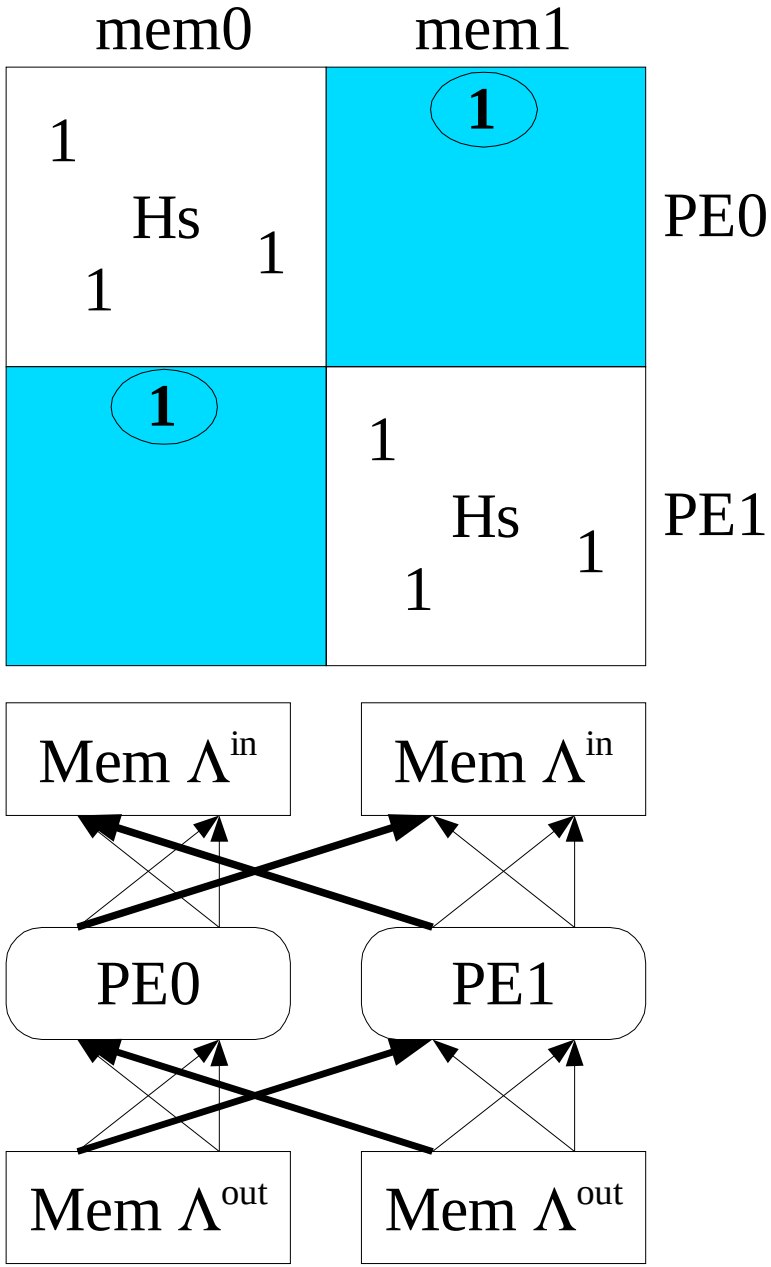
Construction du code parallèle (2PE)



Construction du code parallèle (2PE)

