MajecSTIC 2009
Avignon, France, du 16 au 18 novembre 2009

Algorithme de décodage généralisé par propagation de croyances

Jean-Christophe Sibel et Sylvain Reynal.

Laboratoire ETIS UMR 8051-CNRS, ENSEA - Université de Cergy-Pontoise, 6 avenue du Ponceau, 95014 Cergy-Pontoise Cedex - France.

Contact : jean-christophe.sibel@ensea.fr, reynal@ensea.fr

Résumé
Les communications numériques reposent sur un trio de transmission : émetteur, canal, récepteur. Le canal est l'objet de perturbations physiques qui induisent des erreurs dans le message transmis. Afin de se prémunir contre ces erreurs, on utilise un code correcteur d'erreurs qui consiste au niveau de l'émetteur à ajouter de la redondance au message. Au niveau du récepteur, l'opération de décodage permet alors de récuperer l'information utile. Un des algorithmes de décodage les plus répandus est l'algorithme de propagation de croyances, de type itératif, réputé sous-optimal pour une certaine catégorie de codes, en raison de l'existence de cycles dans le graphe de ces codes. Nous présentons dans cet article une généralisation de la propagation de croyances qui permet de rendre le décodage optimal y compris en présence de cycles, offrant ainsi une meilleure récupération de l'information. Nous décrivons la construction de l'algorithme par analogie avec une technique de physique statistique appliquée à un réseau de spins, puis nous en faisons le lien avec les représentations graphiques des codes LDPC.

Abstract
Digital communications are composed of a transmitter, a channel and a receiver. Errors in the transmitted message come from physical disruptions in the channel. The use of error correcting codes enables to protect information from these errors, it consists in the addition of redundancy in the message before its entrance in the channel. At the receiver, information is got back thanks to a decoding. One of the most widespread decoding algorithms is the Belief Propagation, known for its suboptimal property concerning a particular class of codes whose graphs contain cycles. This article deals with an extension of this algorithm which could make the decoding optimal whatever the cycles would be, so as to get more reliable information. The construction of such an algorithm is based on a technique which comes from statistical mechanics in a spins lattice, that is introduced with an application on LDPC codes.

Mots-clés : décodage, codes LDPC, algorithme itératif, cycles, énergie.

Keywords: decoding, LDPC codes, iterative algorithm, cycles, energy.

Introduction

La théorie de Shannon permet aujourd'hui de construire des protections robustes, les codes, contre le bruit du canal dans la transmission de bits d'information. Ces codes sont assez sophistiqués pour exiger un décodage complexe en vue de récupérer l'information. Les algorithmes itératifs modernes de décodage utilisent la représentation des codes en graphe, le plus souvent ceux de Tanner, dont les branches sont le support de transmission de l'information sous forme de messages et dont les noeuds sont les noeuds de variables et noeuds de parité du code. L'idée principale est d'itérer suffisamment de fois la propagation des messages pour faire converger le graphe vers un point fixe qui est l'information utile transmise. Cependant, cette représentation a l'inconvénient majeur de faire apparaître des structures topologiques réduisant l'efficacité du décodage. L'algorithme de propagation de croyances généralisé vise à absorber ces structures dans

le but de récupérer une information plus fidèle. Nous étudions dans un premier temps le fonctionnement des codes et plus particulièrement des codes LDPC, puis nous appliquons l'algorithme de propagation de croyances sur de tels codes afin d'analyser ses performances et de mettre en lumière ses problèmes topologiques. S'ensuit une étude sur la solution proposée, la généralisation de la propagation de croyances.

1. Les codes LDPC

Selon [1], pour protéger l'information dans la transmission, l'idée principale est de plonger l'espace informatif dans un espace de plus grande dimension, dit espace du code. L'information, représentée en bits, devient ainsi redondante telle qu'un mot de taille k bits devient un mot de taille N bits où les $N-k$ bits supplémentaires sont les bits de redondance. La redondance consiste à créer des corrlélations type opérations logiques entre les bits d'information. On représente cette redondance par $N-k$ équations de parité agissant sur les N bits de la transmission, regroupées dans une matrice H de vérification de parité de taille $N \times (N-k)$. Un mot de taille N est un mot de code si et seulement s'il appartient au noyau de H.
Un code LDPC a la particularité d'avoir une matrice H de faible densité (LDPC $\equiv$ Low Density Parity Check), autrement dit, le nombre de 1 est très inférieur au nombre de 0, propriété très importante pour le décodage (voir section 3). Voici un exemple simple de code LDPC dit régulier - le nombre d_c de 1 par ligne et le nombre d_v de 1 par colonne étant fixes - noté ($N = 6, d_v = 2$, $d_c = 3$) :

$$\mathcal{H} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \text{ de densité } \rho = 0.5$$

