

TS226

-

Codes correcteur d'erreur

Romain Tajan

29 septembre 2020

Plan

- 1 Previously on TS226 ...
- 2 Code convolutif comme machine à états
 - ▷ Diagramme d'état des codes convolutifs
 - ▷ Treillis des codes convolutifs
- 3 Décodage ML des codes convolutifs
 - ▷ Décodage par maximum de vraisemblance (ML)

Exemple de QCM

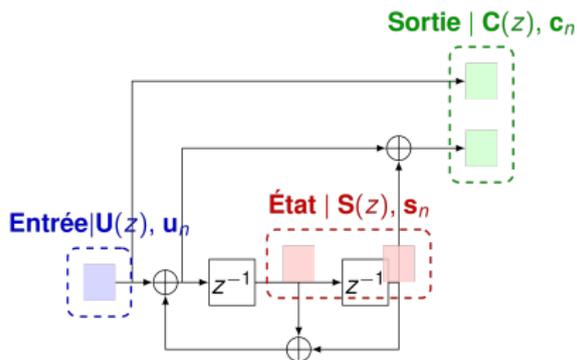
Comment allez vous aujourd'hui ?

- A Très bien
- B Bien
- C Mal
- D Très mal

Plan

- 1 Previously on TS226 ...
- 2 Code convolutif comme machine à états
 - ▷ Diagramme d'état des codes convolutifs
 - ▷ Treillis des codes convolutifs
- 3 Décodage ML des codes convolutifs
 - ▷ Décodage par maximum de vraisemblance (ML)

Rappels / définitions



• **Entrée** : $\mathbf{U}(z) = [U^{(0)}(z), U^{(1)}(z) \dots U^{(n_b-1)}(z)]$

→ dans ce cours $n_b = 1 \Rightarrow \mathbf{U}(z) \rightarrow U(z)$

• **État** : $\mathbf{S}(z) = [S^{(0)}(z), S^{(1)}(z) \dots S^{(m-1)}(z)]$

→ m est appelé "mémoire du code"

→ $\nu = m + 1$ est appelé "longueur de contrainte"

→ dans l'exemple $m = 2$

→ 2^m : nombre d'états

• **Sortie** : $\mathbf{C}(z) = [C^{(0)}(z), C^{(1)}(z) \dots C^{(m-1)}(z)]$

→ n_s nombre de sorties

→ dans l'exemple $n_s = 2$

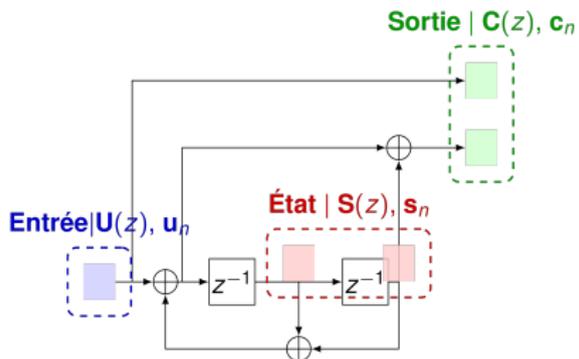
→ De façon générale : $\mathbf{C}(z) = U(z)\mathbf{G}(z)$

→ où $\mathbf{G}(z) = \left[\frac{A^{(1)}(z)}{B^{(1)}(z)} \dots \frac{A^{(n_s)}(z)}{B^{(n_s)}(z)} \right]$

• **Rendement du code** : $R = \frac{\text{\#bits d'info. en entrée}}{\text{\#bits codés en sortie}}$

→ Ici : $R = \frac{n_b}{n_s} = \frac{1}{2}$

Rappels / définitions



- **Code linéaire**

→ Si $\mathbf{C}_1(z)$ et $\mathbf{C}_2(z)$ sont deux mots de codes, alors $\mathbf{C}_3(z) = \mathbf{C}_2(z) + \mathbf{C}_1(z)$ est aussi un mot de code.

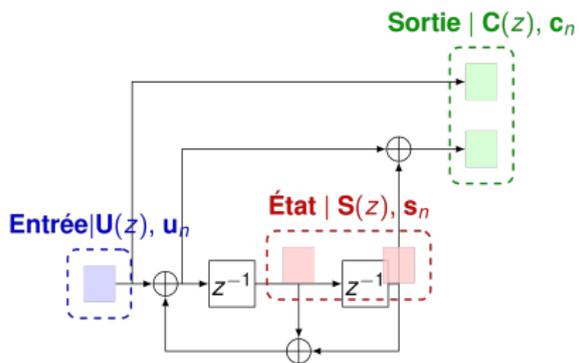
- **Encodeur récursif / non récursif**

- **Encodeur systématique / non systématique**

- **Notation octale**

Sortez vos téléphones...

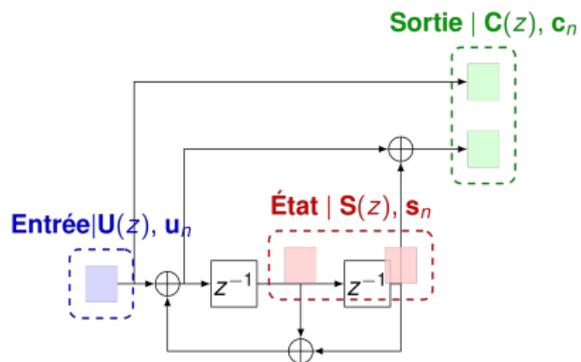
Quizz



Cet encodeur est récursif :

- A Vrai
- B Faux

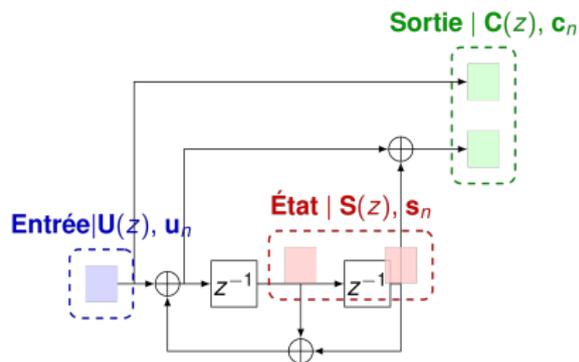
Quizz



Cet encodeur est systématique :

- A Vrai
- B Faux

Quizz



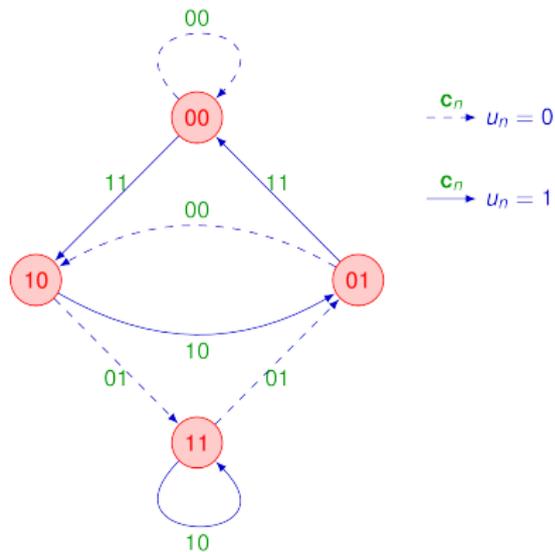
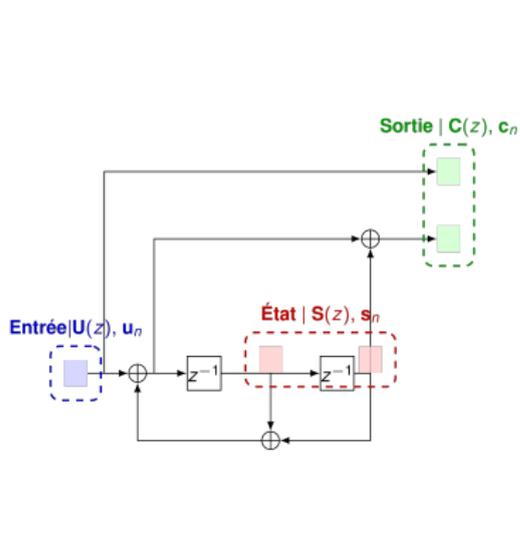
La notation octale de cet encodeur est :

- A** $(7, 5)_8$
- B** $(1, \frac{5}{7})_8$
- C** $(1, \frac{7}{5})_8$
- D** $(5, 7)_8$

Plan

- 1 Previously on TS226 ...
- 2 **Code convolutif comme machine à états**
 - ▷ Diagramme d'état des codes convolutifs
 - ▷ Treillis des codes convolutifs
- 3 Décodage ML des codes convolutifs
 - ▷ Décodage par maximum de vraisemblance (ML)

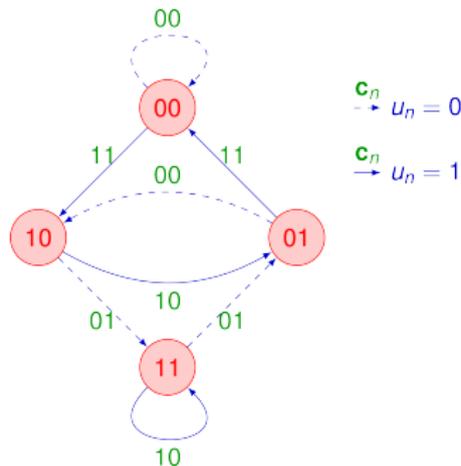
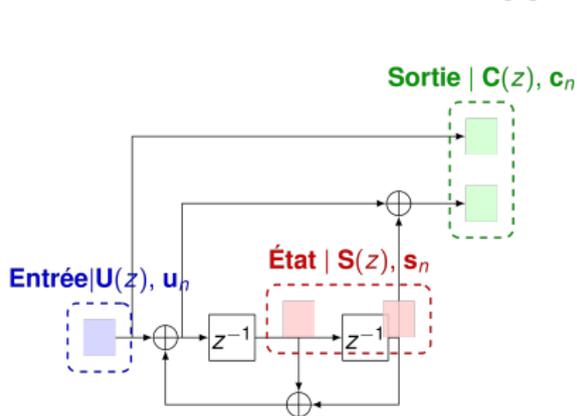
Diagramme d'états



- Message \Leftrightarrow chemin dans le graphe
- Mot de code \Leftrightarrow étiquettes le long du chemin dans le graphe
- Nécessité de définir (au moins) un état initial

Sortez vos téléphones...