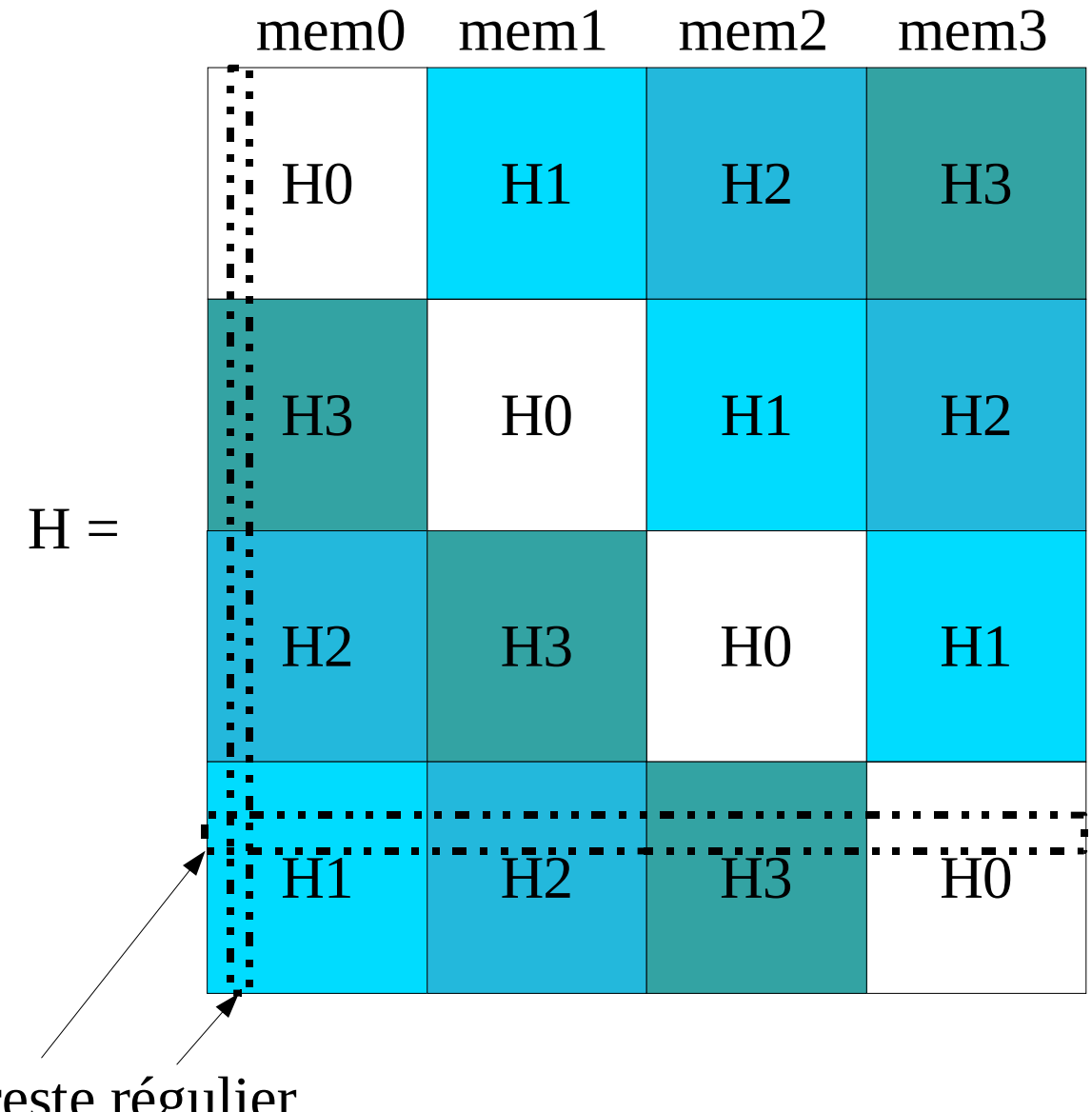


Construction du code parallèle (2PE)

