Décodeurs génériques de codes LDPC

Frédéric Guilloud

Jean-Luc Danger (Télécom Paris)
Emmanuel Boutillon (LESTER)

Télécom Paris – 2 juillet 2004
SPRING – 1999 – IST 12342

Plan

- 
- Les codes LDPC
- L'algorithme λ-Min
- Architecture Générique
- Formalisation
- Conclusion et Perspectives

Le projet SPRING

- Projet Européen
- Participants :
  - Schlumberger
  - CNM Séville
  - UCL Louvain la neuve
  - Télécom Paris

- http://www.imse.cnm.es/spring

Plan

- Introduction
- Les codes LDPC
- L'algorithme λ-Min
- Architecture Générique
- Formalisation
- Conclusion et Perspectives

Historique

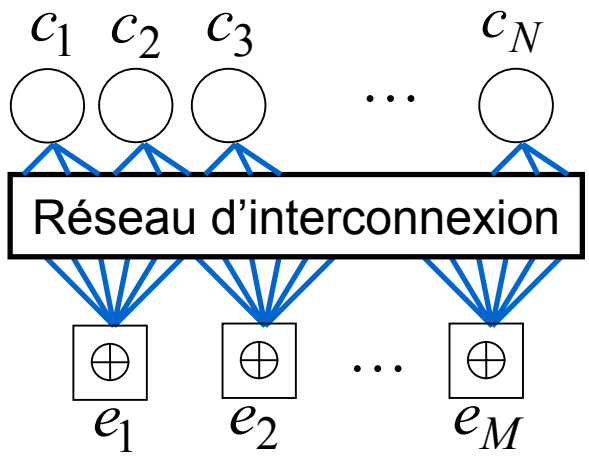
Gallager	LDPC	1962
Berrou et. al.	Turbo Codes	1993
MacKay	LDPC	1995
Wiberg	Graphes	1995
Richardson, Urbanke	Density Evolution	2001

- premières architectures de décodeurs

LDPC : Définitions

- « Low Density Parity Check »
- Code en bloc dont la matrice de parité **H** est peu dense
- Graphe bipartite
  - Variables
  - Parités

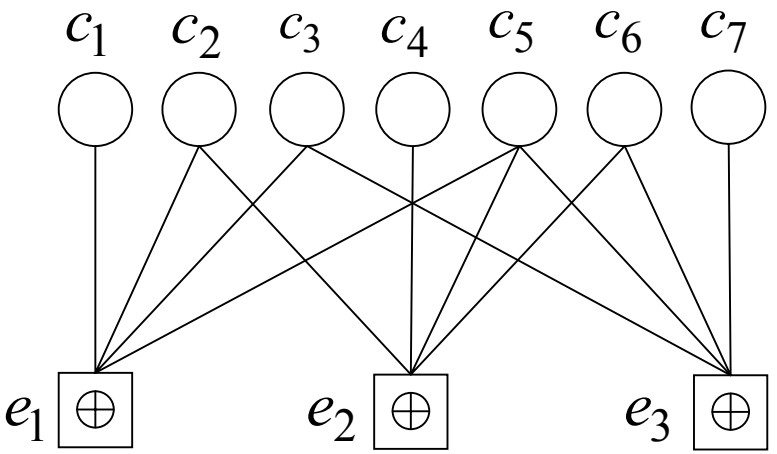
$$\begin{pmatrix} e_1 \\ \vdots \\ e_M \end{pmatrix} = \underbrace{\begin{pmatrix} h_1 \\ \vdots \\ h_M \end{pmatrix}}_{\mathbf{H}} \times \underbrace{\begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix}}_{\mathbf{c}^t} = \mathbf{0}$$



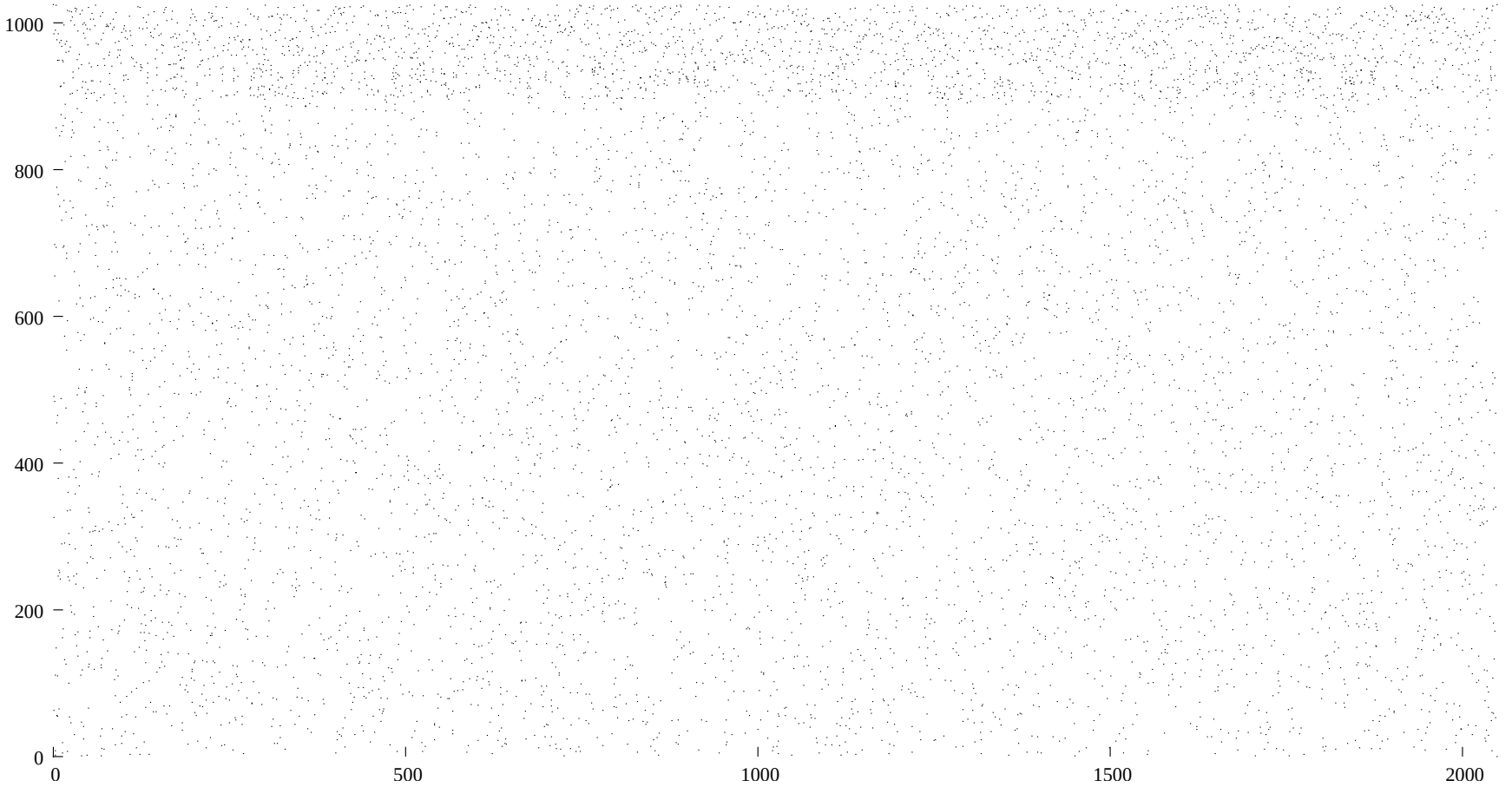
Caractéristiques

- Exemple : Code de Hamming
- Caractéristiques :
  - Taille : N = 7
  - Rendement : *R = 4/7*
    - *R ≥ 1 - M / N*
  - Degré
- Mais dense …
  - $M.2^M/2$ entrées non nulle dans **H**

$$\mathbf{H}=\begin{array}{c} \begin{matrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 \end{matrix} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \end{array} \begin{matrix} e_1 \\ e_2 \\ e_3 \end{matrix}$$

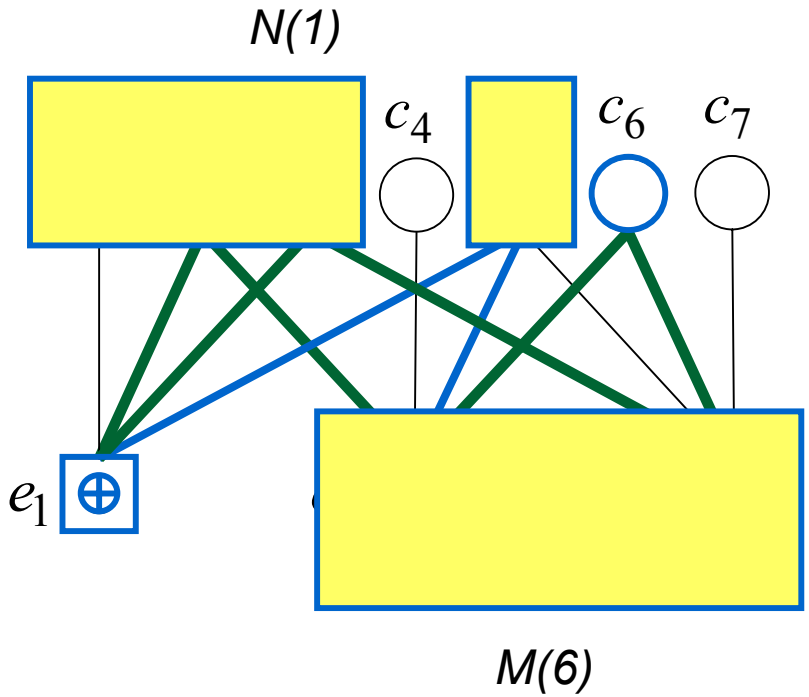


• Code LDPC : N=2048, R=0.5

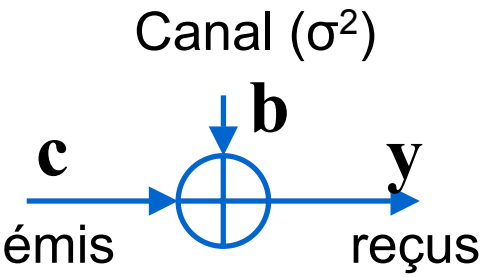


Graphes bipartites

- Cycles
  - Taille 4
  - Taille 6
- Ensembles de parités
  - *M(n), n Є {1,…,N}*
- Ensembles de variables
  - *N(m), n Є {1,…,M}*



Codes LDPC : décodage (1)



- Bruit blanc additif gaussien
- Décodage optimal des symboles :

$$\hat{c}_n = \text{Arg} \max_{s \in \text{alphabet}} \Pr(c_n = s | \mathbf{y})$$

- Cas binaire :

$$\hat{c}_n = 0 \Leftrightarrow \Pr(c_n = 0|\mathbf{y}) > \Pr(c_n = 1|\mathbf{y})$$
$$\hat{c}_n = 1 \Leftrightarrow \Pr(c_n = 0|\mathbf{y}) < \Pr(c_n = 1|\mathbf{y})$$

- Logarithme du rapport de vraissemblance (LLR)

$$\hat{c}_n = \text{sign} \log \frac{\Pr(c_n = 0|\mathbf{y})}{\Pr(c_n = 1|\mathbf{y})} \quad \text{(BPSK)}$$