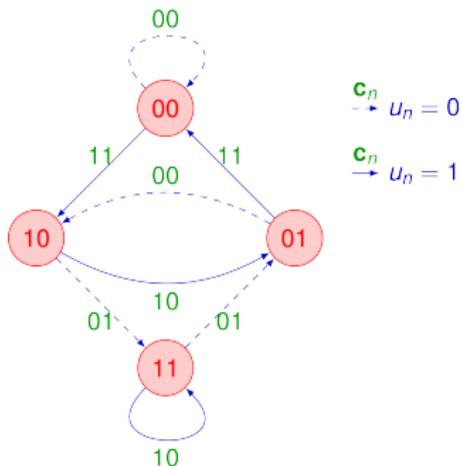
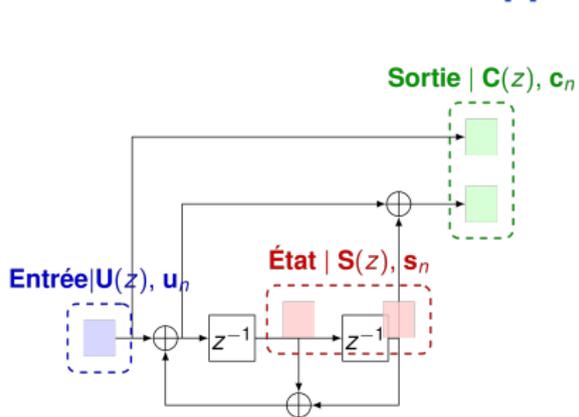
Rappels / définitions



Si $\mathbf{u} = [0, 1, 1, 0]$, en supposant que $\mathbf{s}_0 = [0, 0]$ quelle sera la sortie de l'encodeur :

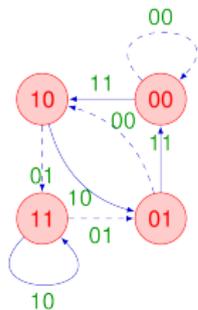
- A** $\mathbf{c} = [0, 0, 0, 1, 0, 1, 1, 1]$
- B** $\mathbf{c} = [0, 0, 1, 1, 0, 1, 1, 1]$
- C** $\mathbf{c} = [1, 0, 1, 1, 1, 0, 0, 1]$
- D** $\mathbf{c} = [0, 0, 1, 1, 1, 0, 0, 0]$

Rappels / définitions



Quel est le poids de Hamming minimal pour un mot de code $\mathbf{c} \neq \mathbf{0}$?

- A 2
- B 3
- C 4
- D 5
- E 6



c_n
 $\rightarrow u_n = 0$

c_n
 $\rightarrow u_n = 1$

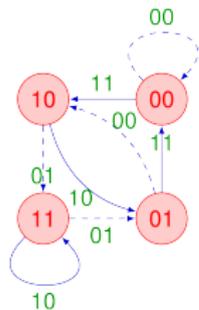
Treillis : représentation de la machine à états faisant explicitement apparaître le temps.

00 ●

01

10

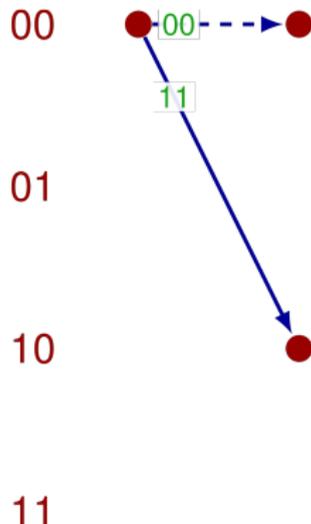
11

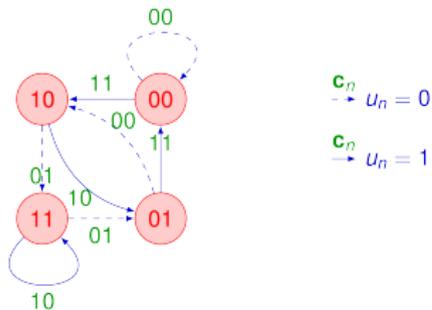


c_n
 $\rightarrow u_n = 0$

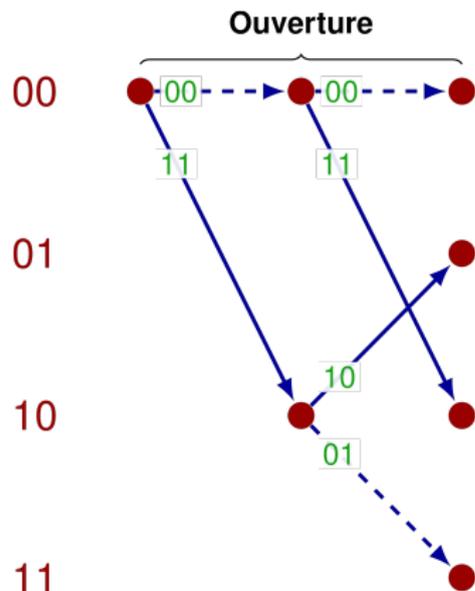
c_n
 $\rightarrow u_n = 1$

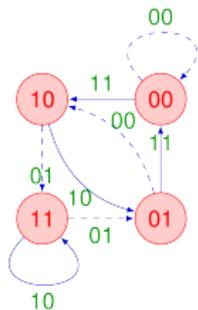
Treillis : représentation de la machine à états faisant explicitement apparaître le temps.





Treillis : représentation de la machine à états faisant explicitement apparaître le temps.

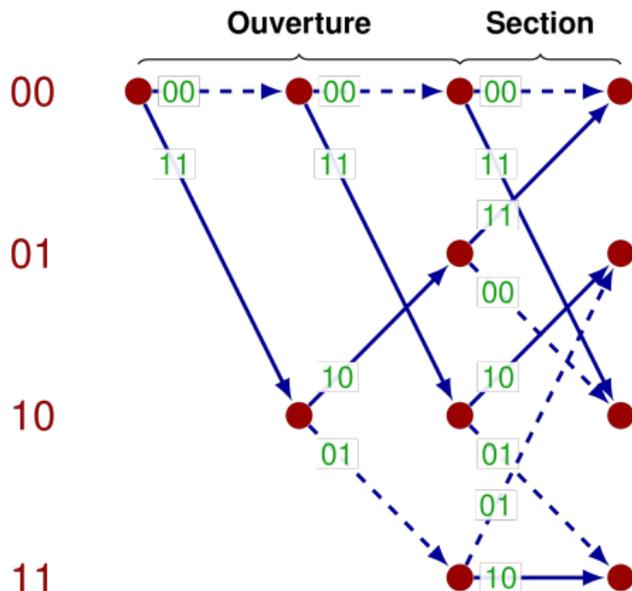


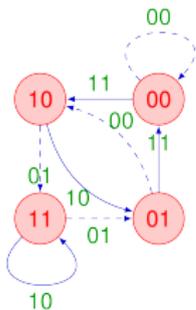


$$c_n \rightarrow u_n = 0$$

$$c_n \rightarrow u_n = 1$$

Treillis : représentation de la machine à états faisant explicitement apparaître le temps.

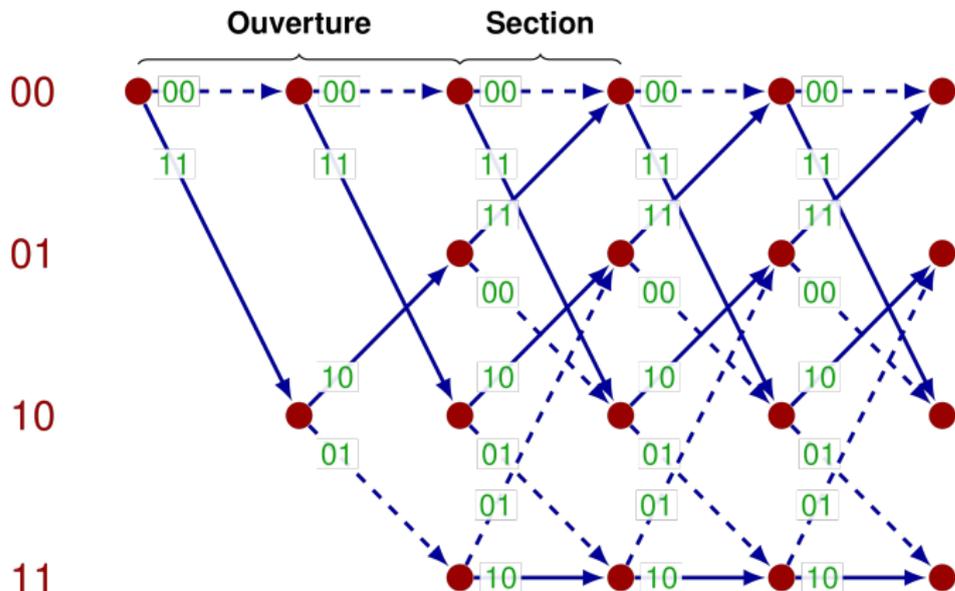


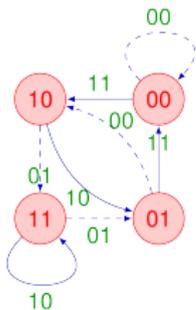


c_n
 $\rightarrow u_n = 0$

c_n
 $\rightarrow u_n = 1$

Treillis : représentation de la machine à états faisant explicitement apparaître le temps.