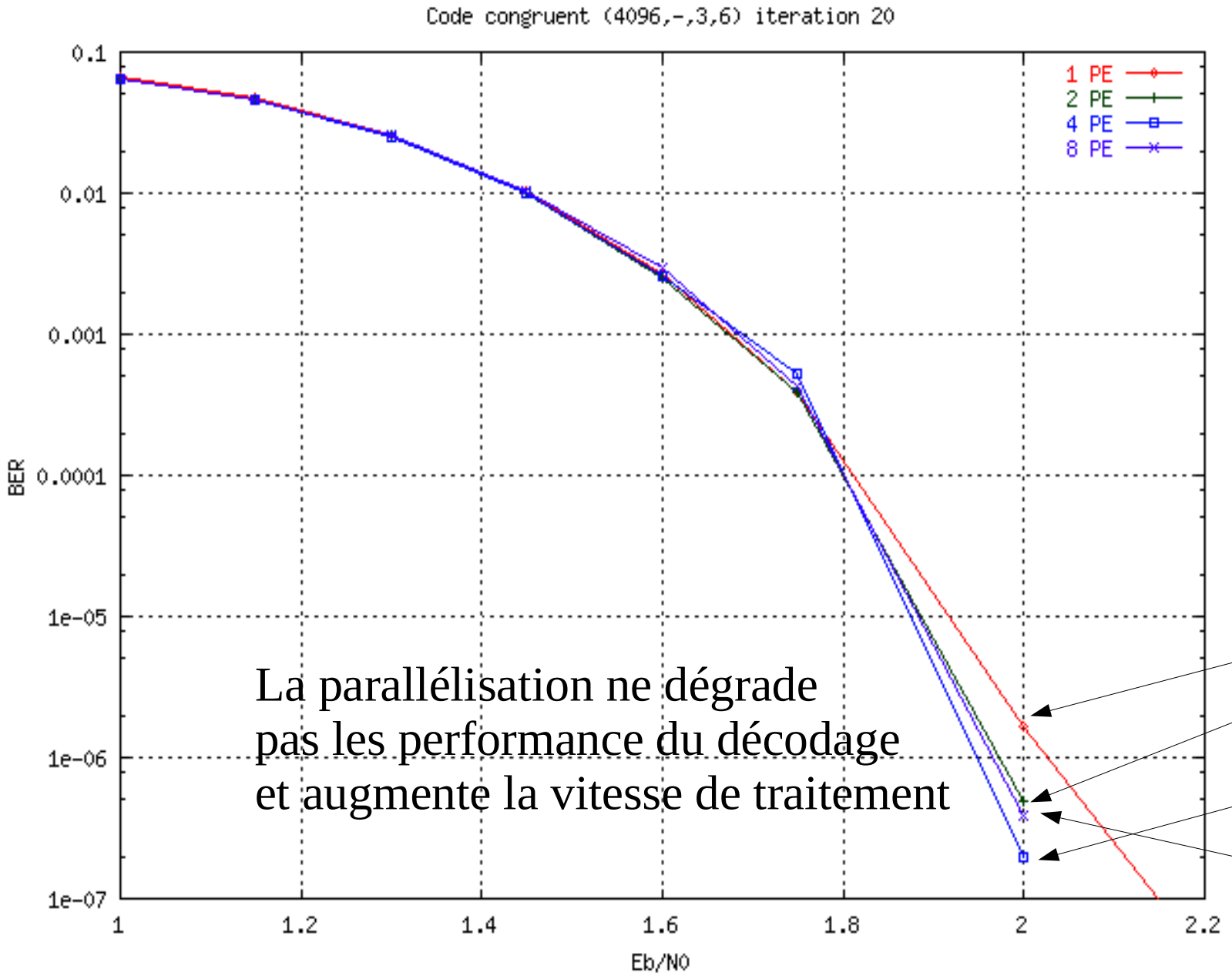


Création des branches inter-partitions

Construction du code parallèle (4 PE)



Effets de la parallélisation



La parallélisation ne dégrade pas les performance du décodage et augmente la vitesse de traitement