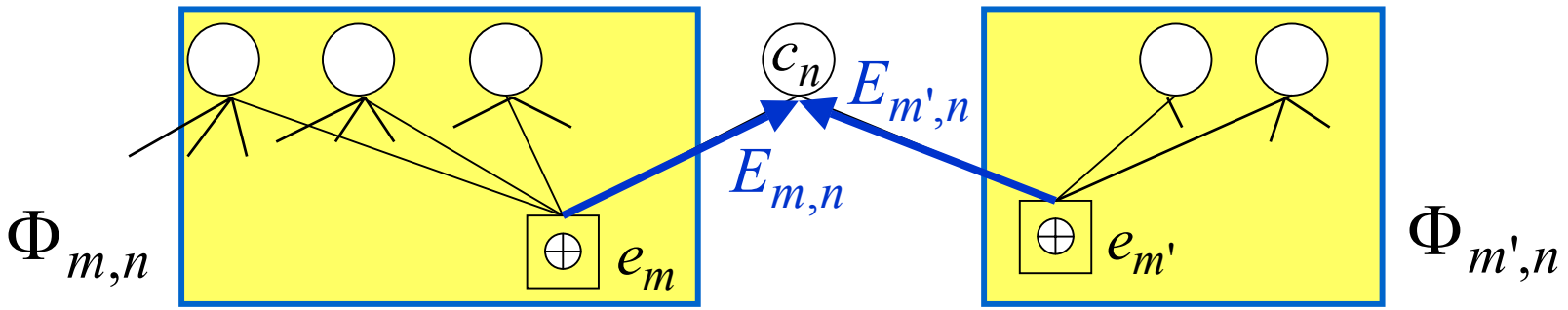
Codes LDPC : décodage (2)

- Règle de Bayes : $\log \frac{\Pr(c_n = 0|\mathbf{y})}{\Pr(c_n = 1|\mathbf{y})} = \log \frac{\Pr(y_n|c_n = 0)}{\Pr(y_n|c_n = 1)} + \log \frac{\Pr(c_n = 0|y_{n' \neq n})}{\Pr(c_n = 1|y_{n' \neq n})}$

  T_n I_n $E_n = \sum_{m \in M(n)} E_{m,n}$

- Information
  - Totale
  - Intrinsèque (canal)
  - Extrinsèque (code)
  - Extrinsèque de branche (Graphe sans cycle)

$$E_{m,n} = \log \frac{\Pr(\Phi_{m,n} = 0|y_{n' \neq n})}{\Pr(\Phi_{m,n} = 1|y_{n' \neq n})}$$

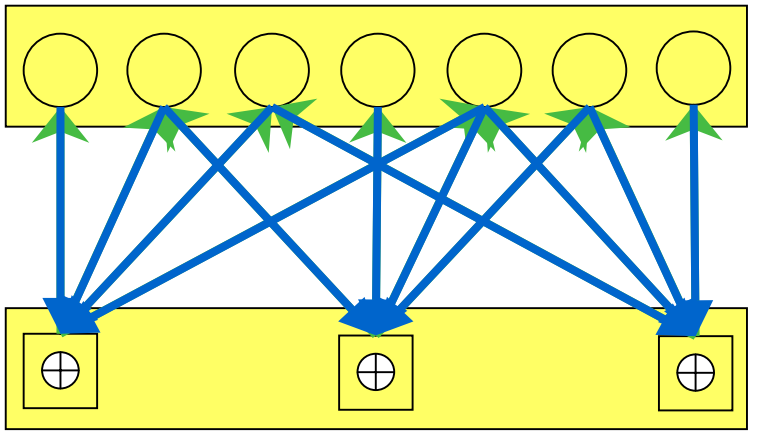


Algorithme BP

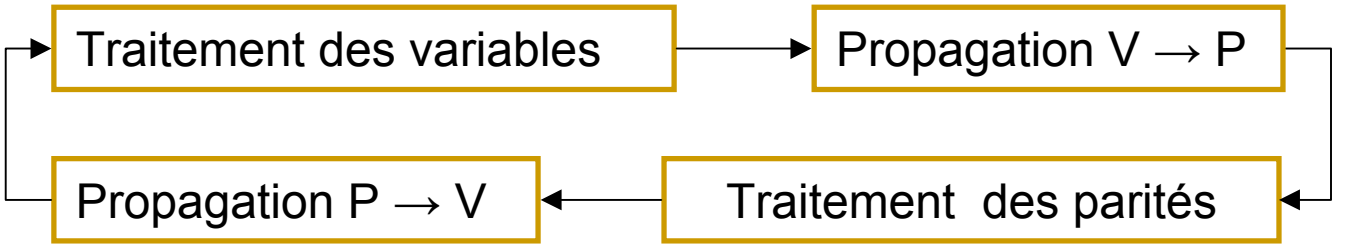
- Propagation de croyance (Belief Propagation)
- Initialisation :
  - $I_n = 2y_n/\sigma^2$
  - Extrinsèque branche : $E_{m,n}=0$
  - Fiabilités : $T_{m,n}=0$
- Itérations : tous les messages sont traités

$$T_{m,n} = T_n - E_{m,n}$$

$$= I_n + \sum_{m' \in M(n) \setminus m} E_{m',n}$$



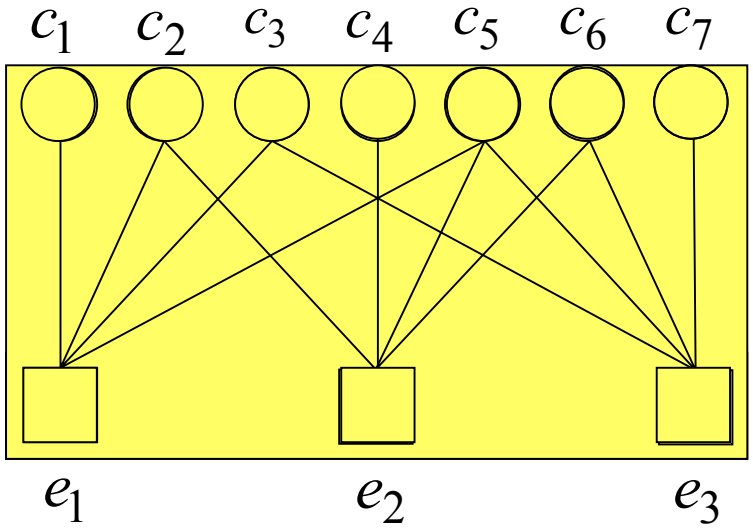
$$E_{m,n} = 2 \tanh^{-1} \prod_{n' \in N(m) \setminus n} \tanh \frac{T_{m,n'}}{2}$$



Problématiques

- Traitement des parités complexes
- Besoin en mémoire
  - Taille
    - Ex : DVB-S2 :

    64k x 3 = 192000 messages
  - Accès
- Architecture, irrégularité
- Séquencement (cycles)



$$E_{m,n} = 2\tanh^{-1} \prod_{n' \in N(m) \setminus n} \tanh \frac{T_{m,n'}}{2}$$

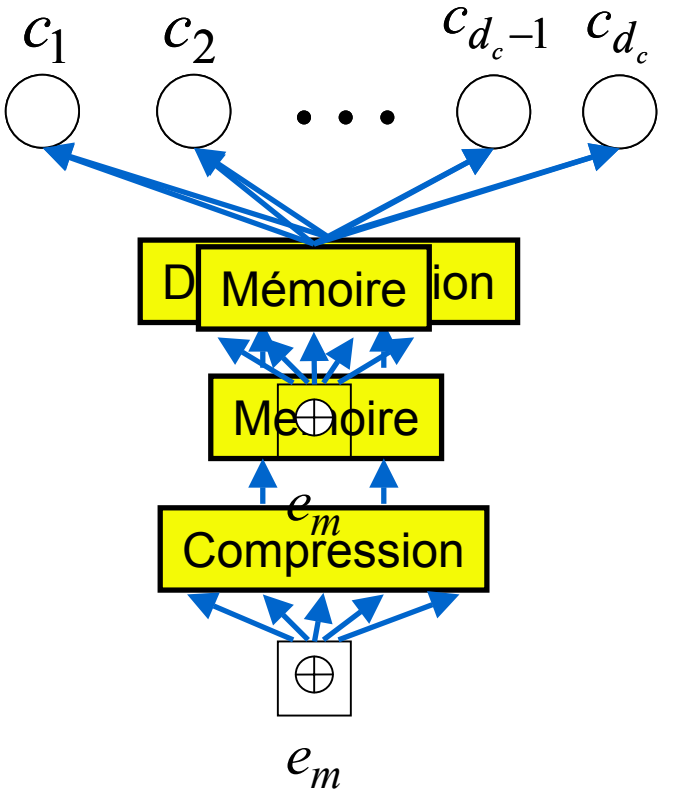
Plan

- Introduction
- Les codes LDPC
- L'algorithme λ-Min
- Architecture Générique
- Formalisation
- Conclusion et Perspectives

Analyse (1)

- Taille des mémoires nécessaires au stockage des messages C → V

→ Compression de l'information

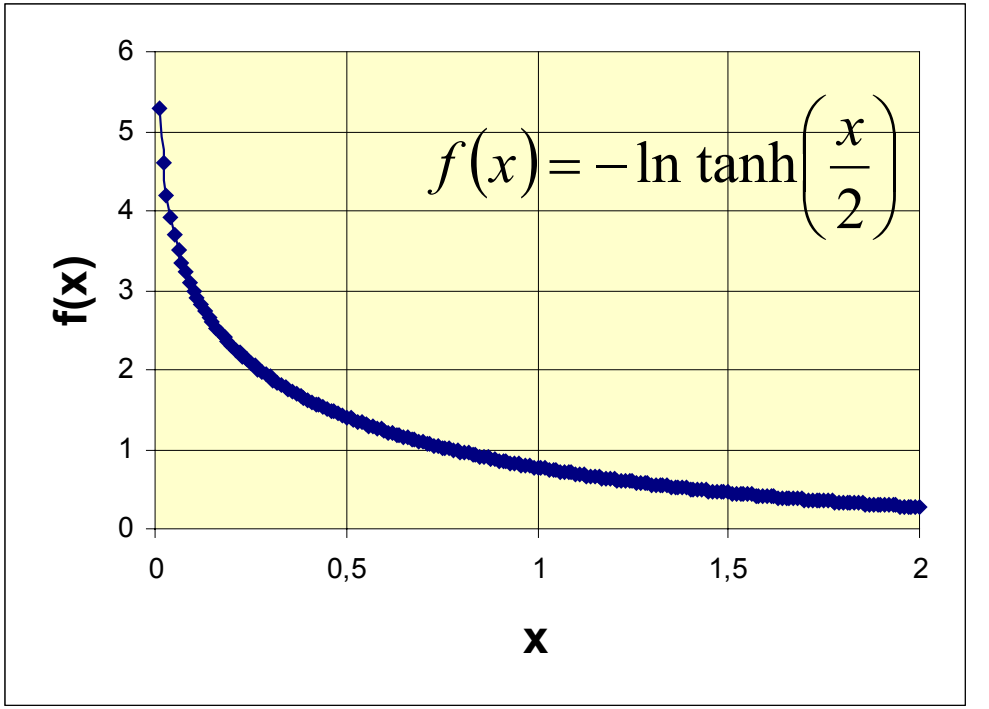


Analyse (2)