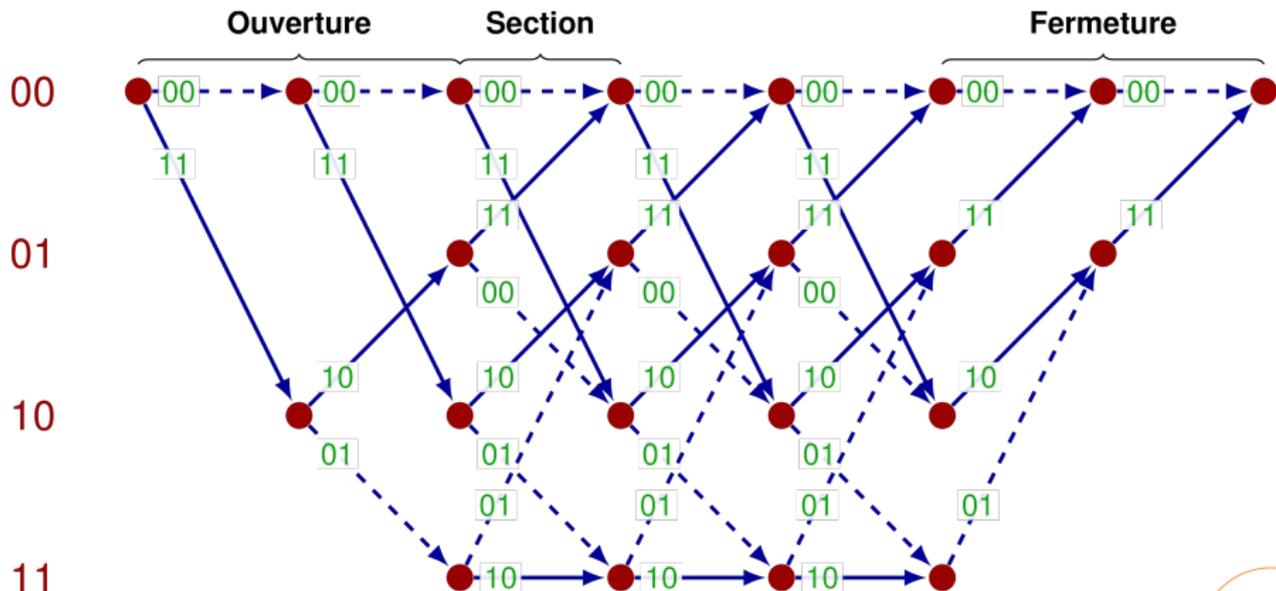


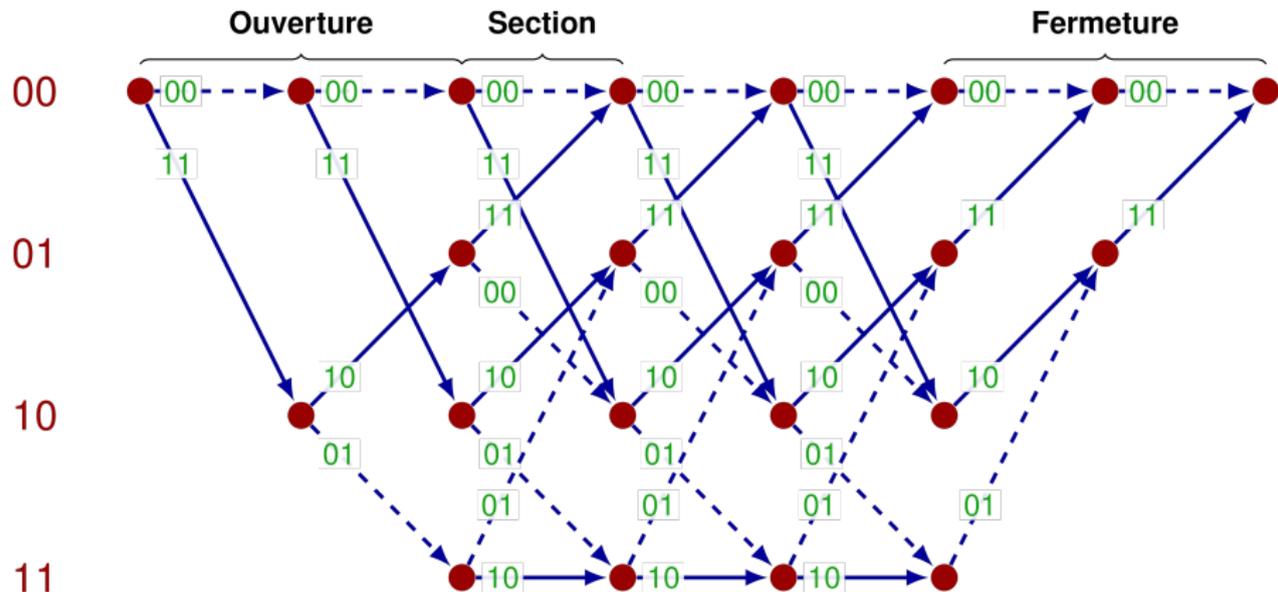


c_n
 $\rightarrow u_n = 0$

c_n
 $\rightarrow u_n = 1$

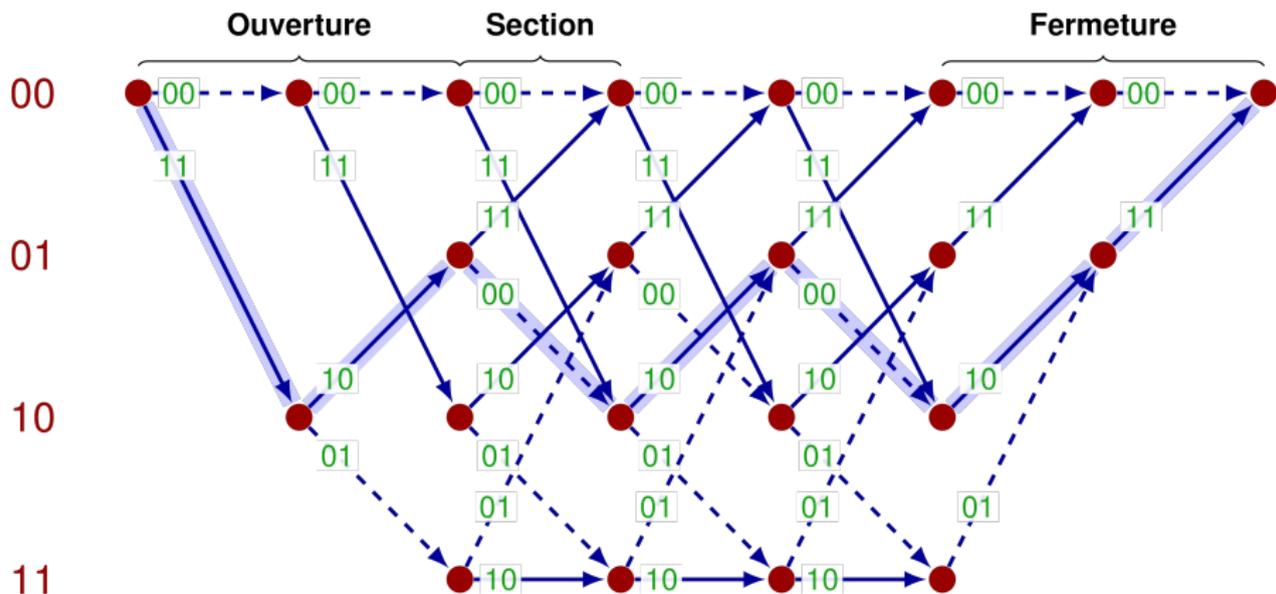
Treillis : représentation de la machine à états faisant explicitement apparaître le temps.





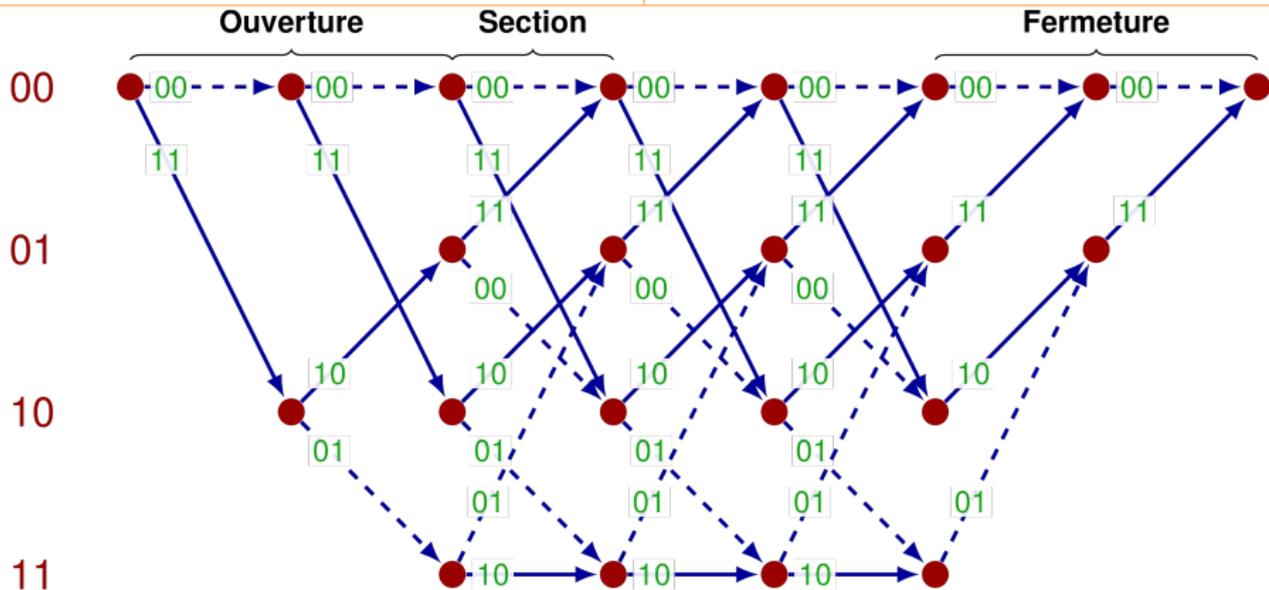
Remarques

- Fermer $\Rightarrow N = n_s(m + K) \Rightarrow R = \frac{K}{n_s(m + K)} \leq \frac{1}{n_s}$
- Tout chemin dans ce graphe représente un mot de code