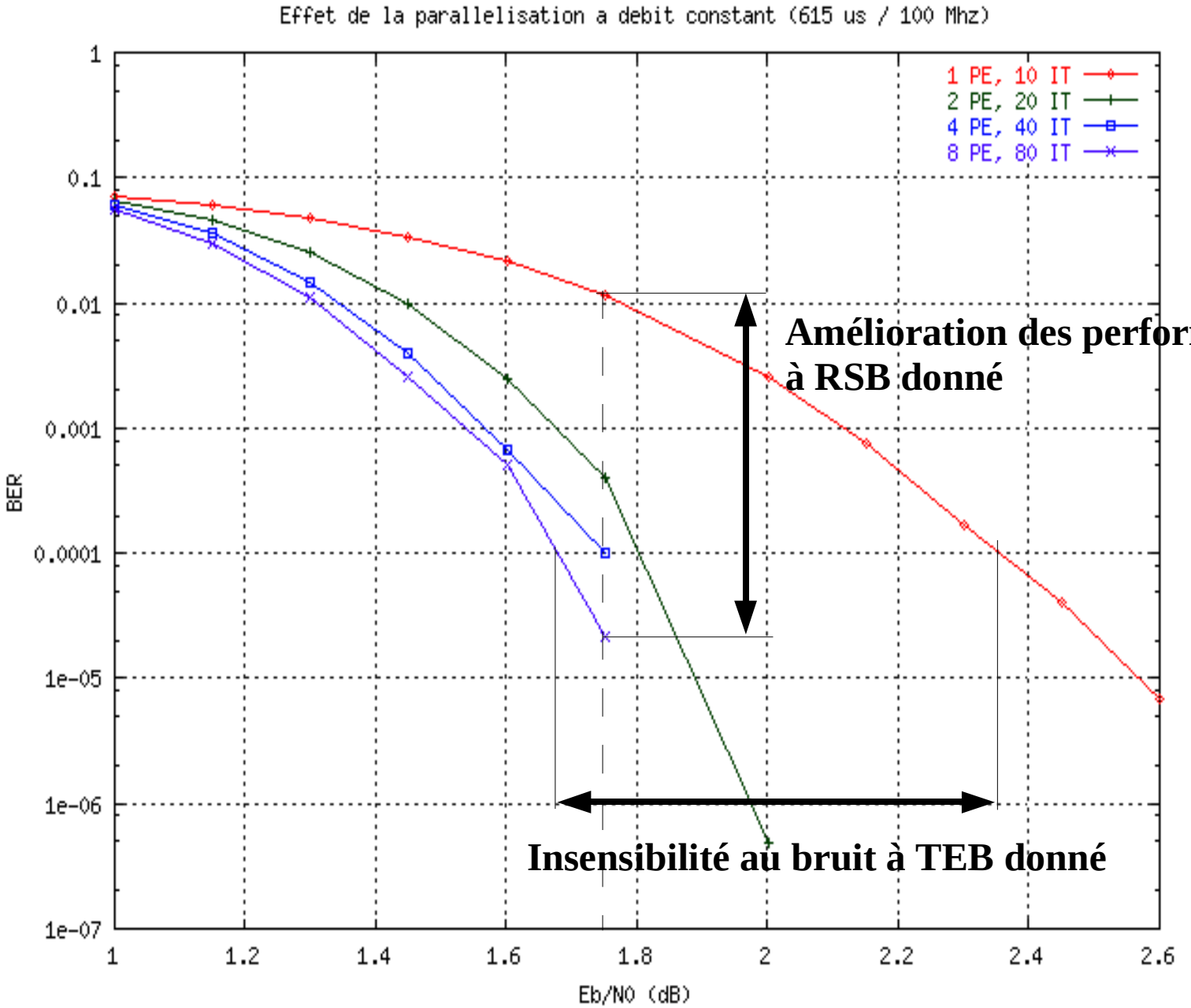
833 Kb/s

1.6 Mb/s

3.33 Mb/s

6.66 Mb/s

Effets de la parallélisation



4. Conséquences du calcul en précision finie...

Conséquences du Codage en Précision Finie

David Declercq[†] & Francois Verdier[†]

ETIS ENSEA/UCP/CNRS-8051

6, avenue du Ponceau 95000 Cergy-Pontoise

declercq@ensea.fr

[†] Dans le cadre de l'action JemSTIC CNRS

Conséquences du Codage en Précision Finie

Problèmes posés :

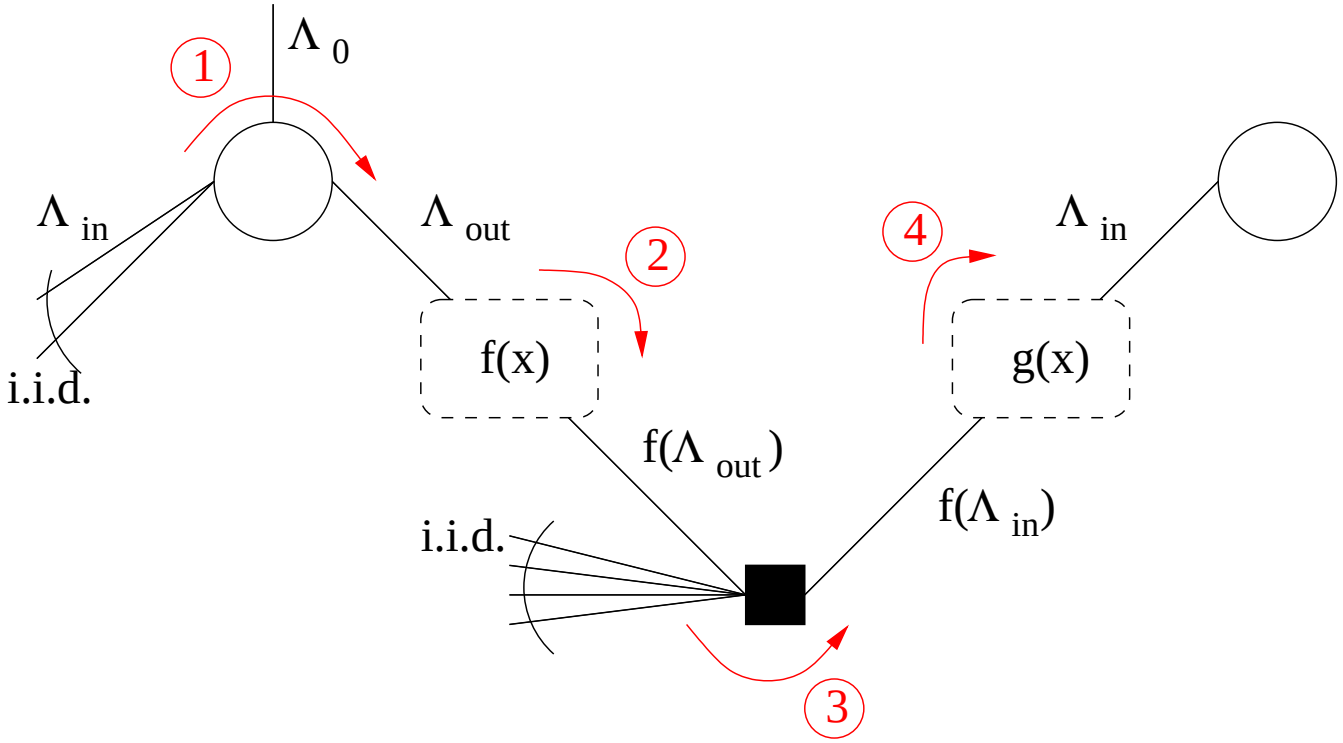
- Quantification des messages.
- Quantification des LUT.