2. Graphe de Tanner

Les algorithmes de décodage reposent sur une représentation en graphe du code, dit graphe de Tanner, incluant les deux structures essentielles : N bits représentés par N noeuds de variable, M équations de parité représentés par M noeuds de parité. Ces deux types de noeuds sont reliés par des branches selon la règle suivante : une branche joint le noeud de parité E_i et le noeud de variable x_j si et seulement si $H_{ij} = 1$ (voir figure 1).

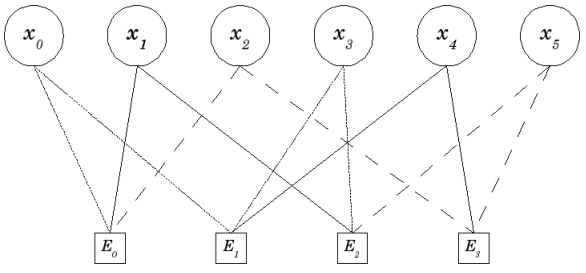
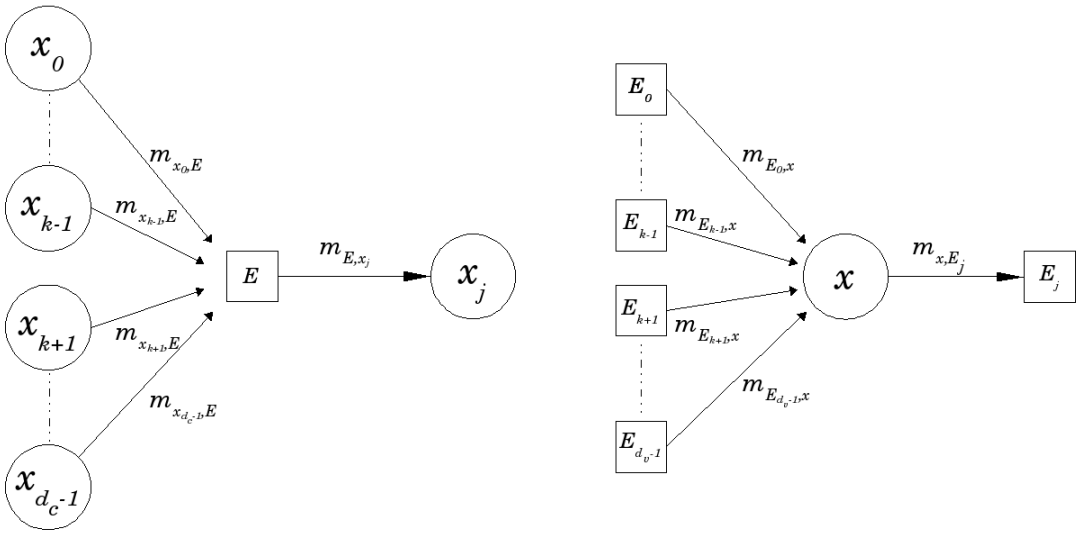


FIG. 1 – Graphe de Tanner du code LDPC ($N = 6, d_v = 2, d_c = 3$)

3. Décodage par propagation de croyances

La branche partant d'un noeud quelconque i et arrivant sur un noeud quelconque j est un message transportant une probabilité sur le noeud j conditionnellement au noeud i et ses voisins. Ces messages sont interprétés comme des propagations d'informations de noeud en noeud, où ces informations appelées croyances, traduisent la pensée majoritaire sur un noeud selon tous ses

voisins, d'après [2] (voir figure 2). La propagation de croyances consite à calculer ces messages selon des mises à jour particulières.



