Codes convolutifs et codes concaténés associés

Charly Poulliat

6 décembre 2020

Plan

Plan

1. Un exemple : le code $(5,7)_8$
   - Exemple du code (5,7)
2. Codes convolutifs : représentations
3. Décodage MAP

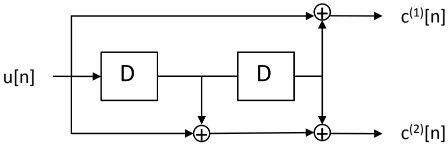
]

Plan

1. Un exemple : le code $(5,7)_8$
   - Exemple du code (5,7)

Introduction : le code $(5,7)_8$

Représentation par registre à décalage



- Représentation vectorielle : $g^{(1)} = [101] = [5]_8$ et $g^{(2)} = [111] = [7]_8$
- Représentation polynomiale (opérateur du retard D) : $g^{(1)}(D) = 1 + D^2$ et $g^{(2)}(D) = 1 + D + D^2$
- Rendement : $R = k/n = 1/2$ (k entrées, n sorties).
- *Mémoire* $\nu = 2$, *longueur de contrainte* $l_c = \nu + 1$.

Introduction : le code $(5,7)_8$

Notations

- Relations entrées-sorties en temps : soit $u[n]$ la séquence d'entrée, on a alors comme sorties

$$c^{(i)}[n] = u \circledast g^{(i)}[n], \ \forall i \in \{1,2\}$$

où $\circledast$ représente le produit de convolution sur $GF(2)$.

- Représentation polynomiale : $c^{(i)}(D) = u(D)g^{(i)}(D), \ \forall i \in \{1,2\}$
- Représentation vectorielle polynomiale :

$$\underline{c}(D) = [c^{(1)}(D), c^{(2)}(D)] = u(D)[g^{(1)}(D), g^{(2)}(D)] = u(D)G(D)$$

où $G(D)$ matrice génératrice polynômiale de taille $k \times n$.

Introduction : le code $(5,7)_8$

Terminaison de codage

- Troncature : pas de terminaison particulière du codage. Pour K bits en entrée, on a émis $N = K/R$ bits codés.
- Fermeture de treillis : On ajoute aux K bits d'information ν bits, dits de fermeture, pour rejoindre l'état 'nul' du registre. Cela entraine une baisse du rendement R. Ici, $R = K/2 * (K+2)$.
- Fermeture circulaire de treillis : tail-biting

Plan

1 Un exemple : le code $(5, 7)_8$

2 Codes convolutifs : représentations
- Codes convolutifs - cas général
- Codes convolutifs récursifs
- Représentation par machine à états finis
- Représentation en treillis

3 Décodage MAP

]

Plan

2 Codes convolutifs : représentations

- Codes convolutifs - cas général
- Codes convolutifs récursifs
- Représentation par machine à états finis
- Représentation en treillis

Codes convolutifs

Notations

- Rendement : k entrées et n sorties d'où le rendement $R = k/n$.
- Représentation vectorielle polynomiale : soient
  $\underline{u}(D) = [u^{(1)}(D), u^{(2)}(D), \ldots, u^{(k)}(D)]$,
  $\underline{c}(D) = [c^{(1)}(D), c^{(2)}(D), \ldots, c^{(n)}(D)]$

  $$\underline{c}(D) = \sum_{i=1}^{k} u^{(i)}(D)\underline{g}_i(D)$$

  où $\underline{g}_i(D) = [g_i^{(1)}(D), g_i^{(2)}(D), \ldots g_i^{(n)}(D)]$ et $g_i^{(j)}(D)$ représente le transfert entre l'entrée i et la sortie j.

  $$\underline{c}(D) = \underline{u}(D)G(D)$$

  où $G(D)$ matrice génératrice polynômiale de taille $k \times n$.
- Matrice de parité : $\underline{c}(D)H^T(D)$ et donc $G(D)H^T(D) = \mathbf{0}$