  $\Rightarrow$ Peut-on prévoir l'effet sur les performances du code.

  - dégradation des performances,
  - apparition d'un plancher de convergence.

Outil statistique : évolution de la densité de probabilité des messages au cours des itérations.

Evolution de Densité



$$f(x) = \log \tanh\left(\frac{|x|}{2}\right) \qquad g(x) = 2 \tanh^{-1} \exp(x)$$

Hypothèses de travail

DYN : nombre de bits codant la partie entière

PREC : nombre de bits codant la partie fractionnaire

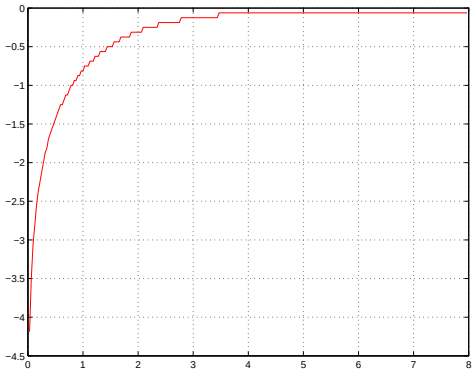
$\{\Lambda_0(\omega), \Lambda_{in}(\omega), \Lambda_{out}(\omega)\}$

$\Rightarrow$ V.A. discrètes à support dans $\left[-DYN + \frac{1}{2^{PREC}} : \frac{1}{2^{PREC}} : DYN - \frac{1}{2^{PREC}}\right]$

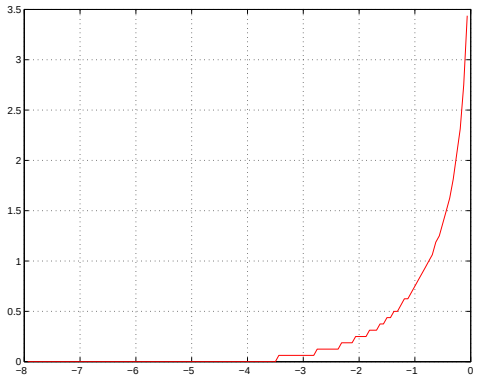
$\{f(\Lambda_{in}(\omega)), f(\Lambda_{out}(\omega))\}$

$\Rightarrow$ V.A. discrètes à support dans $\left[-DYN + \frac{1}{2^{PREC}} : \frac{1}{2^{PREC}} : 0\right]$

Fonction $f(x) = \log \tanh\left(\frac{|x|}{2}\right)$



Fonction $g(x) = 2 \tanh^{-1} \exp(x)$



Evolution des Densités pour 1 Itération

1 $\Lambda_{out} = \Lambda_0 + 2\,\Lambda_{in}$

$$\Rightarrow p\left(\Lambda_{out}\right) = p\left(\Lambda_0\right) * p\left(\Lambda_{in}\right) * p\left(\Lambda_{in}\right)$$

2 $f\left(\Lambda_{out}\right) = \log\ \tanh\left(\frac{|\Lambda_{out}|}{2}\right)$

$\Rightarrow$ Changement de variable + quantification.