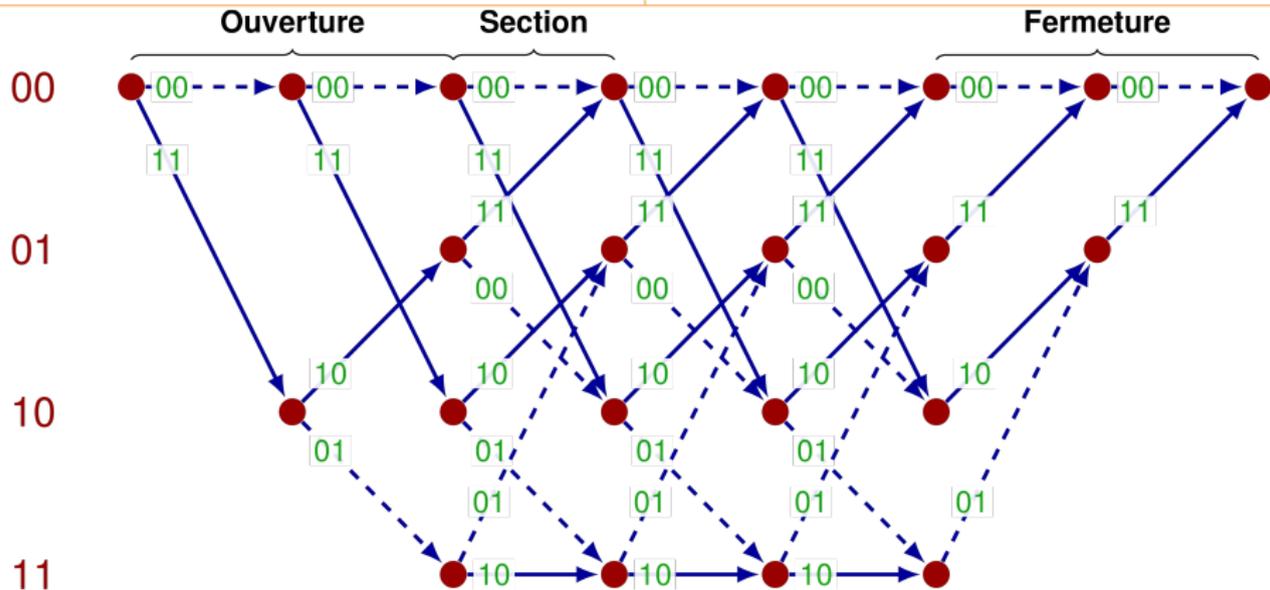
- On souhaite envoyer le message : $\mathbf{u} = [1, 1, 0, 1, 0]$
- On calcule le mot de code : $\mathbf{c} = [1\ 1, 1\ 0, 0\ 0, 1\ 0, 0\ 0, \underline{1\ 0}, \underline{1\ 1}]$
- On envoie le signal : $\mathbf{x} = [-1\ -1, -1\ 1, 1\ 1, -1\ 1, 1\ 1, \underline{-1\ 1}, \underline{-1\ -1}]$

Sortez vos téléphones...



Sur ce treillis, quel est le poids de Hamming minimum pour les mots de codes $\mathbf{c} \neq \mathbf{0}$?

- A 4
- B 5
- C 6
- D 7



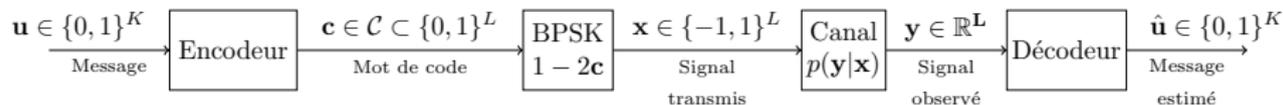
Sur ce treillis, combien de mots de codes $\mathbf{c} \neq \mathbf{0}$ ont un poids de Hamming de 5 ?

- A 4
- B 5
- C 6
- D 7

Plan

- 1 Previously on TS226 ...
- 2 Code convolutif comme machine à états
 - ▷ Diagramme d'état des codes convolutifs
 - ▷ Treillis des codes convolutifs
- 3 **Décodage ML des codes convolutifs**
 - ▷ **Décodage par maximum de vraisemblance (ML)**

Décodage du Maximum de Vraisemblance (ML)



Définition

- Le **décodeur du Maximum de vraisemblance (ML)** est la fonction de \mathbf{y} définie par :

$$\Psi_{ML}(\mathbf{y}) = \arg \max_{\mathbf{u} \in \{0, 1\}^K} p(\mathbf{y}|\mathbf{u})$$

- Le décodeur ML est équivalent au décodeur MAP si les messages sont équiprobables.
- Sur le canal AWGN sans mémoire, le décodeur ML est équivalent à :

$$\hat{\mathbf{u}} = \Psi_{ML}(\mathbf{y}) = \Phi^{-1}(\hat{\mathbf{c}}) \text{ où } \hat{\mathbf{c}} = \arg \min_{\mathbf{c} \in \mathcal{C}} \sum_{\ell=0}^{L-1} \mathbf{c}_\ell \mathbf{y}_\ell^T$$

- Nombre de bits dans le message : K
- Taille message + fermeture : $L = K + m$
- Le message envoyé : $\mathbf{u} = [u_0, u_1, \dots, u_{K-1}]$
- Le mot de code émis : $\mathbf{c} = [\mathbf{c}_0, \mathbf{c}_1, \dots, \mathbf{c}_{L-1}]$, où $\mathbf{c}_\ell = [c_\ell^{(0)}, c_\ell^{(1)}]$
- Le signal observé : $\mathbf{y} = [\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_{L-1}]$, où $\mathbf{y}_\ell = [y_\ell^{(0)}, y_\ell^{(1)}]$
- **Décodeur ML** :
$$\hat{\mathbf{c}} = \arg \min_{\mathbf{c} \in \mathcal{C}} \sum_{\ell=0}^{L-1} \mathbf{c}_\ell \mathbf{y}_\ell^T$$
- Le message reçu : $\hat{\mathbf{u}} = [\hat{u}_0, \hat{u}_1, \dots, \hat{u}_{K-1}] = \phi^{-1}(\hat{\mathbf{c}})$

• **Solution 1** : explorer tous les messages (eq. tous les mots de codes)

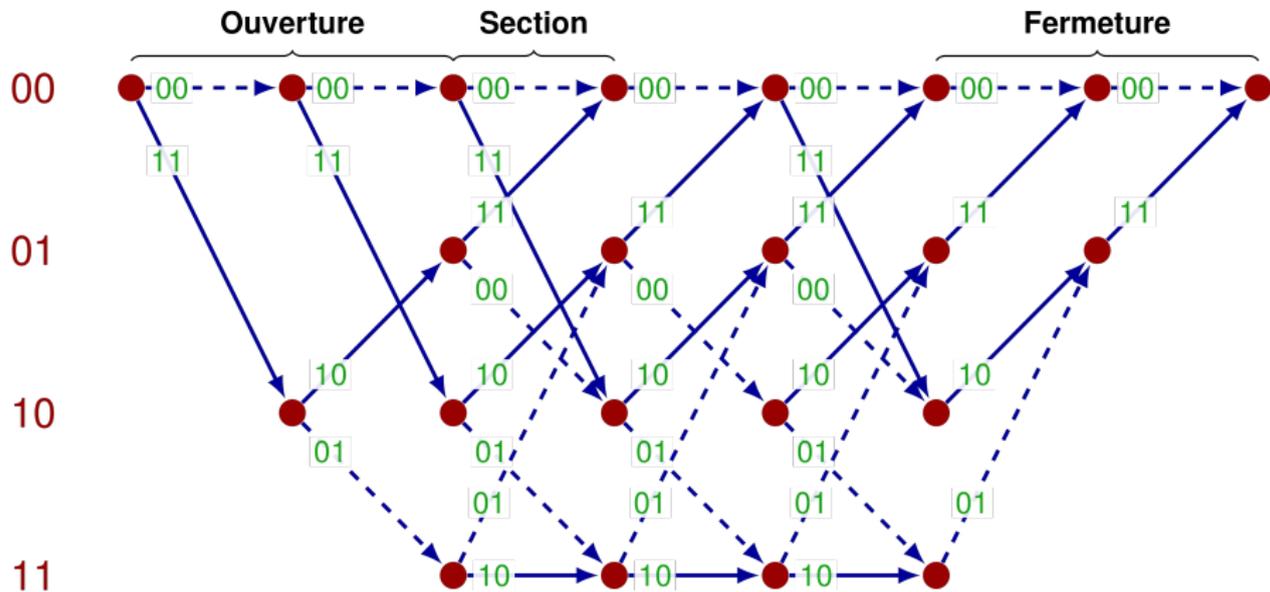
- Nombre de bits dans le message : K
- Taille message + fermeture : $L = K + m$
- Le message envoyé : $\mathbf{u} = [u_0, u_1, \dots, u_{K-1}]$
- Le mot de code émis : $\mathbf{c} = [\mathbf{c}_0, \mathbf{c}_1, \dots, \mathbf{c}_{L-1}]$, où $\mathbf{c}_\ell = [c_\ell^{(0)}, c_\ell^{(1)}]$
- Le signal observé : $\mathbf{y} = [\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_{L-1}]$, où $\mathbf{y}_\ell = [y_\ell^{(0)}, y_\ell^{(1)}]$
- **Décodeur ML** :
$$\hat{\mathbf{c}} = \arg \min_{\mathbf{c} \in \mathcal{C}} \sum_{\ell=0}^{L-1} \mathbf{c}_\ell \mathbf{y}_\ell^T$$
- Le message reçu : $\hat{\mathbf{u}} = [\hat{u}_0, \hat{u}_1, \dots, \hat{u}_{K-1}] = \phi^{-1}(\hat{\mathbf{c}})$

• **Solution 1** : explorer tous les messages (eq. tous les mots de codes) \rightarrow **il y en a 2^K ...**

- Nombre de bits dans le message : K
- Taille message + fermeture : $L = K + m$
- Le message envoyé : $\mathbf{u} = [u_0, u_1, \dots, u_{K-1}]$
- Le mot de code émis : $\mathbf{c} = [\mathbf{c}_0, \mathbf{c}_1, \dots, \mathbf{c}_{L-1}]$, où $\mathbf{c}_\ell = [c_\ell^{(0)}, c_\ell^{(1)}]$
- Le signal observé : $\mathbf{y} = [\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_{L-1}]$, où $\mathbf{y}_\ell = [y_\ell^{(0)}, y_\ell^{(1)}]$
- **Décodeur ML** :
$$\hat{\mathbf{c}} = \arg \min_{\mathbf{c} \in \mathcal{C}} \sum_{\ell=0}^{L-1} \mathbf{c}_\ell \mathbf{y}_\ell^T$$
- Le message reçu : $\hat{\mathbf{u}} = [\hat{u}_0, \hat{u}_1, \dots, \hat{u}_{K-1}] = \phi^{-1}(\hat{\mathbf{c}})$