Codes convolutifs

Notations

- Code récursif : certaines relations entrées sorties peuvent être définies par un code récursif

$$g_i^{(j)}(D) = \frac{b_0 + b_1 D + b_2 D^2 + \ldots + b_m D^m}{1 + a_1 D + a_2 D^2 + \ldots + a_m D^m}$$

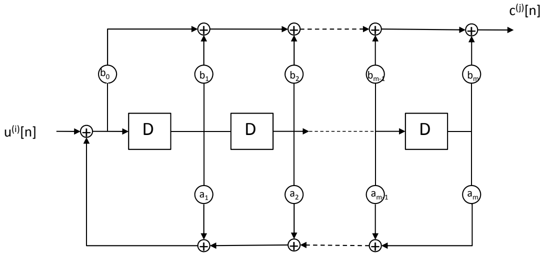


Figure – Implémentation d'un transfert récursif par réalisation d'un filtre récursif à réponse impulsionnelle infinie de Type I

Codes convolutifs

Notations

- **Exemple :** code récursif systématique $(1, 5/7)_8$ de matrice génératrice

$$G(D) = [1 \, \frac{1 + D^2}{1 + D + D^2}]$$

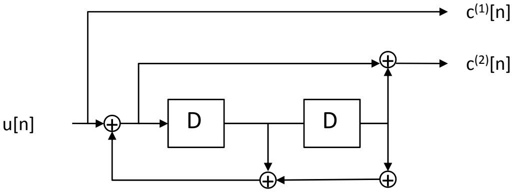
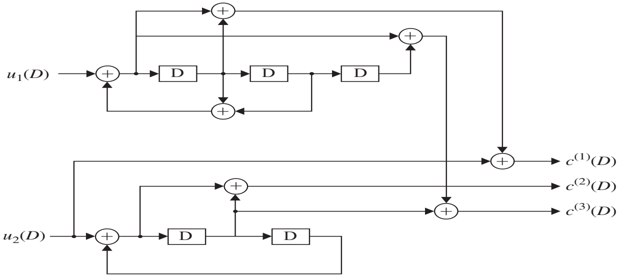


Figure – représentation du code récursif (1,5/7)

Codes convolutifs

Notations - exemple pour un code rendement $R = k/n$



$$G(D) = \begin{pmatrix} \frac{1+D}{1+D+D^2} & 0 & 1+D \\ 1 & \frac{1}{1+D} & \frac{D}{1+D^2} \end{pmatrix}$$

$$\underline{u}(D) = [u_1(D), u_2(D)], \underline{c}(D) = [c_1(D), c_2(D), c_3(D)]$$

Codes convolutifs

Modèle d'état et représentation par machine à états finis

- Représentation par machine à états finis : pour un code de type $R = 1/n$,

$$\{u_k\} \rightarrow \boxed{\underline{S}n} \rightarrow \{c_k\}$$

où $c_k = [c_k^{(1)}, c_k^{(2)}, c_k^{(n)}]$ et $\underline{S}_k$ représente l'état interne du registre à décalage.

- Représentation fonctionnelle associée :
  - Equation d'évolution : passage d'un état à $\underline{S}_{k-1}$ à $\underline{S}_k$.

$$\underline{S}_k = F_1(\underline{S}_{k-1}, u_k)$$

  - Equation d'observation : génération des sorties observables c_k.

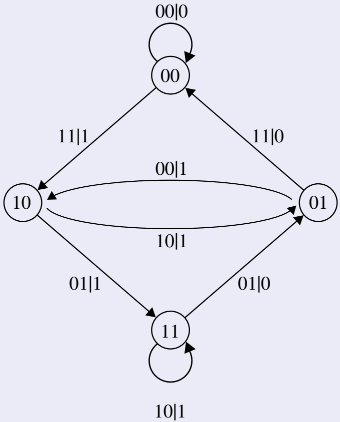
$$c_k = F_2(\underline{S}_{k-1}, u_k)$$

- Exemple : Code $(7, 5)_8$, $\underline{S}_k = [u_k, u_{k-1}]$.