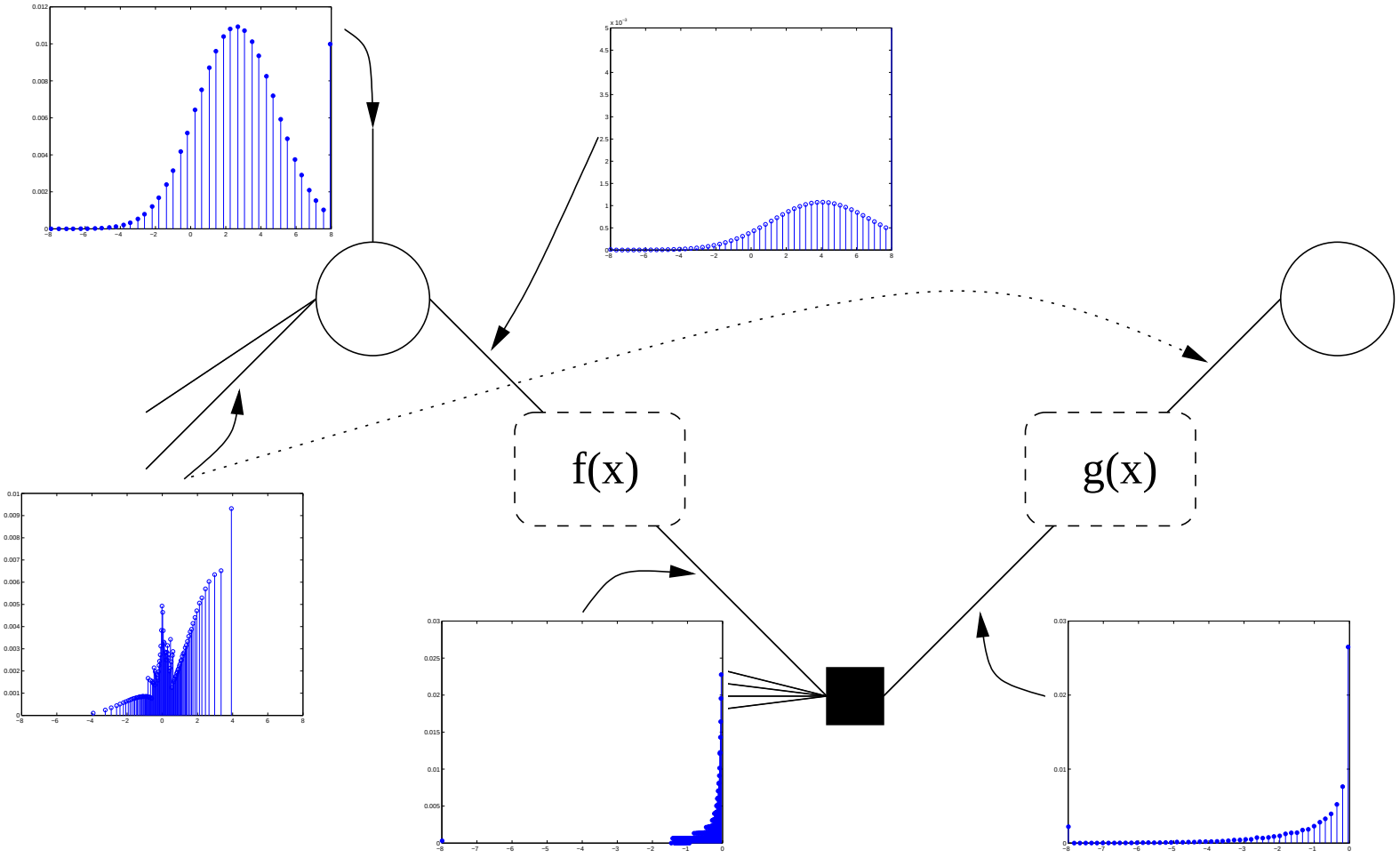
3 $f\left(\Lambda_{in}\right) = 5\, f\left(\Lambda_{out}\right)$