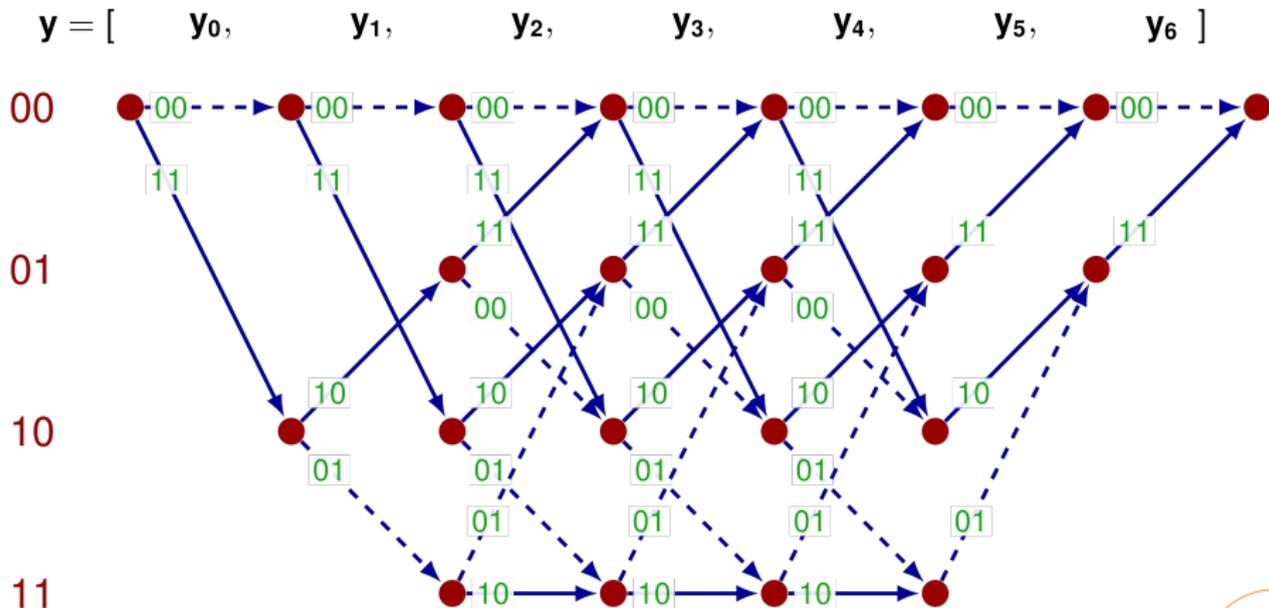
- **Solution 1** : explorer tous les messages (eq. tous les mots de codes) \rightarrow **il y en a 2^K ...**
- **Solution 2** : utiliser le treillis + la forme de la fonction de coût \rightarrow **algorithme de Viterbi**

- Soit $J_n(s_n) = \min_{\mathbf{c} \in \mathcal{C}(s_0 \rightarrow s_n)} \sum_{\ell=0}^{n-1} \mathbf{c}_\ell \mathbf{y}_\ell^T$
- Le décodage ML est équivalent à trouver l'antécédent de $J_L(s_L)$ où $s_0 = s_L = 0$



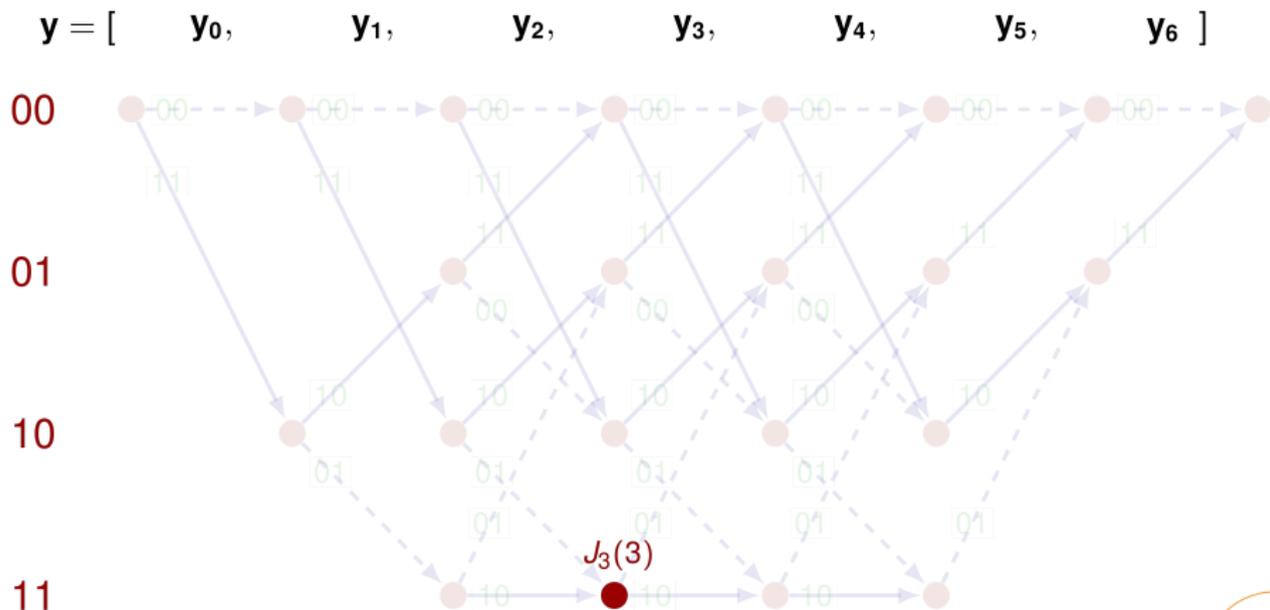
- $\mathcal{P}(s_n)$: **parents de l'état** s_n , i.e. ensemble des s_{n-1} tels que $s_{n-1} \rightarrow s_n$ existe
- **Algorithme de Viterbi** : pour chaque $n \in [0, L - 1]$ et chaque $s_n \in \mathcal{S}_n$ calculer

$$J_n(s_n) = \min_{s_{n-1} \in \mathcal{P}(s_n)} \left[J_{n-1}(s_{n-1}) + \mathbf{c}_{n-1}(s_{n-1} \rightarrow s_n) \mathbf{y}_{n-1}^T \right]$$
- $\hat{\mathbf{u}}$: **chemin survivant minimisant** $J_L(s_L)$



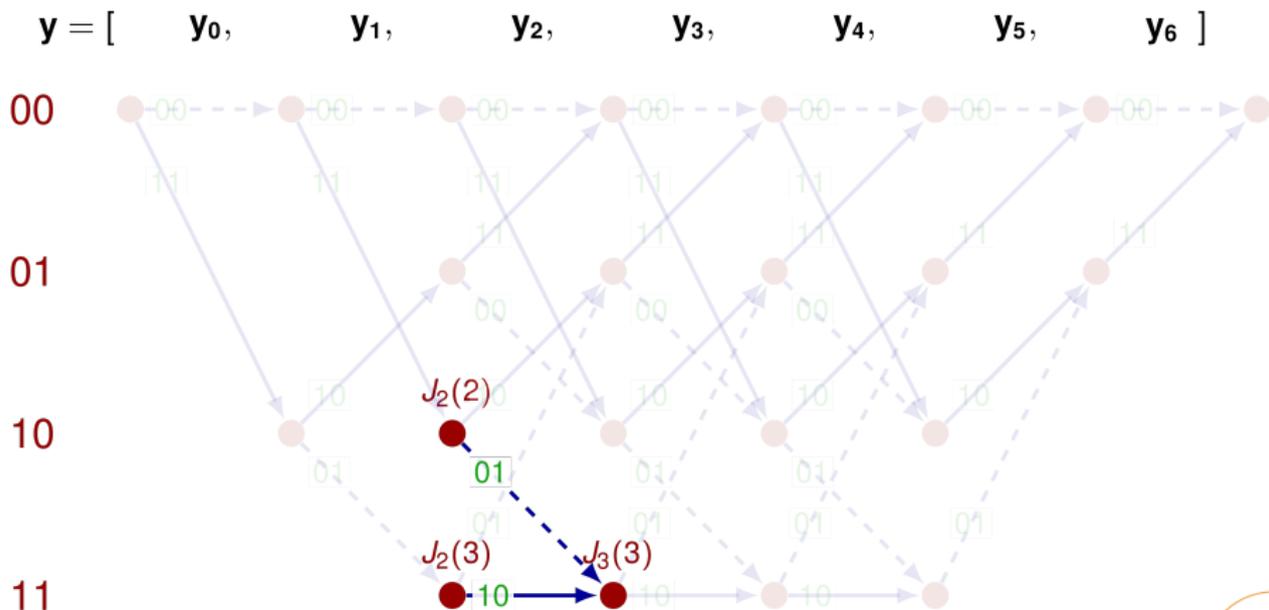
- $\mathcal{P}(s_n)$: **parents de l'état** s_n , i.e. ensemble des s_{n-1} tels que $s_{n-1} \rightarrow s_n$ existe
- **Algorithme de Viterbi** : pour chaque $n \in [0, L - 1]$ et chaque $s_n \in \mathcal{S}_n$ calculer

$$J_n(s_n) = \min_{s_{n-1} \in \mathcal{P}(s_n)} \left[J_{n-1}(s_{n-1}) + \mathbf{c}_{n-1}(s_{n-1} \rightarrow s_n) \mathbf{y}_{n-1}^T \right]$$
- \hat{u} : **chemin survivant minimisant** $J_L(s_L)$



- $\mathcal{P}(s_n)$: **parents de l'état** s_n , i.e. ensemble des s_{n-1} tels que $s_{n-1} \rightarrow s_n$ existe
- **Algorithme de Viterbi** : pour chaque $n \in [0, L - 1]$ et chaque $s_n \in \mathcal{S}_n$ calculer