(a) Noeud de parité → noeud de variable

(b) Noeud de variable → noeud de parité

FIG. 2 – Mise à jour des messages

Le décodeur fonctionne de manière itérative, c'est-à-dire que les messages calculés dans l'algorithme à l'itération (i) dépendent directement de ces mêmes messages à l'itération précédente $(i-1)$, selon les équations ci-dessous :

$$m^{(i)}_{x\to E_j} = f_1(m^{(i-1)}_{E_k\to x}), k \neq j$$
$$m^{(i)}_{E\to x_j} = f_2(m^{(i)}_{x_k\to E}), k \neq j$$

D'après [3], ces équations se définissent comme suit :

$$m_{x\to E_j} \propto \prod_{k\neq j}^{d_v-1} m_{E_k\to x} \tag{1}$$

$$m_{E\to x_j} \propto \sum_{x_0} \dots \sum_{x_{j-1}} \sum_{x_{j+1}} \dots \sum_{x_{d_c-1}} \left(\prod_{k\neq j}^{d_c-1} m_{x_k\to E}\right)\left(\sum_{k=0}^{d_c-1} x_k\right) \tag{2}$$

Remarque : Les sommes sont faites modulo 2 car on travaille dans le corps de Gallois $GF(2)$.

Le but du décodage est, rappelons-le, de retrouver les valeurs des bits ou noeuds de variable en sortie du canal de transmission. Au fur et à mesure des propagations dans le graphe, les valeurs des noeuds de variable convergent vers un point fixe qui est le mot de code. Dans le cas idéal (canal non perturbatif), le mot en sortie de l'algorithme est exactement un mot de code correspondant au mot en entrée du canal, mais en pratique, le canal peut fortement perturber le signal entrant, et l'algorithme peut ne pas converger vers un mot de code : le taux d'erreurs binaire (rapport du nombre de bits en erreur sur la taille du mot) dépend directement du rapport signal à bruit (voir figure 3).
Il suffirait alors de travailler à de forts rapports signal à bruit pour obtenir de bons décodages, mais cette solution n'est pas optimale. Premièrement, un fort rapport signal à bruit implique un

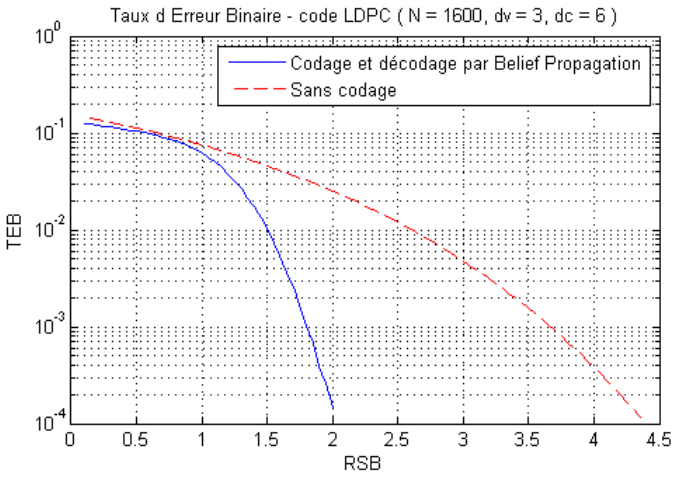


FIG. 3 – Taux d'erreur binaire du code LDPC ($N = 1600, d_v = 3, d_c = 6$) décodé par propagation de croyances

coût matériel élevé, deuxièmement, il existe d'autres causes de dégradations des performances de décodage bien plus difficiles à contrer (voir section 4).

Remarque : d'un point de vue algorithmique, d'après [1], la propagation de croyances traverse toutes les branches un même nombre de fois, ce qui permet d'affirmer que le nombre d'opérations est linéaire en le nombre de noeuds. Ainsi, plus le code est creux (à faible densité), plus le nombre de branches diminue ce qui rend le décodage plus rapide.

4. Cycles et trapping sets

Dans l'exemple de la section 2, le graphe présente des structures topologiques appelées cycles.

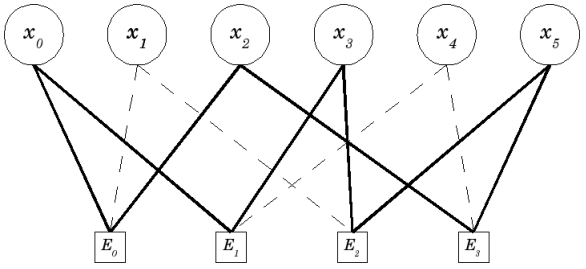


FIG. 4 – Cycles dans le code LDPC ($N = 6, d_v = 2, d_c = 3$)

Ces cycles sont tels que les messages bouclent sur eux-mêmes. Dans un premier cas de figure, ce bouclage peut faire osciller des croyances entre plusieurs valeurs sans jamais les faire converger vers un point fixe. Par conséquent, le mot de sortie a une très forte probabilité de ne pas être un mot de code, quelque soit le nombre d'itérations de l'algorithme. Dans un second cas de figure, ces croyances convergent vers des valeurs fausses donnant lieu à des faux mots de codes, dits pseudo-mots de code, dégradant également les performances du décodage (voir figure 5). Ces deux cas de figure indiquent que l'information est finalement piégée dans le cycle, on parle alors de Trapping Set. Le taux d'erreurs binaire devient ainsi trop élevé pour des valeurs de rapport si-

gnal à bruit encore trop importantes. Pour contrer ces effets, l'idée est de passer par une approche de la physique statistique.