Codes convolutifs

Représentation graphique en Diagramme d'Etat

Code convolutif $(7,5)_8$



Codes convolutifs : Représentation en treillis

- Definition : représentation graphique du code dans son espace d'état en considérant la dimension temporelle.
  - Noeuds : noeuds du graphe associé à un état $\underline{S}_k$.
  - Transitions : branches entre deux noeuds associées à $F_1(.)$.
  - Etiquettes : informations portées par les branches (u_k, c_k) et données par $F_2(.)$
- Propriétés :
  - Pour un treillis de longueur L, tout chemin du treillis de $\underline{S}_0$ à $\underline{S}_L$ est mot de code obtenu par concaténation des étiquettes c_k du chemin.
  - Le chemin où tous les $\underline{S}_k$ sont l'état **0** (représentation binaire naturelle) est le mot de code nul.
  - *Événement minimal :* chemin qui par de l'état $\underline{S}_k = \mathbf{0}$ et ne revient à cet état que à $\underline{S}_{k+l}$ pour un certain l. On lui associe un poids de Hamming grâce aux étiquettes du chemin.
  - $d_{\min}$ est donnée par le poids minimal d'un évênement minimal.

Code convolutif : représentation en treillis

Code $(7,5)_8$

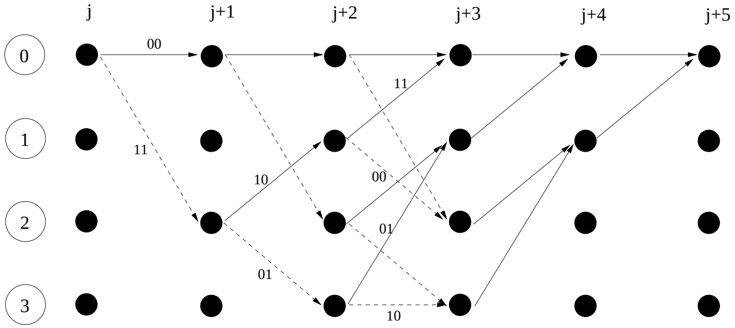


Figure – Treillis du code (7,5)

Plan

1. Un exemple : le code $(5,7)_8$
2. Codes convolutifs : représentations
3. Décodage MAP
   - Critère MAP bit
   - Algorithme BCJR
   - BCJR - domaine logarithmique

]

Plan

3. Décodage MAP
   - Critère MAP bit
   - Algorithme BCJR
   - BCJR - domaine logarithmique

Décodage MAP bit

Notations 1/2

Critère MAP bit

$$\begin{aligned} \hat{u}_n &= \arg\max_{u_n} p(u_n|\mathbf{y}) \\ &= \text{signe}(L(u_n)) \end{aligned} \quad (1)$$

avec

- mapping BPSK : $\{`0' \leftrightarrow +1, `1' \leftrightarrow -1\}$.
- $u_n \in \{-1, +1\}, \forall n \in [1, L]$
- LLR MAP (Log Likelyhood Ratio) :

$$L(u_n) = \log\left[\frac{p(u_n = +1|\mathbf{y})}{p(u_n = -1|\mathbf{y})}\right]$$

Décodage MAP bit

Notations 2/2

Critère MAP bit

- Longueur de treillis $L = K + Nf$ (K bits d'info. + Nf bits de fermeture).
- Notations vectorielles :

$$\mathbf{c} = [c_1, c_2, \dots c_L]$$

(mot de code émis) avec $c_k = [c_k^{(1)}, c_k^{(2)}, \dots, c_k^{(n_c)}]$

$$\mathbf{y} = [y_1, y_2, \dots y_L]$$

(mot de code reçu) avec $y_k = [y_k^{(1)}, y_k^{(2)}, \dots, y_k^{(n_c)}]$ et $y_k^{(j)} = c_k^{(j)} + b_k^{(j)}$

$$\mathbf{y}_k^l = [y_k, y_{k+1}, \dots y_{l-1}, y_l]$$

Décodage MAP par bit

Algorithme BCJR 1

LLR MAP revisité

$$L(u_n) = \log\left[\frac{p(u_n = +1|\mathbf{y})}{p(u_n = -1|\mathbf{y})}\right] = \log\left[\frac{\sum_{\mathcal{S}^+} p(s_{n-1} = s', s_n = s, \mathbf{y})}{\sum_{\mathcal{S}^-} p(s_{n-1} = s', s_n = s, \mathbf{y})}\right] \quad (2)$$