- Complexité des traitements des parités

$$\left|E_{m,n}\right| = f^{-1}\left(\sum_{n'\in N(m)\setminus n} f\left(\left|T_{m,n'}\right|\right)\right)$$

→ Seules les fiabilités faibles peuvent être retenues

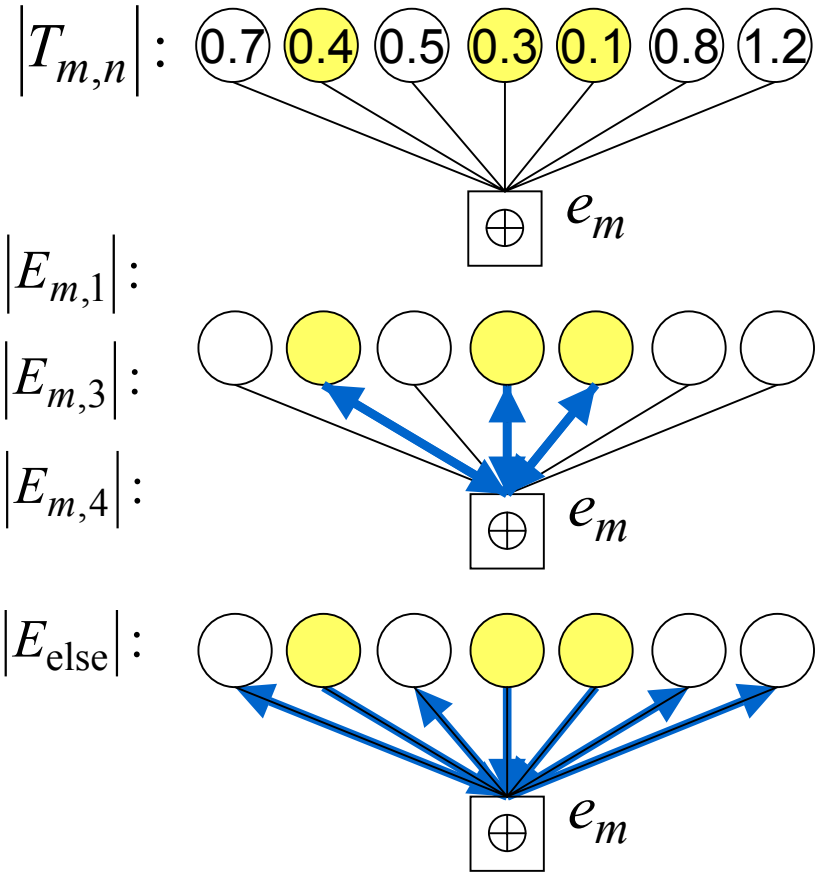


Algorithme λ-Min

- Algorithme sous optimal
- On ne garde que les λ plus faibles fiabilités, indicées par $N_\lambda(m)$

$$\left|E_{m,n}\right| = f^{-1}\left(\sum_{n' \in N_\lambda(m) \setminus n} f\left(\left|T_{m,n'}\right|\right)\right)$$

- Si n Є $N_\lambda(m)$: λ calculs sur λ-1 données
- Sinon : 1 calcul sur λ données



$$E_\lambda = \left\{E_{m,1}, E_{m,3}, E_{m,4}, E_{\text{else}}\right\}$$

Gain mémoire (1)

- Algorithme BP :
  - d_c messages x (Nb bits)
  - d_c signes (1 bit)
- Algorithme λ-Min :
  - d_c messages :
    - d_c signes
    - λ+1 modules sur Nb bits
  - λ indices des minimum : λ $\log_2(d_c)$ bits

- Taux de compression :

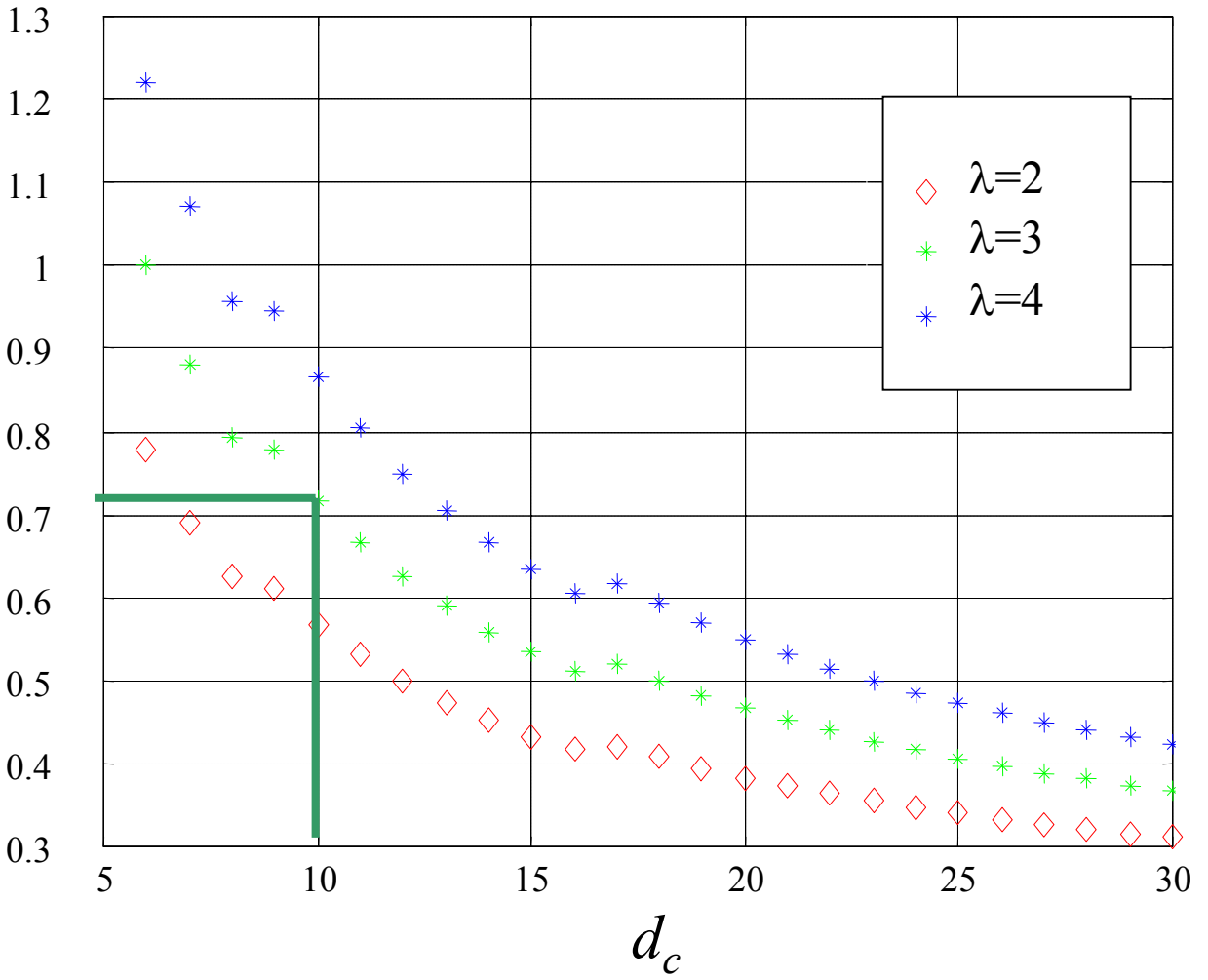
$$\frac{1+d_c+\lambda\log_2(d_c)+(\lambda+1)N_b}{d_c(N_b+1)}$$

Gain mémoire (2)

Taux de compression par rapport à BP

- Exemple :
  - d_c = 10
  - λ =3

Économie mémoire: ~30%



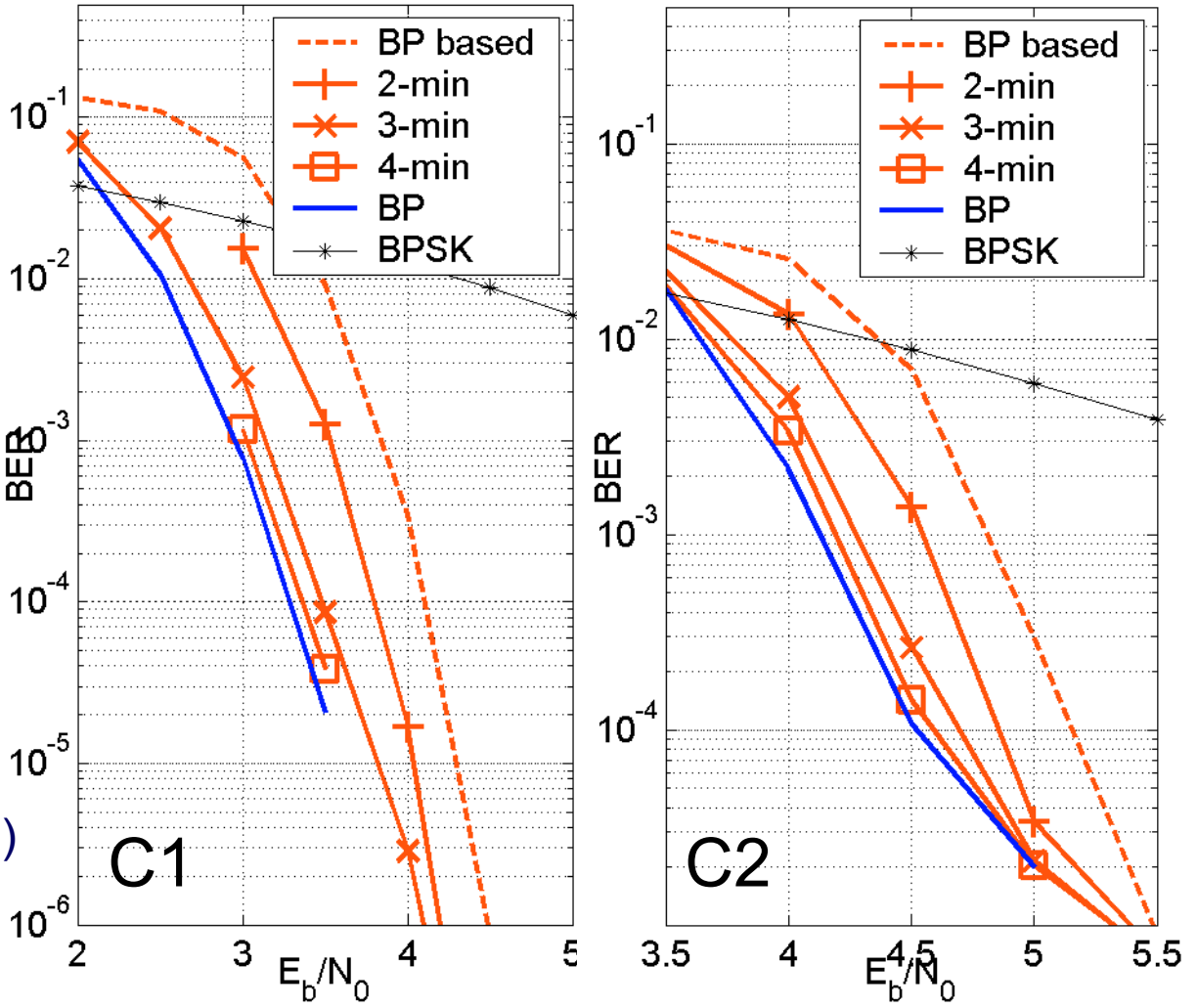
Autres algorithmes sous optimaux

Développés indépendamment :