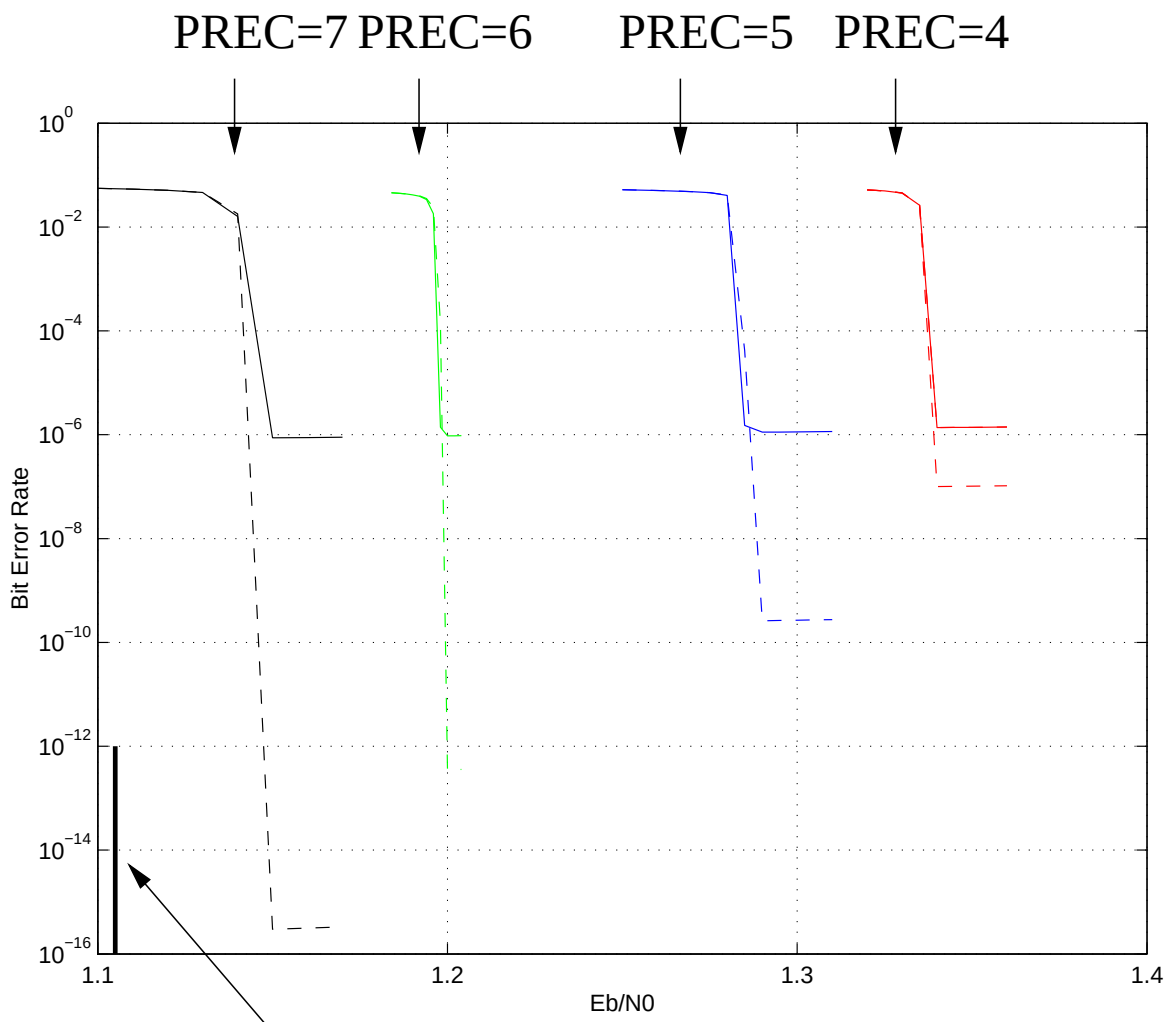
$$\Rightarrow p\left(f\left(\Lambda_{in}\right)\right) = p\left(f\left(\Lambda_{out}\right)\right)^{*5}$$

4 $\Lambda_{in} = 2\ \tanh^{-1}\ \exp\left(f\left(\Lambda_{in}\right)\right)$

$\Rightarrow$ Changement de variable + quantification.