Notations

- Ensemble des transitions associées à $u_n = +1$:

$$\mathcal{S}^+ = \{(s', s) \text{ où } (s_{n-1} = s') \mapsto (s_n = s) | u_n = +1\}$$

- Ensemble des transitions associées à $u_n = -1$:

$$\mathcal{S}^- = \{(s', s) \text{ où } (s_{n-1} = s') \mapsto (s_n = s) | u_n = -1\}$$

Décodage MAP par bit

Algorithme BCJR 2

Factorisation de $p(s', s, \mathbf{y})$

$$
\begin{aligned}
p(s_{n-1} = s', s_n = s, \mathbf{y}) &= \alpha_{n-1}(s')\gamma_n(s', s)\beta_n(s) && (3) \\
&&& (4) \\
\alpha_n(s) &= p(s_n = s, \mathbf{y}_1^n) && (5) \\
\beta_n(s) &= p(\mathbf{y}_{n+1}^L | s_n = s) && (6) \\
\gamma_n(s', s) &= p(s_n = s, y_n | s_{n-1} = s') && (7)
\end{aligned}
$$

Récursions forward-backward

$$
\begin{aligned}
\alpha_n(s) &= \sum_{s'} \gamma_n(s', s)\alpha_{n-1}(s') && (8) \\
\beta_{n-1}(s') &= \sum_{s} \gamma_n(s', s)\beta_n(s) && (9)
\end{aligned}
$$

Décodage MAP par bit

Algorithme BCJR 3

Calcul des probabilités de transitions

$$\begin{aligned} \gamma_n(s', s) &= p(s_n = s, y_n | s_{n-1} = s') && (10) \\ &= p(y_n | s', s).p(s | s') && (11) \\ & && (12) \end{aligned}$$

avec

$$p(s|s') = \begin{cases} 0 & , \text{si } \{s' \rightarrow s\} \text{ non valide} \\ \pi(u_n) & , \text{sinon} \end{cases}$$

$$\gamma_n(s', s) = p(y_n | c_n(s', s)) \pi(u_n) \mathbb{1}_{s' \rightarrow s} \qquad (13)$$

avec

$\pi(u_n)$ probabilité à priori de u_n

$c_n(s', s)$ les bits associés à l'étiquette $(s' \rightarrow s)$

Décodage MAP par bit

Algorithme BCJR 4

Calcul des probabilités de transitions : cas Gaussien

$$y_n^{(m)} = c_n^{(m)} + b_n^{(m)},\, b_n^{(m)} \sim \mathcal{N}(0, \sigma_b^2)$$

$$\gamma_n(s', s) \propto \pi(u_n) \exp\left(- \frac{\sum_{m=1}^{n_c} |y_n^{(m)} - c_n^{(m)}(s', s)|^2}{2\sigma_b^2} \right) \mathbb{1}_{s' \to s}$$

Initialisation (fermeture treillis)

$$\alpha_0(0) = 1 \quad , \quad \alpha_0(s) = 0 \text{ sinon} \tag{14}$$

$$\beta_L(0) = 1 \quad , \quad \beta_L(s) = 0 \text{ sinon} \tag{15}$$

Décodage MAP par bit

Algorithme BCJR dans domaine logarithmique

Définitions dans le domaine logarithmique

$$
\begin{aligned}
\tilde{\alpha}_n(s) &\triangleq \log\left(\alpha_n(s)\right) \\
&= \log \sum_{s'} \exp\left(\tilde{\alpha}_{n-1}(s') + \tilde{\gamma}_n(s', s)\right) \qquad (16)
\end{aligned}
$$