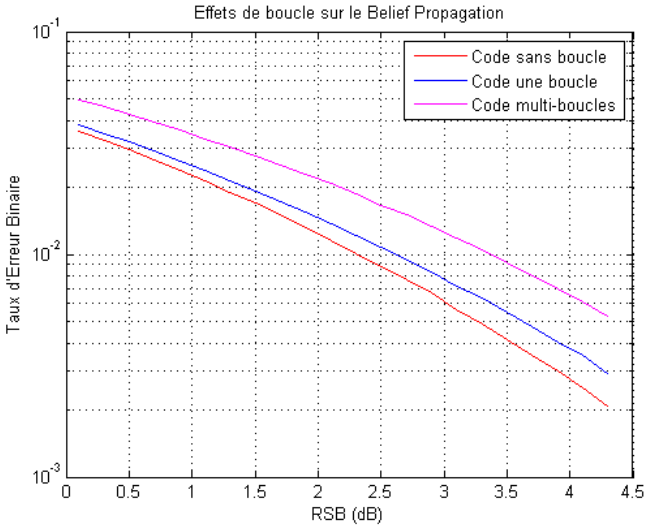


FIG. 5 – Effets de boucle sur le Belief Propagation

5. Physique statistique et codage

On peut modéliser des mots de code par un réseau de spins plongé dans un champ magnétique uniforme, où chaque spin représente un noeud de variable du graphe, les noeuds de parité eux ne sont pas représentés car physiquement, ce sont des interactions. L'état de chaque spin prend ses valeurs dans $\{-1, +1\}$, correspondant simplement à une modulation BPSK [1], et les états de tous les spins du réseau déterminent l'énergie globale E du réseau. En effet, selon [4], l'énergie du réseau se décompose linéairement selon chaque spin i avec deux types d'énergies : l'énergie E_i échangée entre le spin i et le champ magnétique, l'énergie E_{ij} échangée entre le spin i et le spin j du réseau.

$$E = \sum_i E_i + \sum_i \sum_j E_{ij} \tag{3}$$

D'après [5], on peut déduire de cette équation les mises à jour des messages notées à la section 3. Cependant, ce modèle, appelé approximation de Bethe, n'est pas rigoureusement exact. En effet, une hypothèse de ce modèle est de ne pas considérer un voisinage indirect : dans le cadre des équations expliquées en figure 2, les voisins de l'émetteur sont supposés indépendants. Si le graphe présente des cycles, on obtient donc un modèle approché, comme le montrent les courbes de la figure 5. Pour éviter cet inconvénient, on utilise une approximation plus géneérique, l'approximation de Kikuchi. Il s'agit de regrouper les noeuds de variables par les interactions de parité qui les lient : c'est une manière d'étendre l'écriture de l'énergie d'un réseau avec des énergies d'interaction mettant en jeu un nombre arbitraire de spins.

6. Le graphe des régions

Ce graphe regroupe les noeuds de variable en interaction, c'est-à-dire les noeuds de variables appartenant à une même équation de parité. Chaque équation de parité du code constitue ainsi ce