- APP (A-Posteriori Probabilities) (MacKay)
  - Pas de mémorisation des extrinsèques de branches
    - Gain mémoire
    - Performances moindres, nombre d'itération limité
- BP-Based (Fossorier)
  - Équivalent au 2-min sans les calculs de LLR
    - Performances moindres

Performances

- Code C1
  - C1 : (5,10)
    N= 816
    (mackay)
- Code C2
  - Irrégulier
    taux = 0.85
    N=2000
    (41~42 bits/parité)
    (Urbanke)



Conclusion

- Algorithme réduisant :
  - la complexité des calculs
  - la taille de mémoire de branche
- Architecture associée : compression de la mémoire
- Nouvelle proposition
  - A-Min* (Jones et. al., MilCom 2003)

Plan

- Introduction
- Les codes LDPC
- L'algorithme λ-Min
- **Architecture Générique**
- Formalisation
- Conclusion et Perspectives

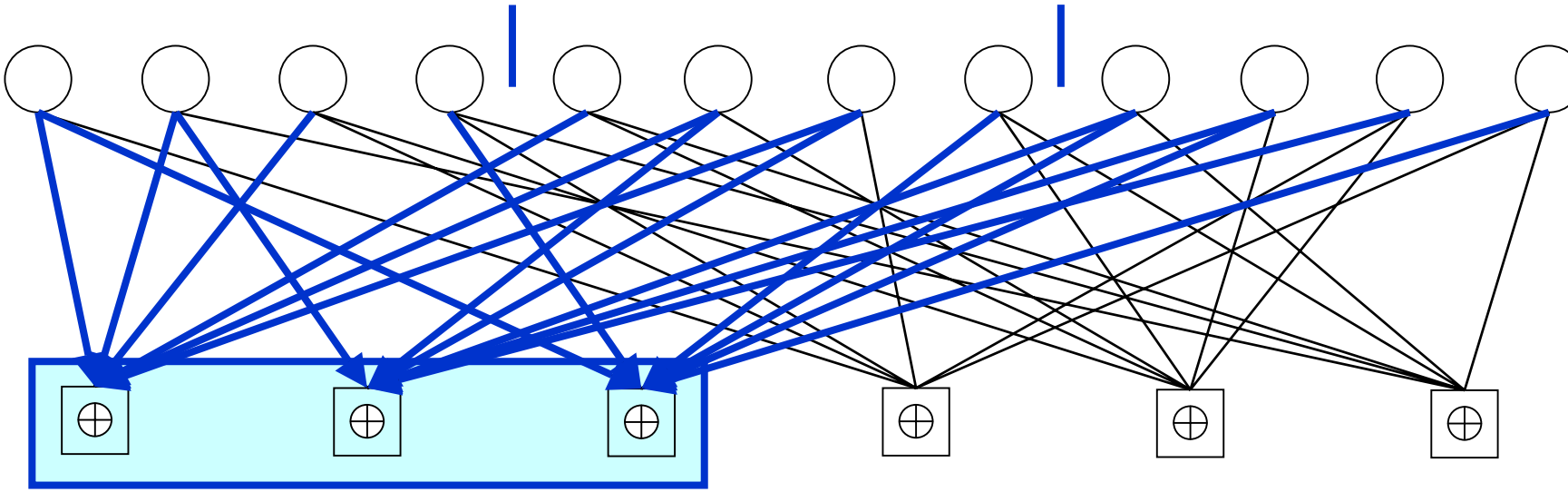
Objectifs

- Validation Algorithme Architecture du λ-Min
- Accélération matérielle
  - Implantation sur plateforme d'évaluation FPGA
    (Xilinx Virtex 1000 E)
- Souplesse et réutilisation
  - Généricité du code et de l'architecture

Séquencement série

- P processeurs simultanément
- Utilisation de bancs mémoires (Boutillon et. al.)
- Accès série

- Exemple : P = 3
  - Lecture en mémoire
  - Permutation

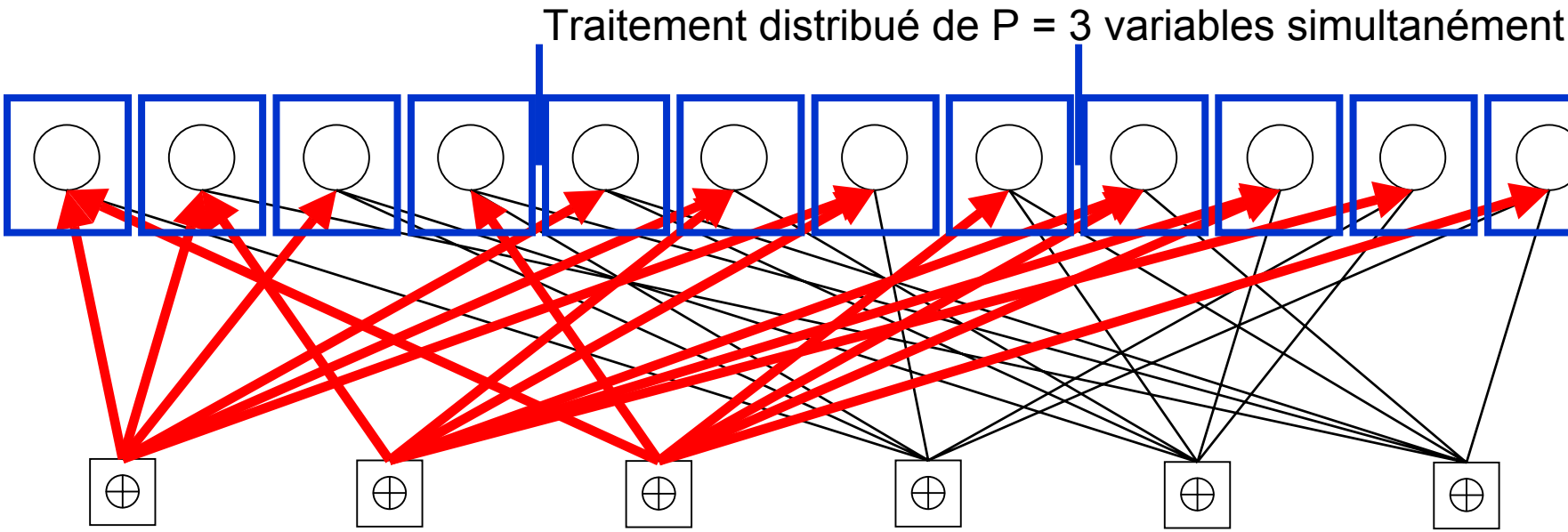


Traitement compact de P = 3 parités simultanément

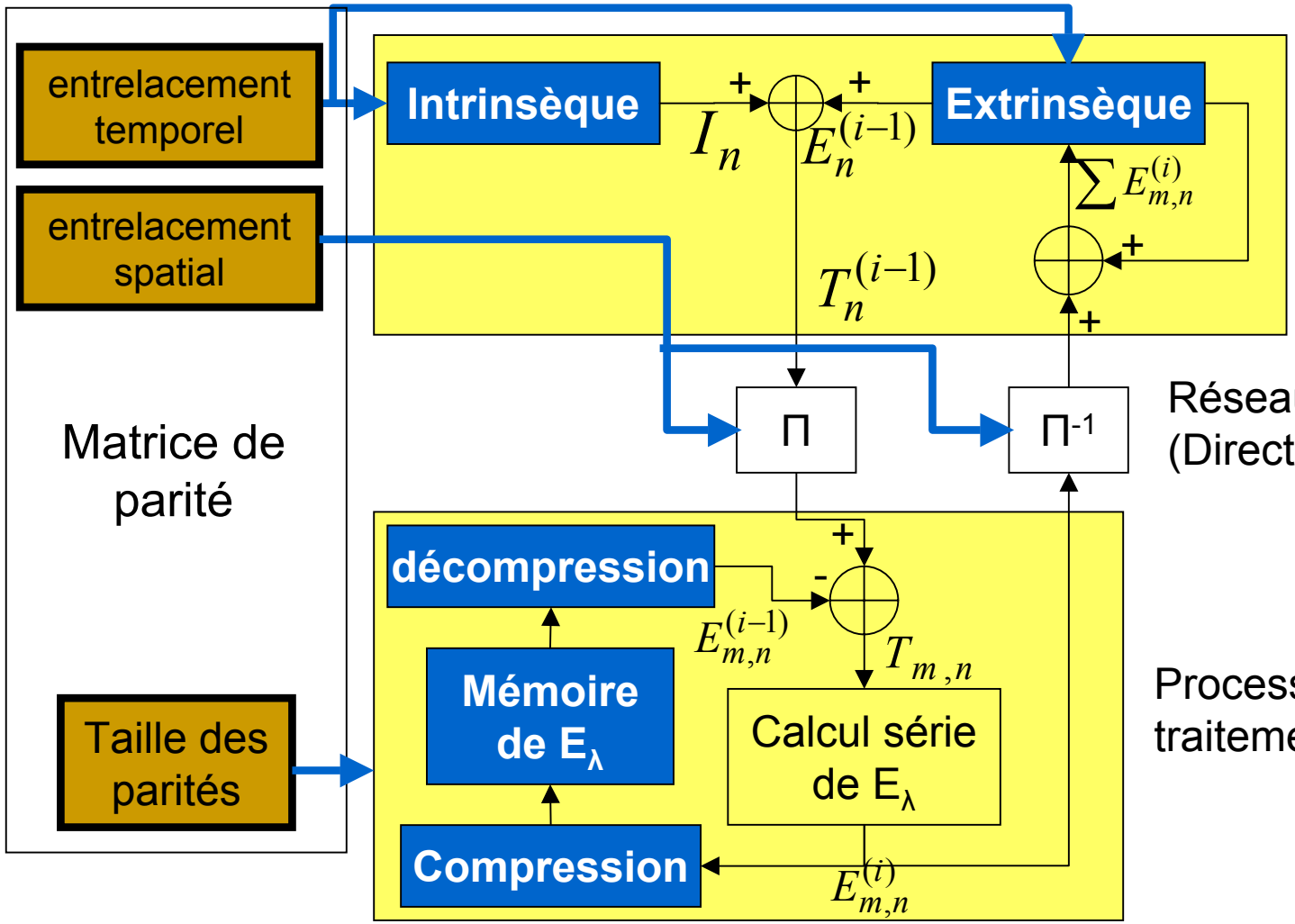
Séquencement série

- P processeurs simultanément
- Utilisation de bancs mémoires (Boutillon et. al.)
- Accès série