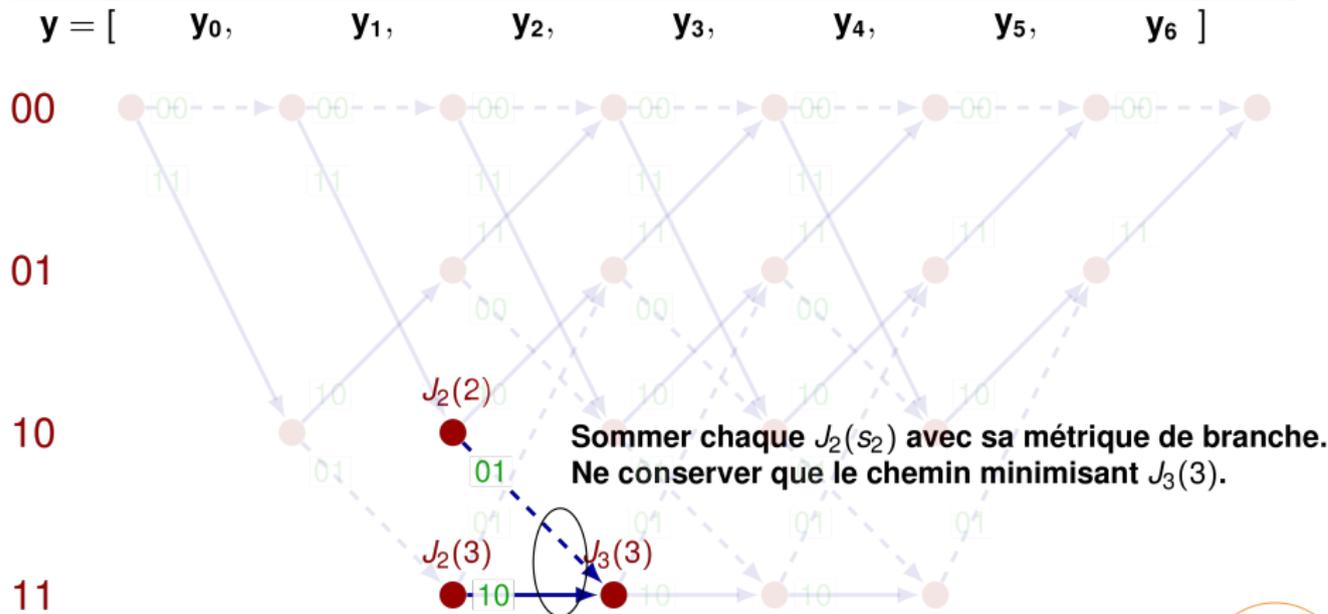
$$J_n(s_n) = \min_{s_{n-1} \in \mathcal{P}(s_n)} \left[J_{n-1}(s_{n-1}) + \mathbf{c}_{n-1}(s_{n-1} \rightarrow s_n) \mathbf{y}_{n-1}^T \right]$$
- \hat{u} : **chemin survivant minimisant** $J_L(s_L)$



- $\mathcal{P}(s_n)$: parents de l'état s_n , i.e. ensemble des s_{n-1} tels que $s_{n-1} \rightarrow s_n$ existe
- **Algorithme de Viterbi** : pour chaque $n \in [0, L - 1]$ et chaque $s_n \in \mathcal{S}_n$ calculer

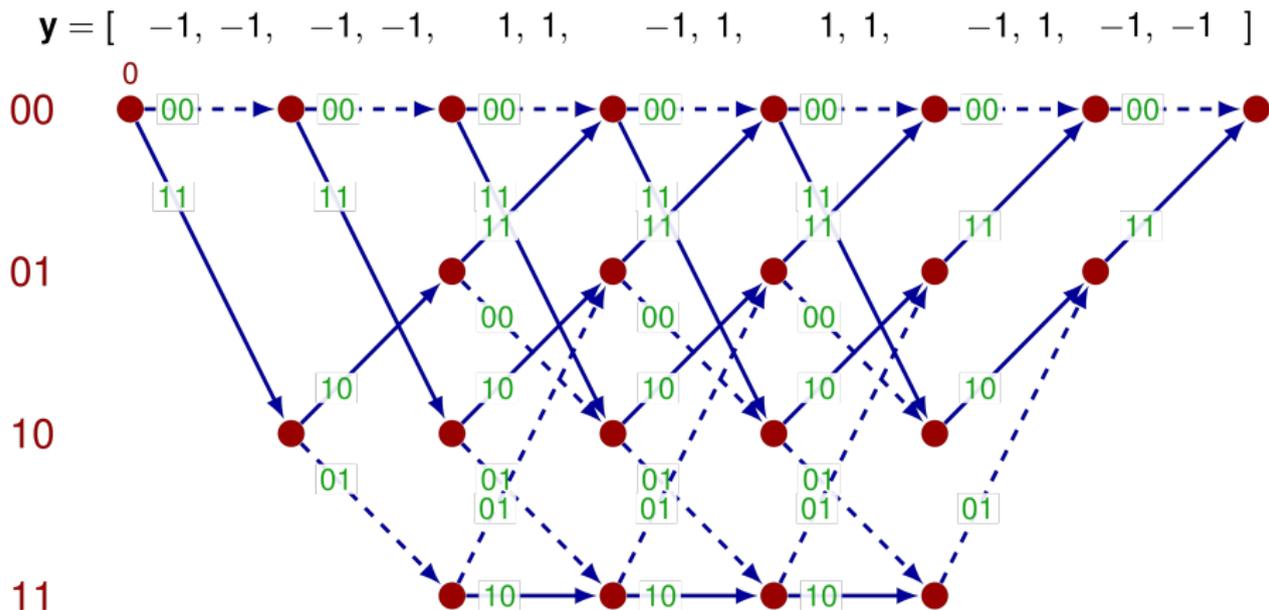
$$J_n(s_n) = \min_{s_{n-1} \in \mathcal{P}(s_n)} \left[J_{n-1}(s_{n-1}) + \mathbf{c}_{n-1}(s_{n-1} \rightarrow s_n) \mathbf{y}_{n-1}^T \right]$$

- \hat{u} : chemin survivant minimisant $J_L(s_L)$



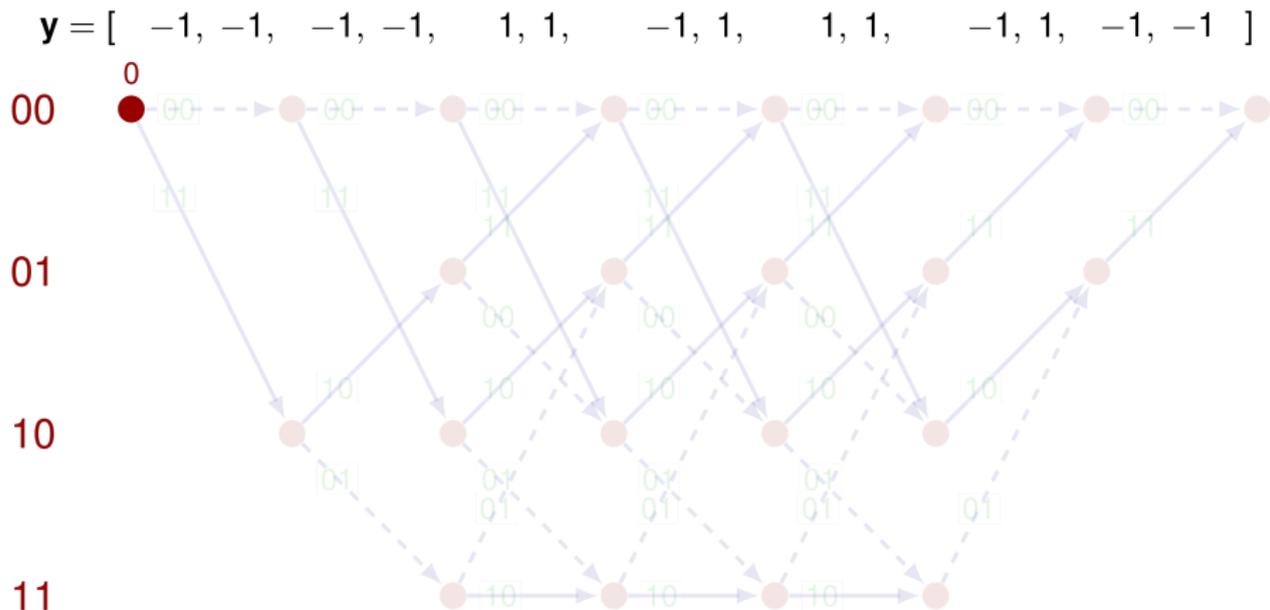
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



Exemple de question...

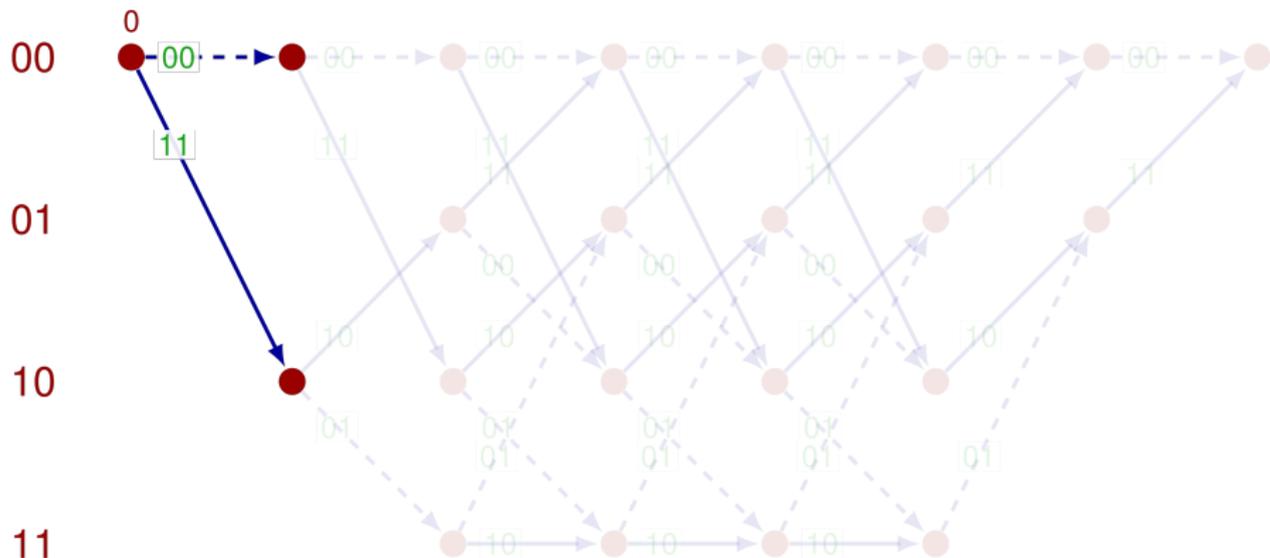
Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?