$$
\begin{aligned}
\tilde{\beta}_{n-1}(s') &\triangleq \log\left(\beta_n(s')\right) \\
&= \log \sum_{s} \exp\left(\tilde{\beta}_n(s) + \tilde{\gamma}_n(s', s)\right) \qquad (17)
\end{aligned}
$$

$$
\tilde{\gamma}_n(s', s) \triangleq \log\left(\gamma_n(s', s)\right) \qquad (18)
$$

$$
\begin{aligned}
L(u_n) &= \log\left(\sum_{\mathcal{S}^+} \exp\left(\tilde{\alpha}_{n-1}(s') + \tilde{\gamma}_n(s', s) + \tilde{\beta}_n(s)\right)\right) \\
&\quad - \log\left(\sum_{\mathcal{S}^-} \exp\left(\tilde{\alpha}_{n-1}(s') + \tilde{\gamma}_n(s', s) + \tilde{\beta}_n(s)\right)\right)
\end{aligned}
$$

Décodage MAP par bit

Algorithme BCJR dans domaine logarithmique

Opérateur $\max^*(x, y)$

$$\max(x, y) = \log\left(\frac{e^x + e^y}{1 + e^{-|x-y|}}\right) \tag{19}$$

$$\begin{aligned} max^*(x, y) &\triangleq \log\left(e^x + e^y\right) \\ &= \max(x, y) - \log\left(1 + e^{-|x-y|}\right) \end{aligned} \tag{20}$$

$$\begin{aligned} max^*(x, y, z) &\triangleq \log\left(e^x + e^y + e^z\right) \\ &= max^*(max^*(x, y), z) \end{aligned} \tag{21}$$

Décodage MAP par bit

Log-MAP (log-BCJR)

$$\tilde{\alpha}_n(s) = \max_{s'}{}^*(\tilde{\alpha}_{n-1}(s') + \tilde{\gamma}_n(s', s)) \quad (22)$$

$$\tilde{\beta}_{n-1}(s') = \max_{s}{}^*(\tilde{\beta}_n(s) + \tilde{\gamma}_n(s', s)) \quad (23)$$

$$\begin{aligned} L(u_n) &= \max_{\mathcal{S}^+}{}^*\left(\tilde{\alpha}_{n-1}(s') + \tilde{\gamma}_n(s', s) + \tilde{\beta}_n(s)\right) \\ &\quad -\max_{\mathcal{S}^-}{}^*\left(\tilde{\alpha}_{n-1}(s') + \tilde{\gamma}_n(s', s) + \tilde{\beta}_n(s)\right) \end{aligned} \quad (24)$$

- Implémentation simplifiée par l'opérateur max(.) et utilisation de lookup table.
- stabilité numérique accrue.

Décodage MAP par bit

Algorithme BCJR dans domaine logarithmique

Log-MAP (log-BCJR) : cas Gaussien

$$\begin{aligned}\tilde{\gamma}_n(s',s) &= \log\left(\gamma_n(s',s)\right)\\ &= \log(P(u_n)) - \log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{m=1}^{n_c}|y_n^{(m)} - c_n^{(m)}(s',s)|^2\end{aligned}$$

en se réferant au critère MAP final, la constante peut-être supprimée.

Max-Log-MAP

- Remplacer l'opérateur $\max^*(.)$ par l'opérateur $\max(.)$ seul.
- Complexité diminuée, mais perte de performances (raisonnable). $\Rightarrow$ décodeur implémenté en pratique

Bibliographie

- A. Glavieux and all, *Channel coding in communication networks : from theory to turbocodes*, Volume 3 de Digital Signal Image Processing Series, John Wiley Sons, 2007.
- Claude Berrou and all, *Codes and Turbo Codes*, Collection IRIS Series, IRIS International, Springer, 2010.
- W.E. Ryan, Shu Lin, *Channel codes : classical and modern*, Cambridge University Press, 2009.
- Shu Lin, Daniel J. Costello, *Error control coding : fundamentals and applications*, Édition 2, Pearson-Prentice Hall, 2004.
- T. Richardson, R. Urbanke, *Modern coding theory*, Cambridge University Press, 2008.