- Exemple : P = 3
  - Lecture en mémoire
  - Permutation

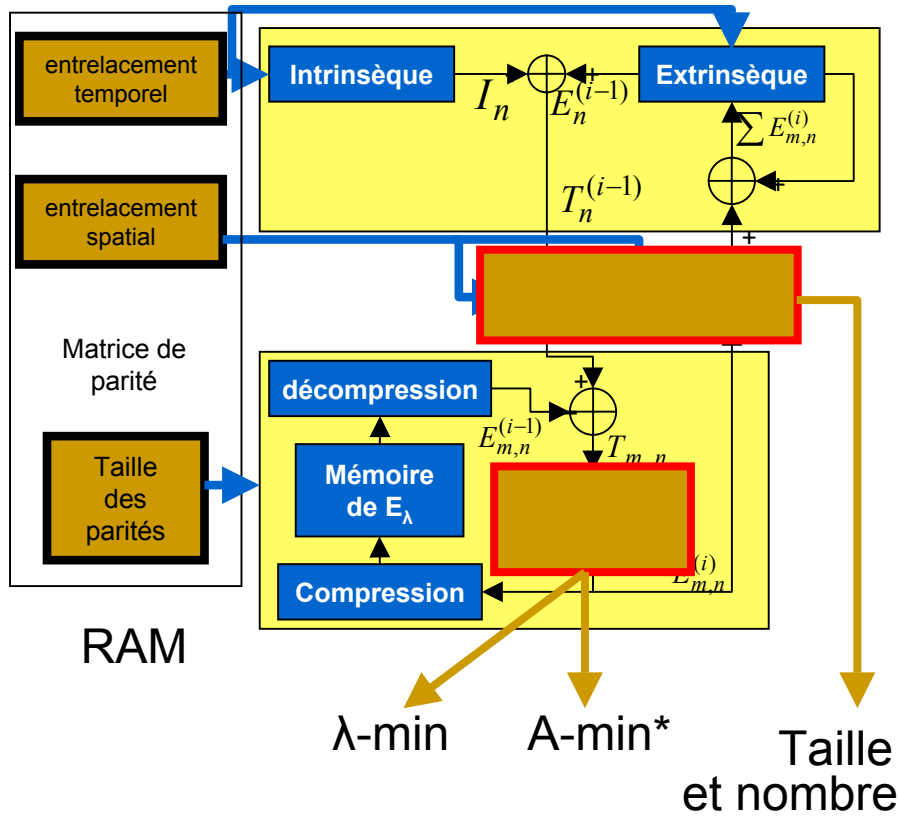


• Architecture générique (1)



Architecture générique (2)

- Généricité statique (avant synthèse) :
  - Description de l'architecture paramétrée
    - Latences, tailles des bus
  - calcul des parités
  - Mémoires
  - réseaux d'interconnexions
- Généricité dynamique (après synthèse) :
  - RAM pour les entrelacements et les tailles des parités
  - Nombre d'itération max.



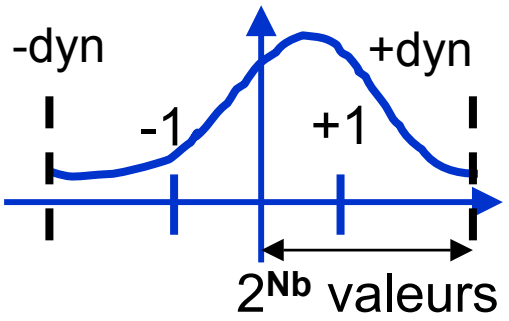
Résultats

- Ecriture en VHDL puis synthèse de l'architecture

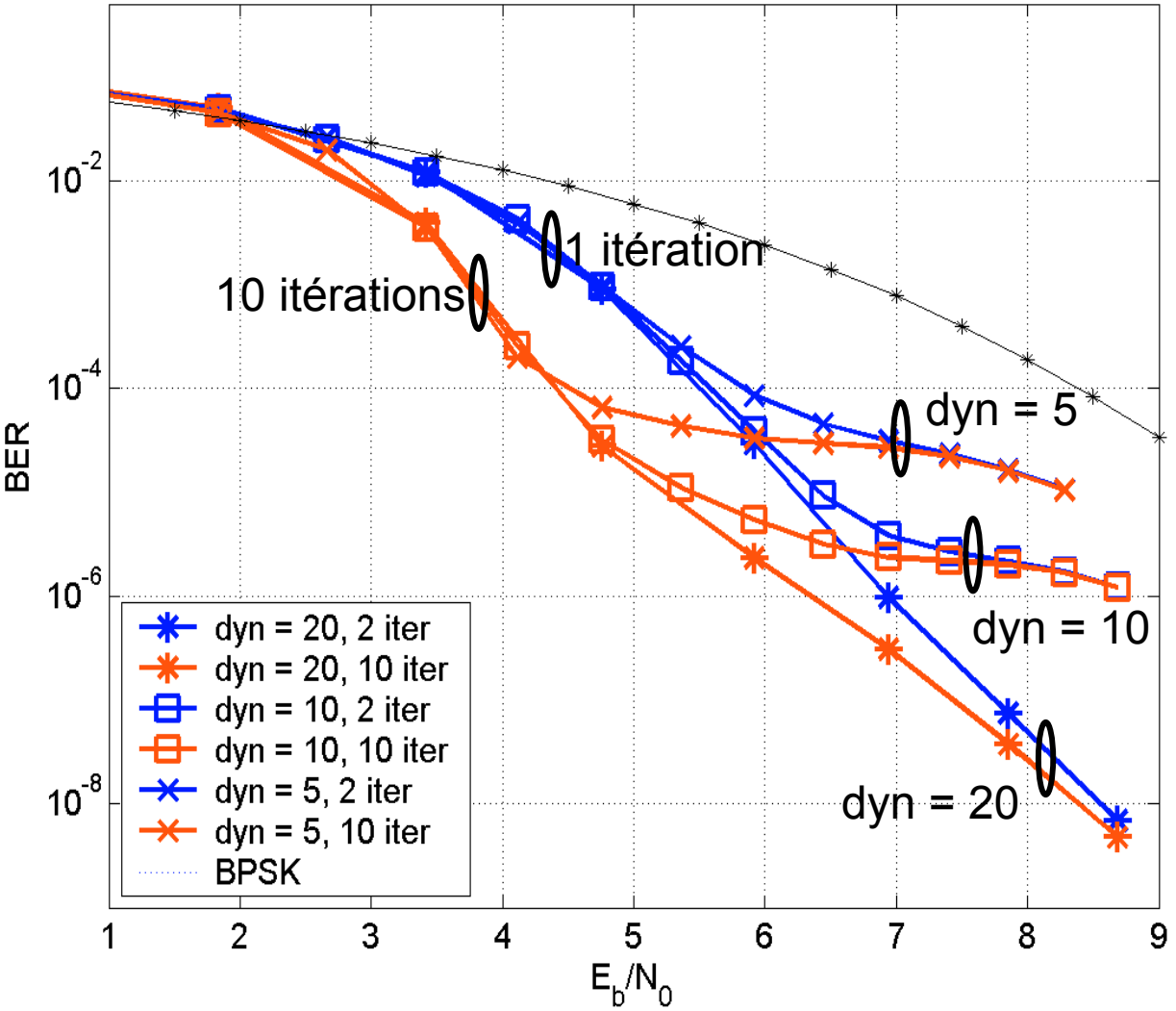
- Exemple : (N=2000, R=0.8)
  - Débits (25 MHz) :
    - Hardware : 5 Mb/s/iteration
    - Software : 15kb/s/iteration

    Gain : 300
  - Synthèse
    - 100 % des blocs mémoires
    - 25 % des cellules FPGA

Influence de la dynamique

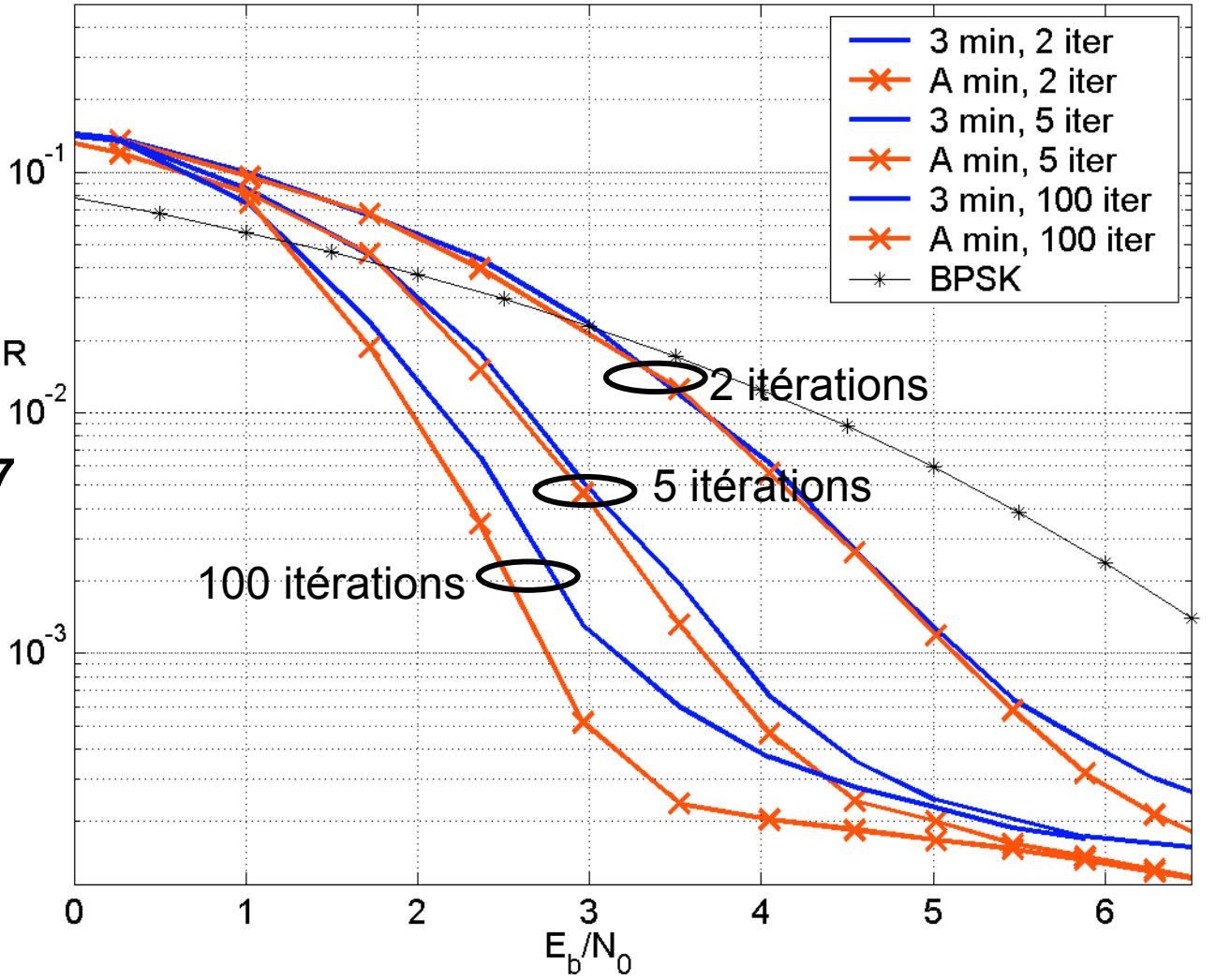


- N = 2000
- R = 0,8
- Parité de degré 6, 18 et 51
- P = 5
- Nb = 6
- 1000 erreurs