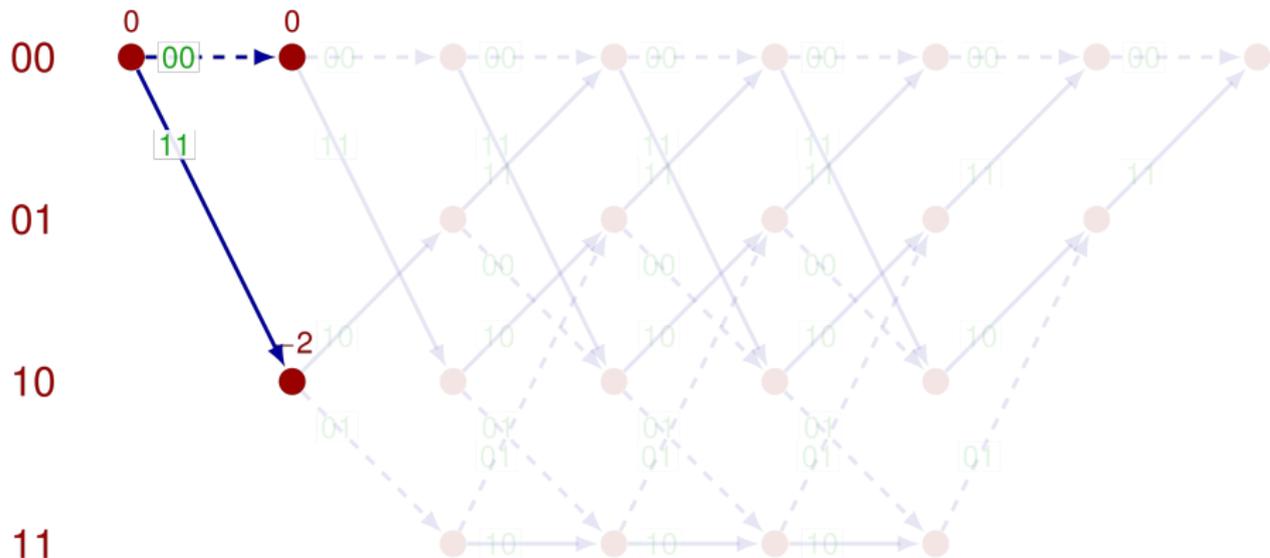
$\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$



Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?

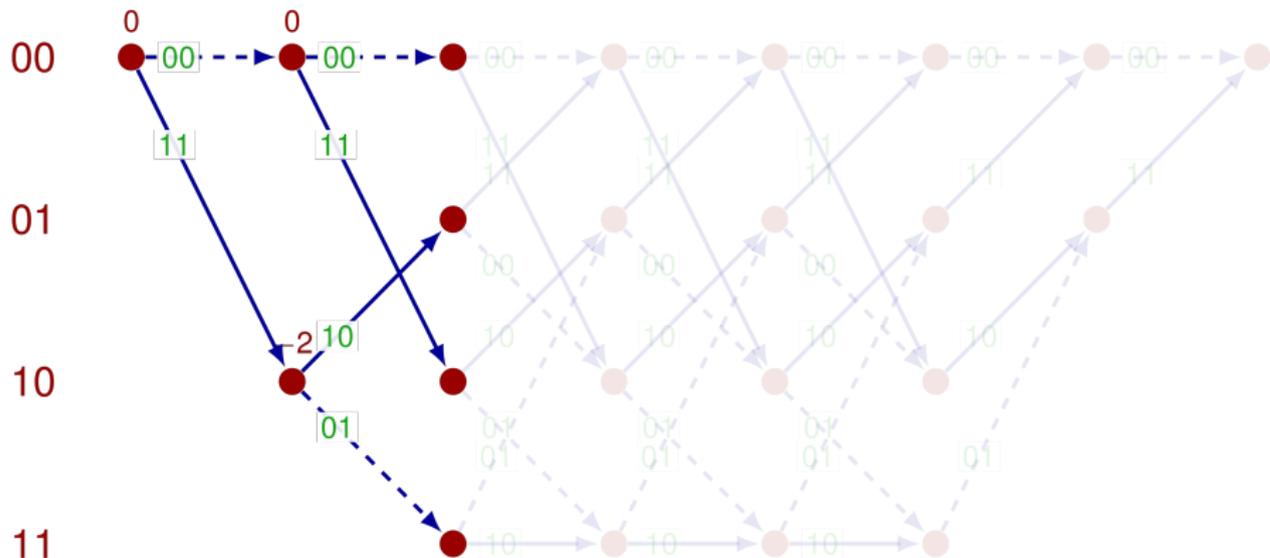
$\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$



Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?

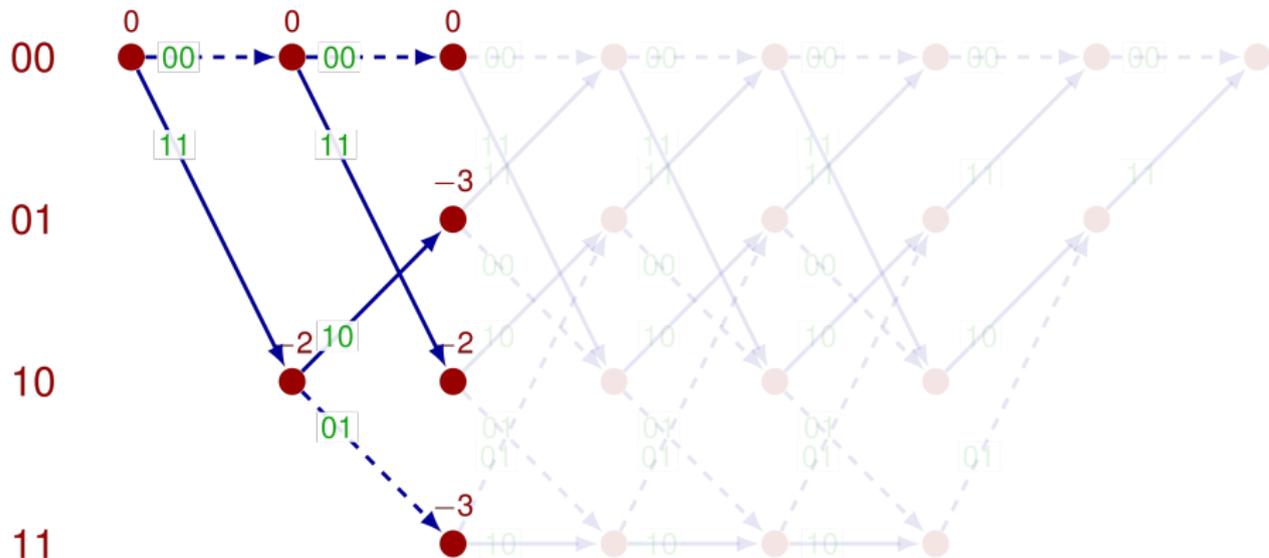
$\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$



Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?

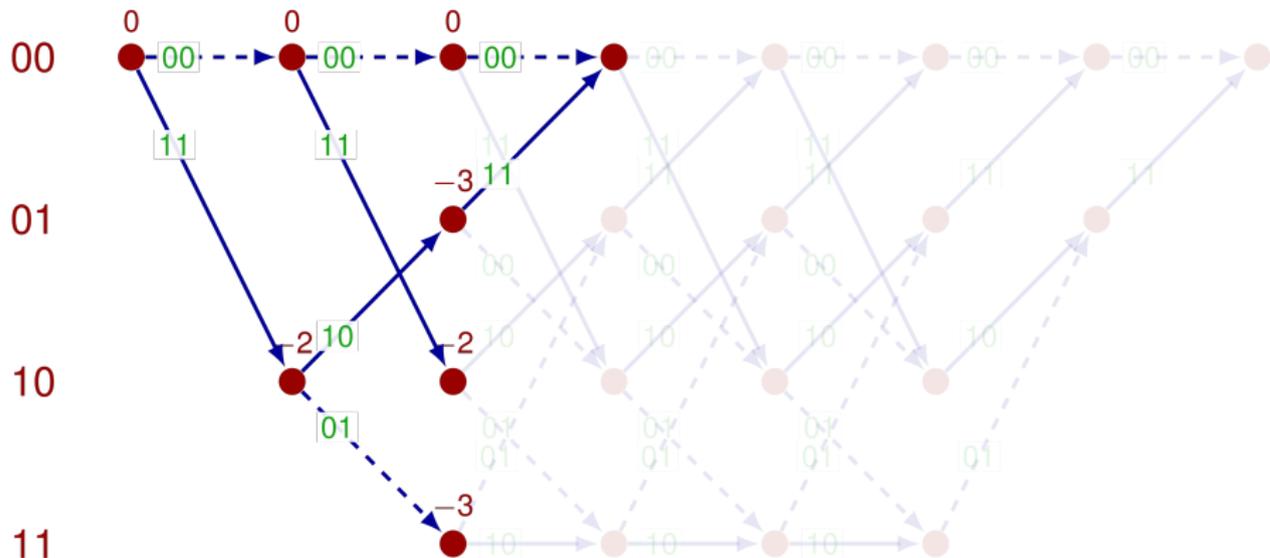
$$\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$$



Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?

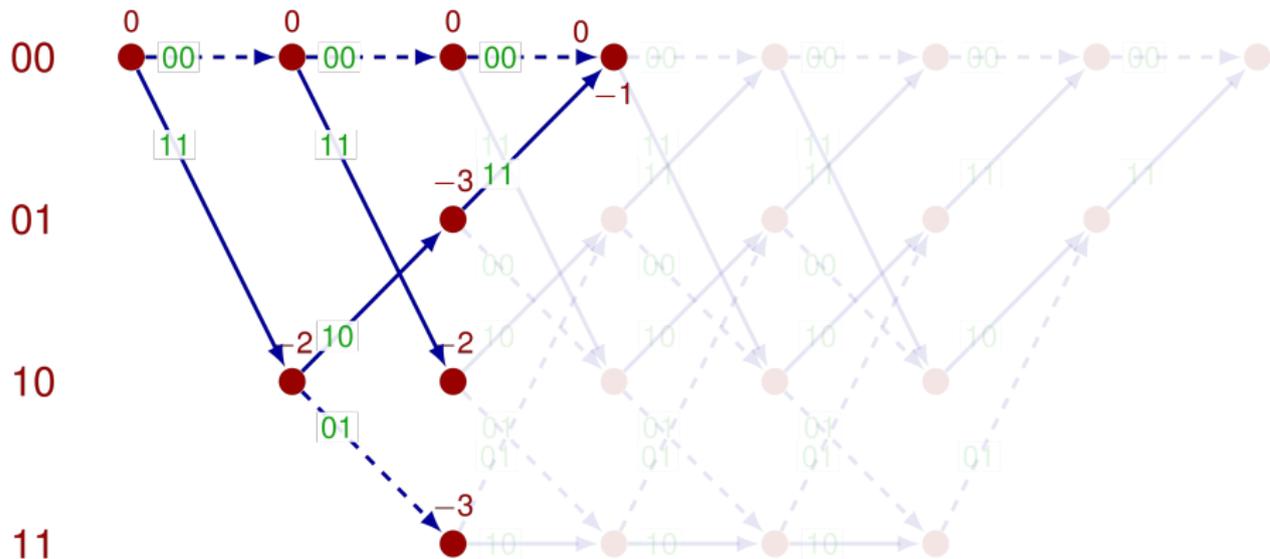
$$\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$$



Exemple de question...