[1] Binary Phase Shift Keying

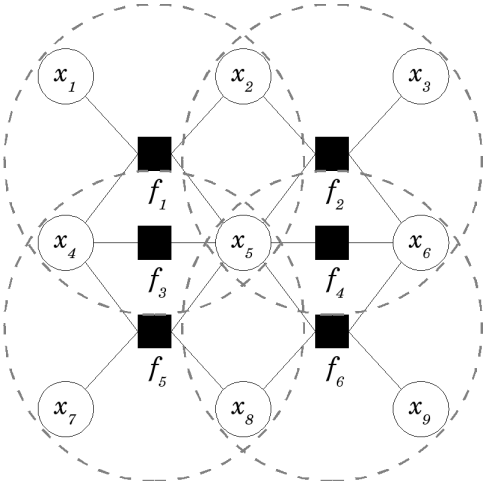


FIG. 6 – Approximation de Kikuchi

qu'on appelle une *région globale* ou *cluster global*. Le but est de remplacer les échanges de message entre noeuds isolés dans le graphe de Tanner par des échanges entre régions, dans lesquelles certains cycles seraient *absorbés*. Ces échanges sont effectués grâce aux intersections entre ces régions gloables, ces intersections formant de nouvelles régions. La construction du graphe des régions se poursuit en recherchant les intersections entre les régions calculées en dernier, ceci jusqu'à obtenir des régions élémentaires formées donc d'un seul noeud de variable. Une région R_G dont les noeuds de variables sont l'ensemble $V_G = \{v_1, ..., v_j | j = card(V_G)\}$ contient toutes les régions ascendantes dont les noeuds de variables sont des sous-ensembles de V_G. La figure 7 (a) montre un exemple de graphe des régions pour le code $(N = 4, d_v = 3, d_c = 3)$ dont la matrice est

$$\mathcal{H} = \begin{pmatrix} \boxed{1} & \boxed{1} & 1 & 0 \\ \boxed{1} & \boxed{1} & 0 & 1 \\ 1 & 0 & \boxed{1} & \boxed{1} \\ 0 & 1 & \boxed{1} & \boxed{1} \end{pmatrix}$$

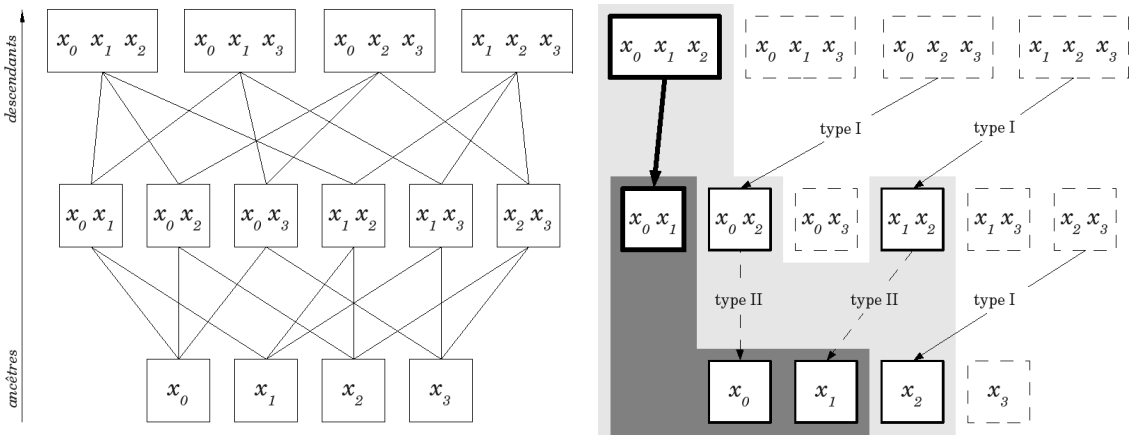
Les éléments de H encadrés représentent deux intersections de taille 2 :
– $\{x_0, x_1\}$ pour les lignes 0 et 1 ;
– $\{x_2, x_3\}$ pour les lignes 2 et 3.

Il existe au total, pour ce code, 6 intersections de taille 2 dont celles citées ci-dessus et les suivantes :
– $\{x_0, x_2\}$ pour les lignes 0 et 2 ;
– $\{x_0, x_3\}$ pour les lignes 1 et 2 ;
– $\{x_1, x_2\}$ pour les lignes 0 et 3 ;
– $\{x_1, x_3\}$ pour les lignes 1 et 3.

Remarque : ce code n'est pas exploitable en pratique en raison de sa très forte densité, mais le graphe des régions associé est complet, ce qui permet une bonne compréhension.

Algorithme - Message

La mise à jour des messages est calculée d'après [5], que l'on peut interpréter comme un algorithme de parcours du graphe des régions. Le calcul est tel qu'on ne considère plus de messages bidirectionnels comme dans la propagation de croyances simple (voir section 3) mais des messages unidirectionnels se propageant à l'intérieur des régions dans le sens ascendant (des enfants vers les parents). Pour le message allant d'une région enfant R_E ($\{x_0, x_1, x_2\}$ sur la figure 7 (b)) vers une région parent R_P ($\{x_0, x_1\}$ sur la figure), le principe de l'algorithme est de considérer tous les



(a) Graphe des régions

(b) Mise à jour du message $m_{\{x_0,x_1,x_2\}\to\{x_0,x_1\}}$ dans le graphe des régions