Comparaison 3-min / A-min*

- N=400, M=200
- P=5
- Parités de degré 5 et 7
- dyn = 6
- Nb = 6



Conclusion

- Accélération
- Vaste classe de code décodable
- Grande souplesse de l'architecture
- Outil d'étude
  - Permutation
  - Format du codage
  - Plancher d'erreur
  - Comparaison d'algorithme sur un grand nombre de codes

Plan

- Introduction
- Les codes LDPC
- L'algorithme λ-Min
- Architecture Générique
- Formalisation
- Conclusion et Perspectives

Introduction

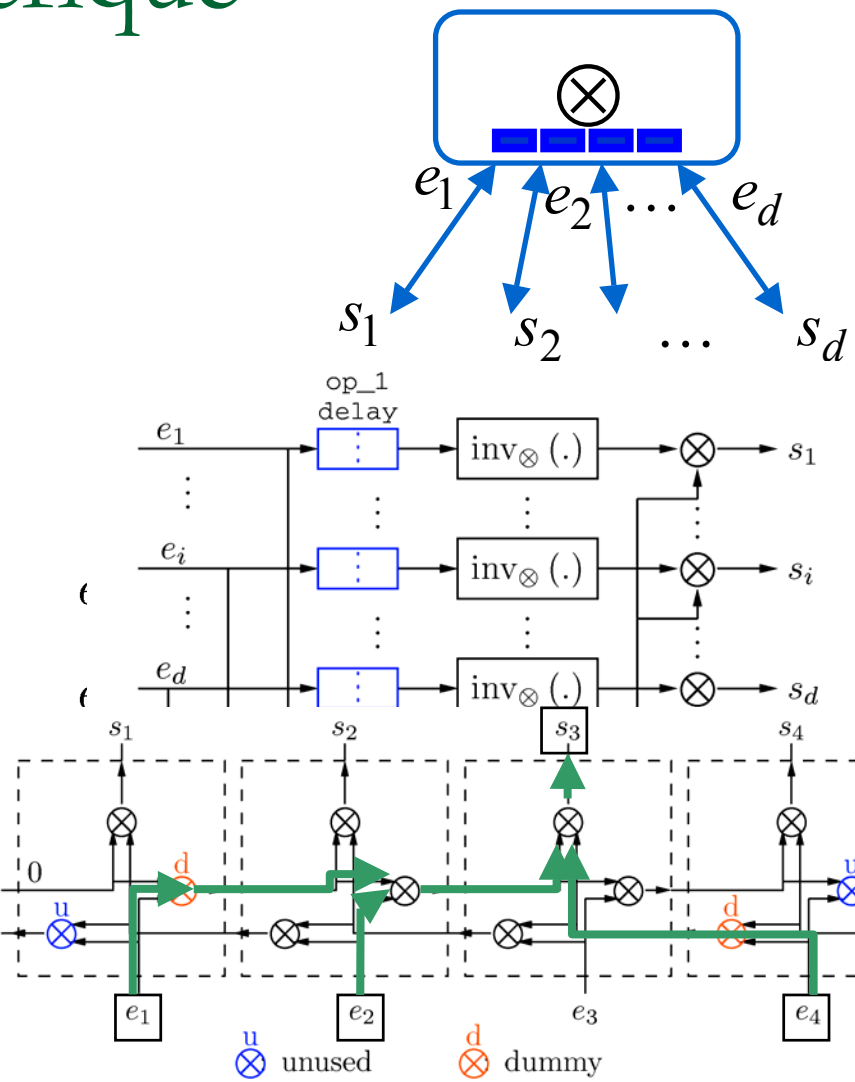
- Constat :
  - Multiplication des solutions pour décodeurs LDPC
  - Comparaisons difficiles
- Proposition
  - Un cadre formel d'analyse et de synthèse d'architecture de décodeurs LDPC
    - Processeur de nœud générique (parité, variable)
    - Architecture générique de propagation de messages
  - Application à l'état de l'art

Processeur générique

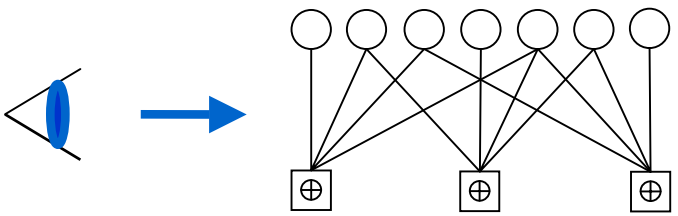
- Un opérateur associatif et commutatif $\otimes$
- d entrées e_i / sorties s_i
- mémoire
- Relation e/s

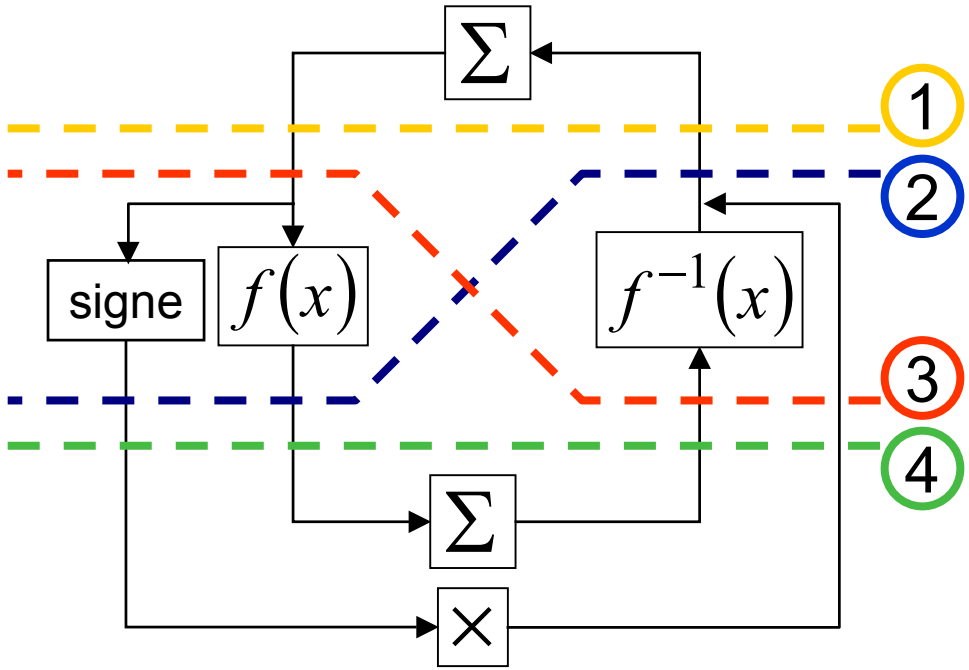
$$s_j = \bigotimes_{i \neq j} \{e_i\}$$

- Implantations possibles :
  - Directe
  - Somme totale (si opérateur inverse)
  - Treillis



• Quel Opérateur ?

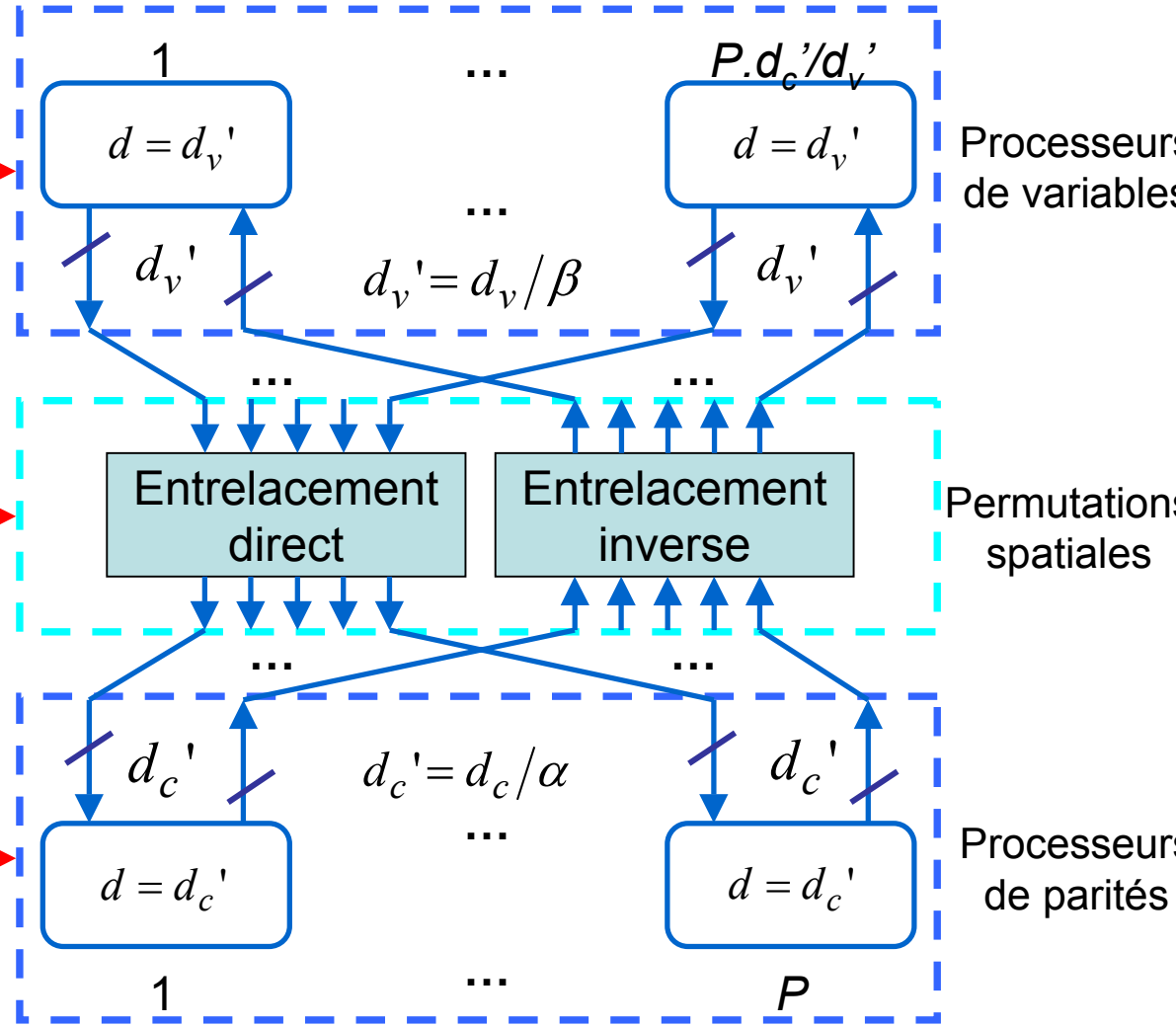




Position du réseau		1	2	3	4
Opérateur générique ⊗	Processeur de variable	Σ	$f \circ \Sigma$	$\Sigma \circ f^{-1}$	$f \circ \Sigma \circ f^{-1}$
	Processeur de parité	$f^{-1} \circ \Sigma \circ f$	$f^{-1} \circ \Sigma$	$\Sigma \circ f$	Σ
		× (produit des signes)			