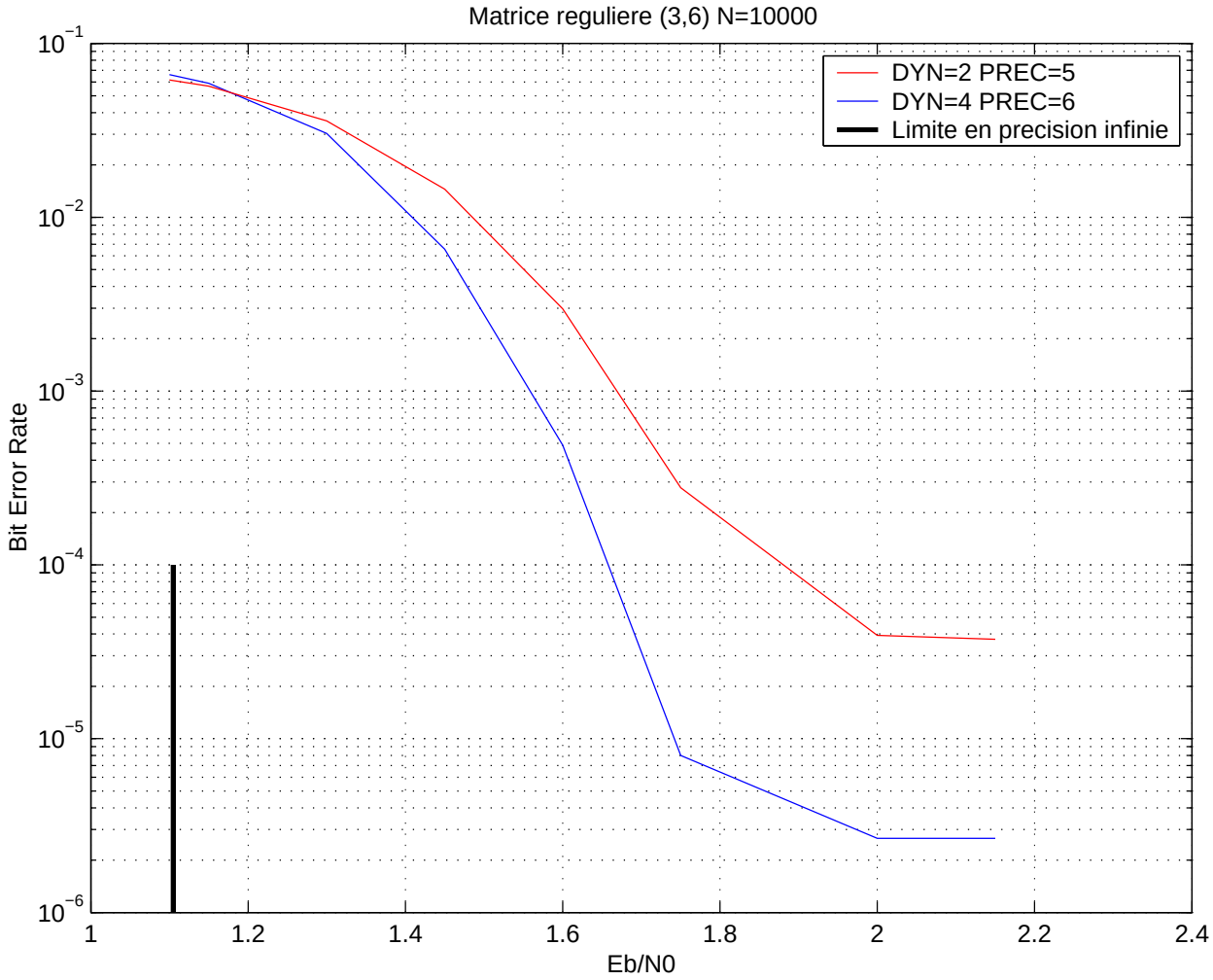
Forme des Densités




PREC=7
PREC=6
PREC=5
PREC=4
Bit Error Rate
Eb/N0
1.1
1.2
1.3
1.4
Limite en précision infinie

DYN=3

DYN=4


Matrice reguliere (3,6) N=10000
DYN=2 PREC=5
DYN=4 PREC=6
Limite en precision infinie
Bit Error Rate
Eb/N0
10^-1
10^-2
10^-3
10^-4
10^-5
10^-6
1
1.2
1.4
1.6
1.8
2
2.2
2.4

Conclusion et Perspectives

1. Modèle de parallélisation des codes LDPC.
   - architecture simple et scalable,
   - stockage du graphe de façon congruente,
   - méthode de parallélisation sans dégradation de performances.
2. Parallélisme pour les codes LDPC irréguliers
   - problèmes de stockage du graphe,
   - effets de la précision finie.