FIG. 7 – Graphe des régions et mise à jour d'un message

messages de type I entrant dans la région R_E (en gris clair sur la figure) et les messages de type II entrant dans la région R_P (en gris foncé sur la figure). On différencie deux types de messages car l'action sur la mise à jour n'est pas la même (voir Algorithme 1). On démontre que l'équation de mise à jour, conformément à la figure 7.b, s'écrit :

$$m^{update}_{\mathcal{E}\to\mathcal{P}}(x_{\mathcal{P}}) = \frac{\sum_{\mathbf{x}_E\setminus\mathbf{x}_P} L_{\mathcal{E}\setminus\mathcal{P}}(\mathbf{x}_{\mathcal{E}}\setminus\mathbf{x}_{\mathcal{P}}) \prod_{\substack{X\not\subset\mathcal{D}(\mathcal{E})\\ Y\subset\mathcal{D}(\mathcal{E})\setminus\mathcal{D}(\mathcal{P})}} m^{old}_{X\to Y}(x_Y)}{\prod_{\substack{X\subset\mathcal{E}(\mathcal{E})\setminus\mathcal{D}(\mathcal{P})\\ Y\subset\mathcal{E}(\mathcal{P})}} m^{update}_{X\to Y}(\mathbf{x}_Y)} \tag{4}$$

où : $\mathcal{D}(\mathcal{E}) = \mathcal{E}\bigcup_{\mathcal{R}\subset\mathcal{E}}\mathcal{R}$,
$L_{\mathcal{E}\setminus\mathcal{P}}(\mathbf{x}_{\mathcal{E}}\setminus\mathbf{x}_{\mathcal{P}}) =$ vraisemblances $\times$ parite de l'ensemble $\mathcal{E}\setminus\mathcal{P}$.

Algorithme - Croyance
Le but du décodeur est de trouver les croyances sur les noeuds de variables élémentaires, dont le calcul est expliqué dans [2]. De manière synthétique, il suffit, pour une région R quelconque du graphe, de trouver tous les messages entrant dans la région R (voir Algorithme 2).

Conclusion

L'algorithme de la propagation de croyances généralisée donne de meilleurs résultats que la propagation de croyances. Cependant, l'absorption des cycles n'est pas invincible : les cycles contenant deux noeuds de variable sont très bien gérés par l'algorithme mais les cycles contenant au moins trois noeuds de variable rendent le calcul des messages approximatifs. L'étude de la propagation de croyances généralisée nécessite donc une étude précise sur les méthodes bayésiennes dans les réseaux markoviens : il est possible d'améliorer l'algorithme en incluant des méthodes locales de maximum a posteriori pour absorber des cycles de petites taille, et d'appliquer l'algorithme sur le graphe ainsi modifier. Cette amélioration est une perspective d'étude, et demande donc une étude approfondie.

Algorithme 1 : Mise à jour du message de l'enfant E vers le parent P

```
début
    R_E ⟵ Région de l'enfant
    R_P ⟵ Région du parent
    M = 1 ⟵ valeur d'initialisation du message
    pour chaque niveau n du graphe faire
        pour chaque région r du niveau n faire
            si r ∈ R_E\R_P alors
                Message de type I
                pour chaque enfant e_r du groupe r faire
                    si e_r ∉ R_E alors
                        M = M × m_{e_r→r}
            si r ∈ R_P alors
                Message de type II
                pour chaque enfant e_r du groupe r faire
                    si e_r ∈ R_E\R_P alors
                        M = M / m_{e_r→r}
fin
```

Algorithme 2 : Calcul de la croyance sur une région R

```
début
    R ⟵ Région considérée
    pour chaque niveau n du graphe faire
        pour chaque région r du niveau faire
            si r ∉ R alors
                pour chaque parent p_r de r faire
                    si p_r ∈ R alors
                        M = M × m_{r→p_r}
fin
```

Bibliographie

1. Amin Shokrollahi. LDPC Codes : An Introduction. avril 2003.
2. Jonathan S. Yedida, William T. Freeman, and Yair Weiss. Understanding Belief Propagation and its Generalizations. janvier 2002.
3. Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. Février 2001.
4. Hidetoshi Nishimori. *Statistical Physics of Spin Glasses and Information Processing : An Introduction*. Clarendon Press, 2001.
5. Jonathan S. Yedida, William T. Freeman, and Yair Weiss. Constructing free energy approximations and Generalized Belief Propagation algorithms. décembre 2004.