Propagation des messages

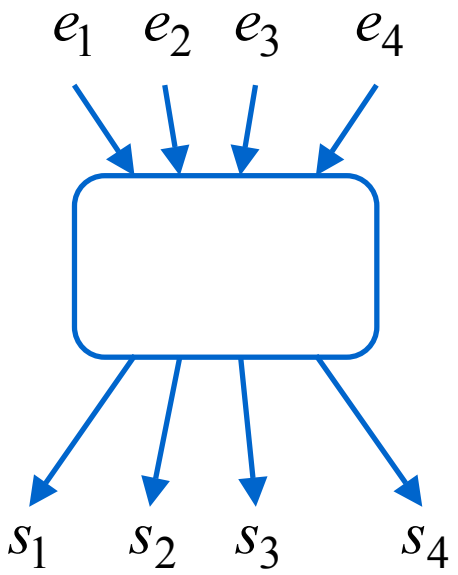
- P processeurs de parités
  (α cycles)
- Processeurs de variables
  (β cycles)
- Parallèle :
  P=M, α=1, β=1
- Série :
  P=1, α=d_c, β=d_v



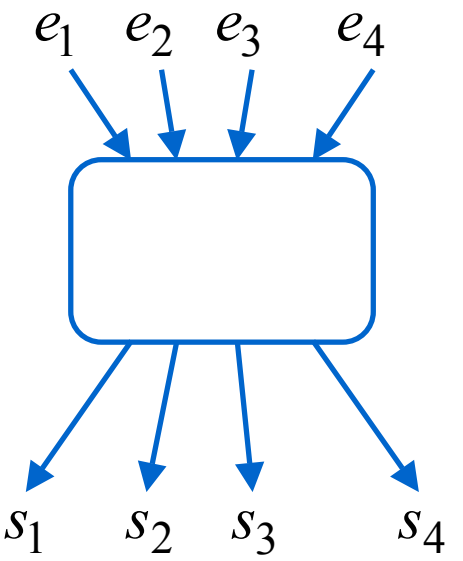
Contrôle

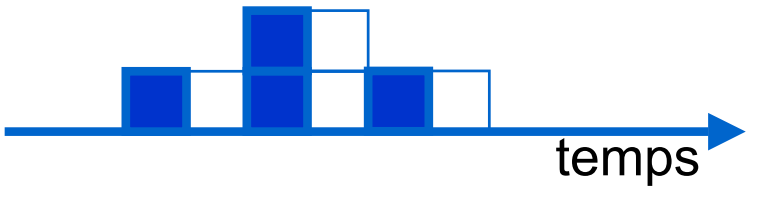
Quel Contrôle ?

- Flot d'entrées / sorties compact



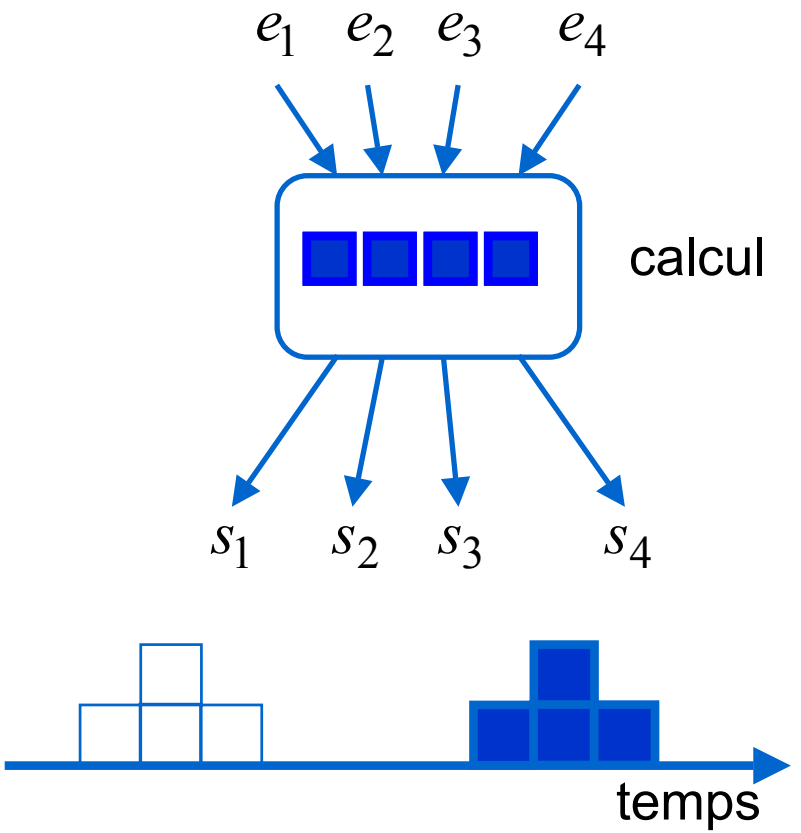
- Flot d'entrées / sorties distribué

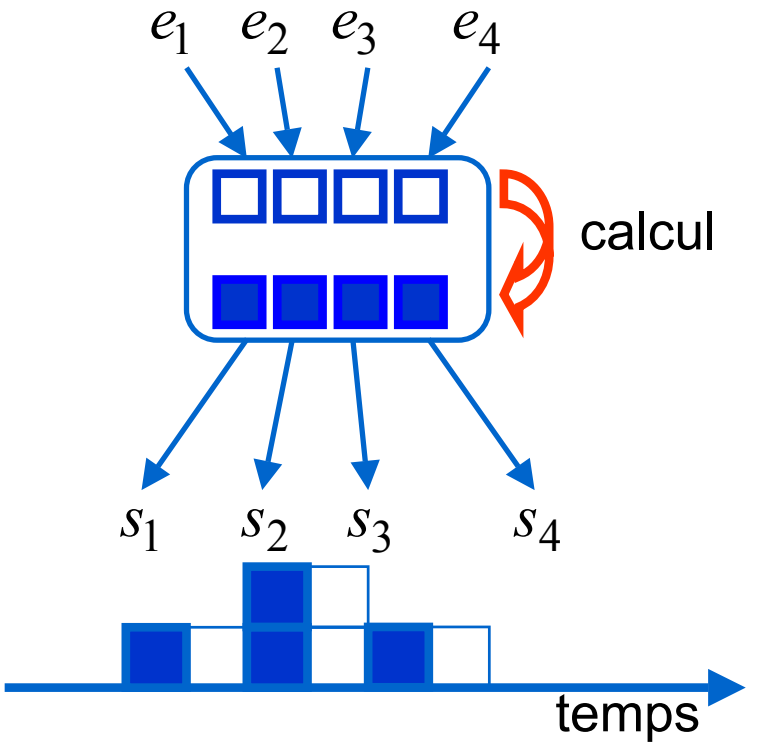




Quelle taille mémoire ?

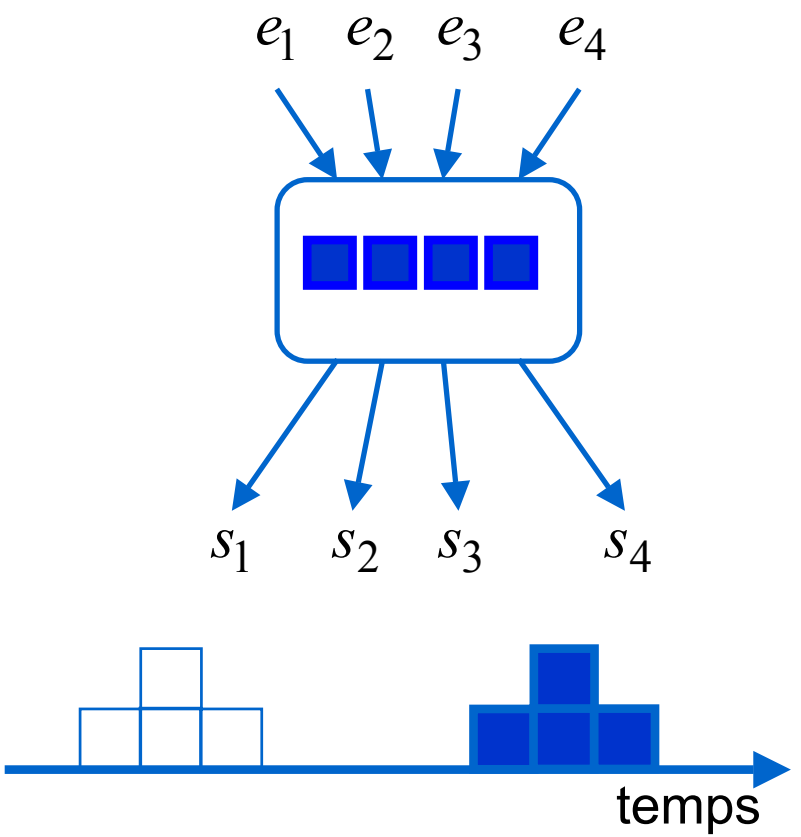
- Flot Compact *d*
- Flot Distribué
  - Mise à jour différée *2d*



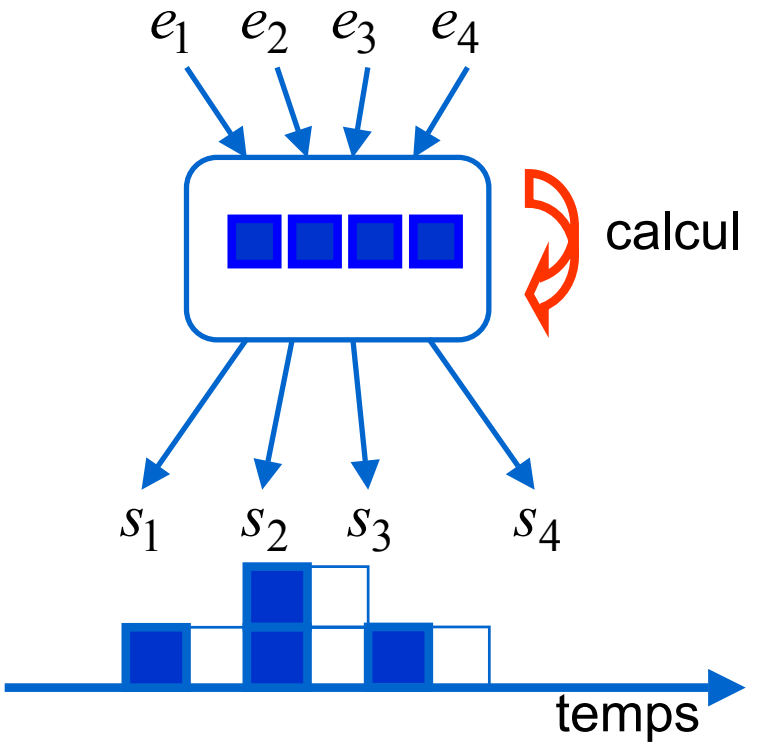


Quelle taille mémoire ?

- Flot Compact : *d* bancs



- Flot Distribué
  - Mise à jour différée *2d*
  - Mise à jour immédiate *d*



Résumé

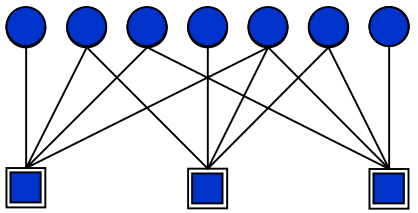
Paramètres :

- Processeurs de nœuds:
  - Architecture (directe, trellis, somme totale)
  - Position du réseau d'interconnexion (1,2,3,4)
- Parallélisme : P, α, β
- Contrôle des processeurs

Toutes les combinaisons sont possibles

• Combinaison des Contrôles

Entrelacement (vertical)



			VARIABLES: Compacte	VARIABLES: Distribué – Mise à jour différée	VARIABLES: Distribué – Mise à jour immédiate
PARITES: Compacte			**Inondation (d_c memoires)**	**Inondation parités ($2d_c$ mémoires)**	**Entrelacement horizontal (d_c mémoires)**
PARITES: Distribué	Mise à jour différée		**Inondation variables ($2d_c$ mémoires)**	Contrôle des branches	
PARITES: Distribué	Mise à jour immédiate		**Entrelacement vertical (d_c mémoires)**		

• Etat de l'art

			VARIABLES: Compacte	VARIABLES: Distribué – Mise à jour différée	VARIABLES: Distribué – Mise à jour immédiate
PARITES	Compacte		**Blanksby (parallèle)**	**Chen, Zhang, Générique**	**Mansour, Brevet**
PARITES	Distribué	Mise à jour différée	**Nouveau**	-	
PARITES	Distribué	Mise à jour immédiate	**Nouveau (algo Fossorier)**		

• Etat de l'art : taille mémoire

E : Nombre de branches

F( ) : compression

PARITES			VARIABLES: Compacte	VARIABLES: Distribué – Mise à jour différée	VARIABLES: Distribué – Mise à jour immédiate
Compacte			N + E	3N+ f(E)	N+ f(E)
Distribué	Mise à jour différée		N + 2M + E	-	
	Mise à jour immédiate		N + M + E		

Conclusion

- Proposition d'une étude formelle des architectures de décodeurs LDPC
  - Processeur de nœud
    - Architecture
    - Réseau d'interconnexion
  - Parallélisme
  - Contrôle (séquencement)
- Classement de l'état de l'art
- Proposition d'une nouvelle architecture

Plan

- Introduction
- Les codes LDPC
- L'algorithme λ-Min
- Architecture Générique
- Formalisation
- Conclusion et Perspectives

Conclusion et perspectives

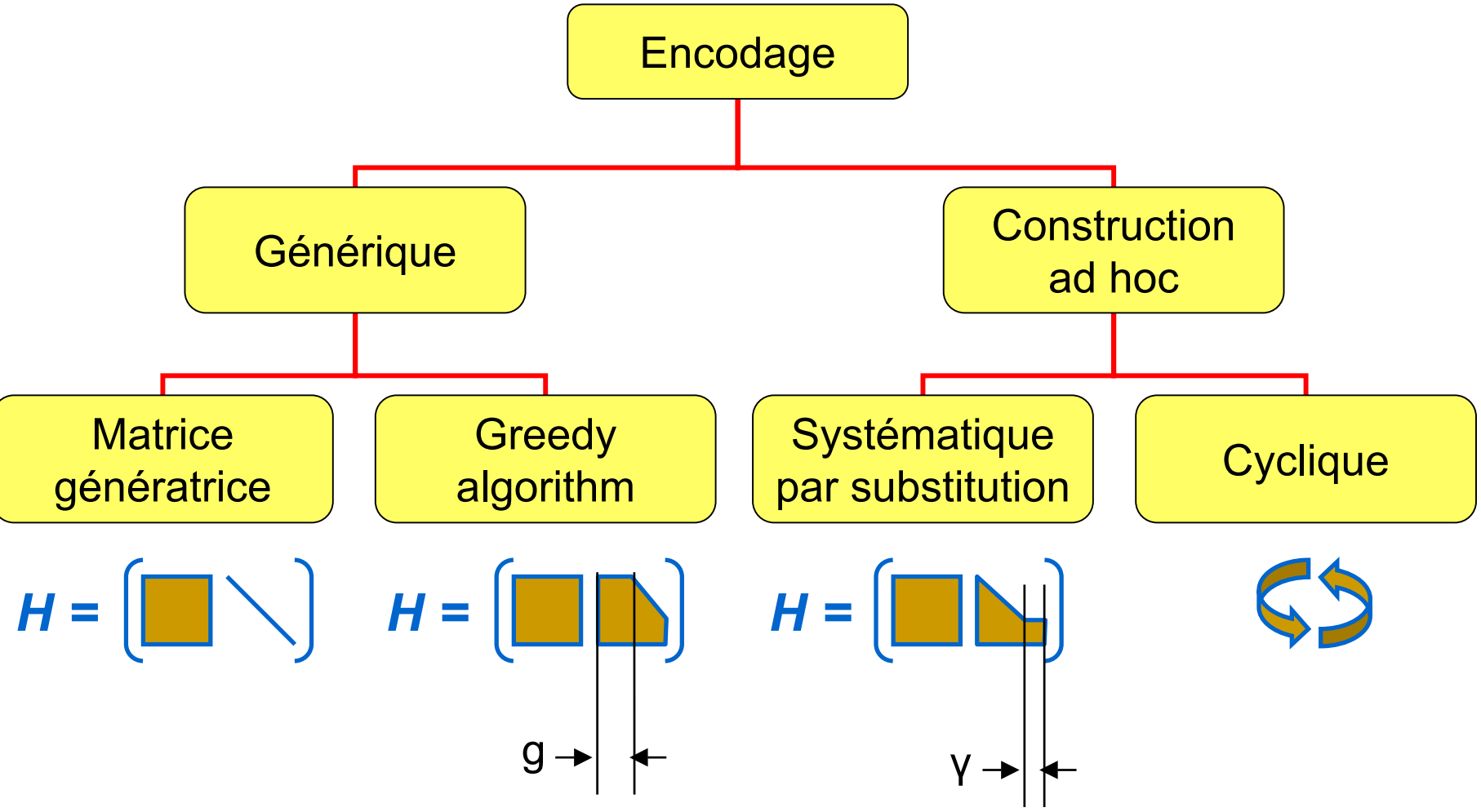
- Algorithmes sous optimaux performants
- Architecture Générique de décodeur LDPC
- Formalisme de description des architectures de décodeurs LDPC

- Évolution de la plateforme
  - Nouveaux Séquencements, Permutations
- Optimisation des codes tailles finies
- Extension aux corps GF(2^q)

Fin de l'exposé

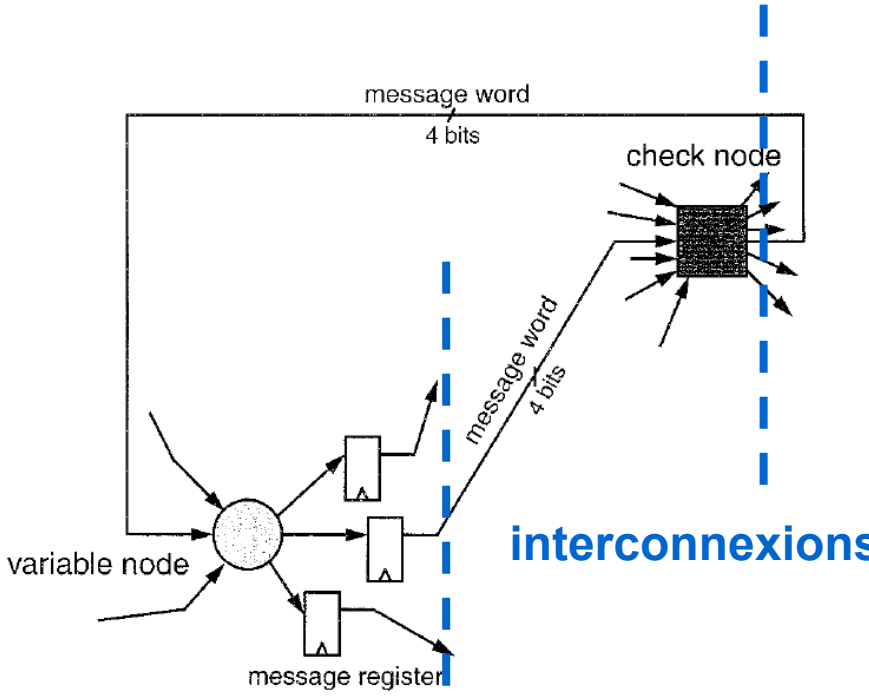
Merci !

Code LDPC : encodage



Architecture parallèle

- Utilisation du parallélisme du graphe
- Avantages
  - performance: 1Gb/s 64 iterations
  - Dissipation : 690mW
  - TEP=2.10^{-4} @ 2,5 dB
- Inconvénients
  - Routage Complexe
    - → ad hoc outils CAD
  - Chip : 52.5mm^2 , 0.16µ
  - Code figé



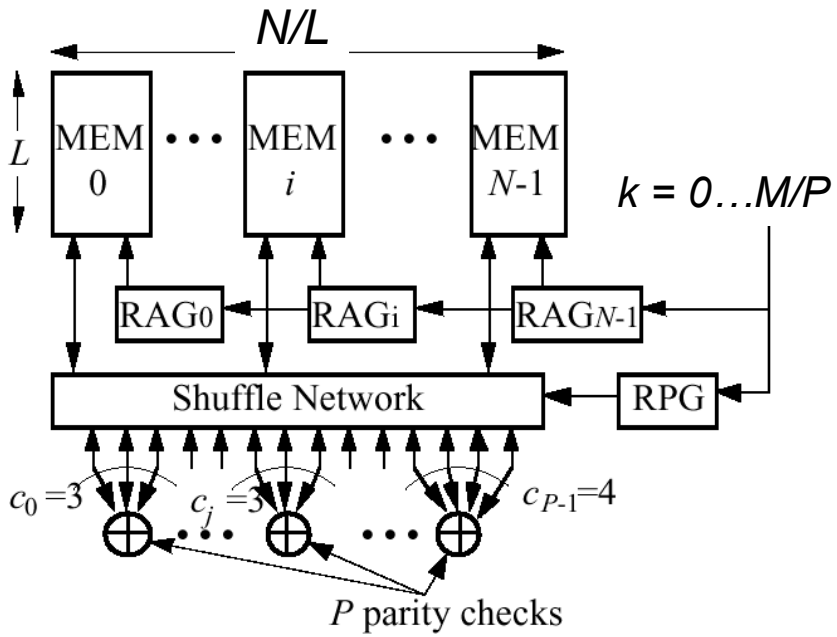
Blanksby, Howland

« A 690-mW 1-Gb/s 1024-b Rate-1/2 Low-Density Parity-Check Code Decoder »

(IEEE Trans. on Solid-State Circuits, 2002.)

Architecture mixte

- Traitements de P<M parités en parallèle
- Une itération en en M/P cycles
- Un cycle :
  - Lecture en mémoire
  - Réseau d'interconnexion
  - Parité
  - Réseau inverse
  - Ecriture en mémoire
- Entrelacement temporel et spatial



Boutillon, Castura , Kshishang « *Decoder-First Code Design* » Proc. of the 2nd Int. Symp. on Turbo Codes and Related Topics, Brest, 2000.

Architecture