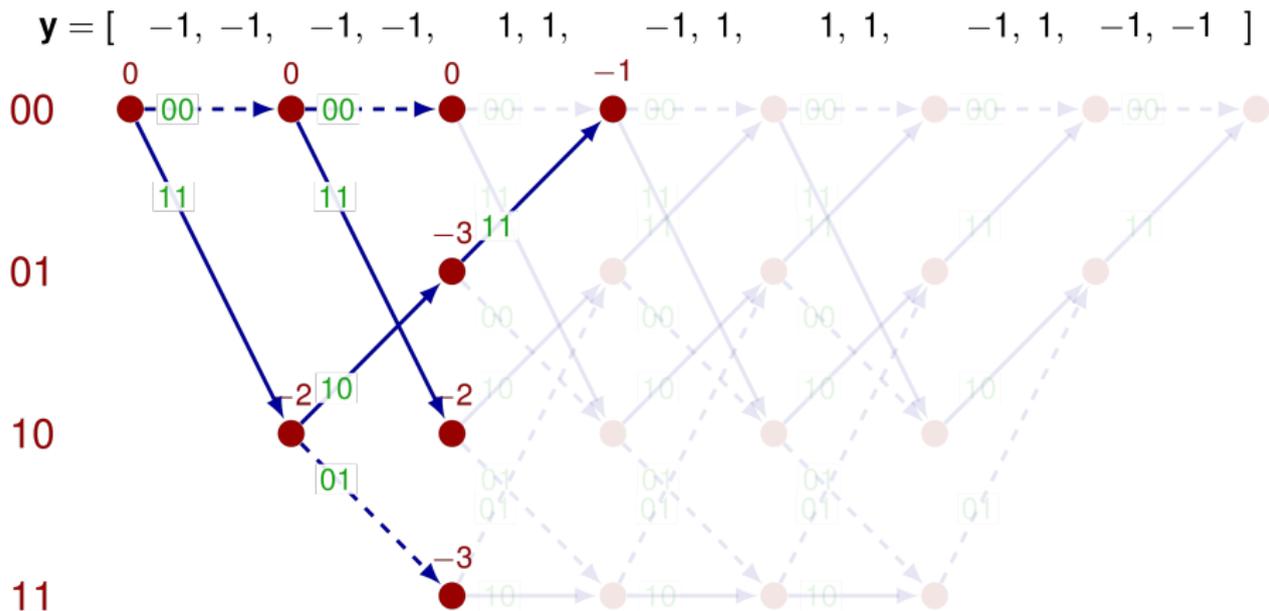
Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?

$$\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$$



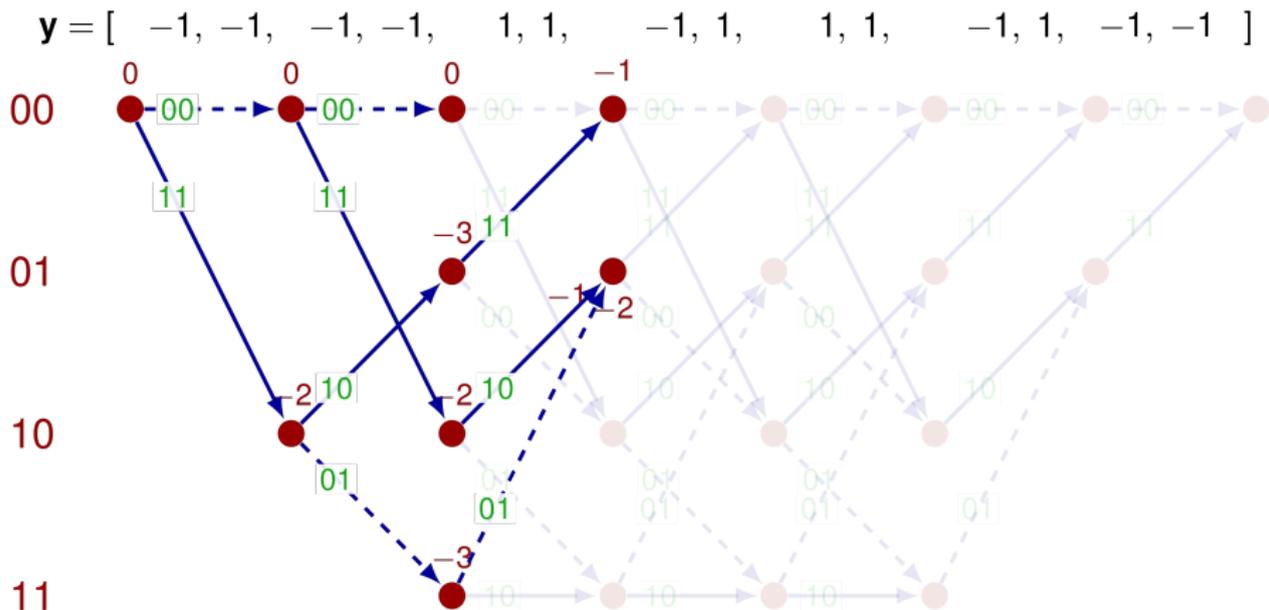
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



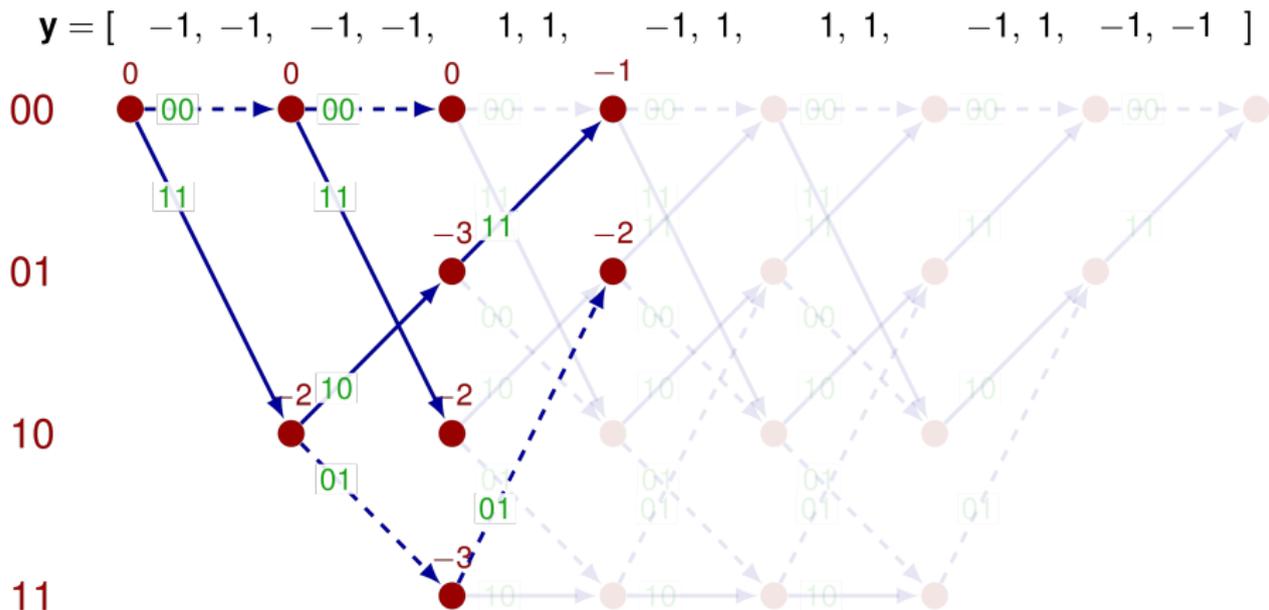
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



Exemple de question...

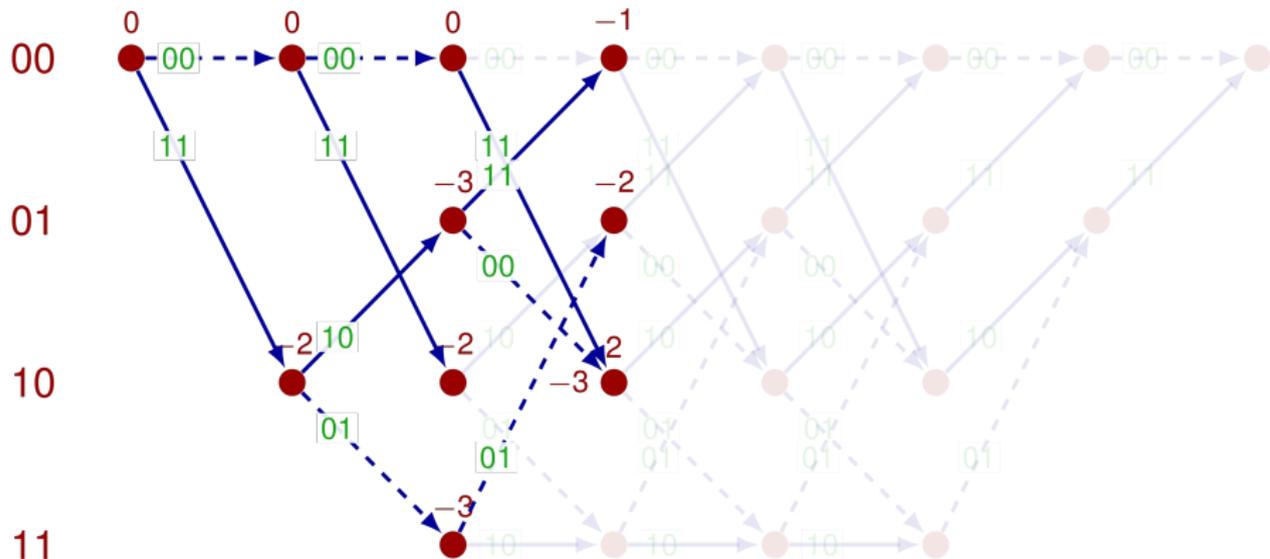
Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



Exemple de question...

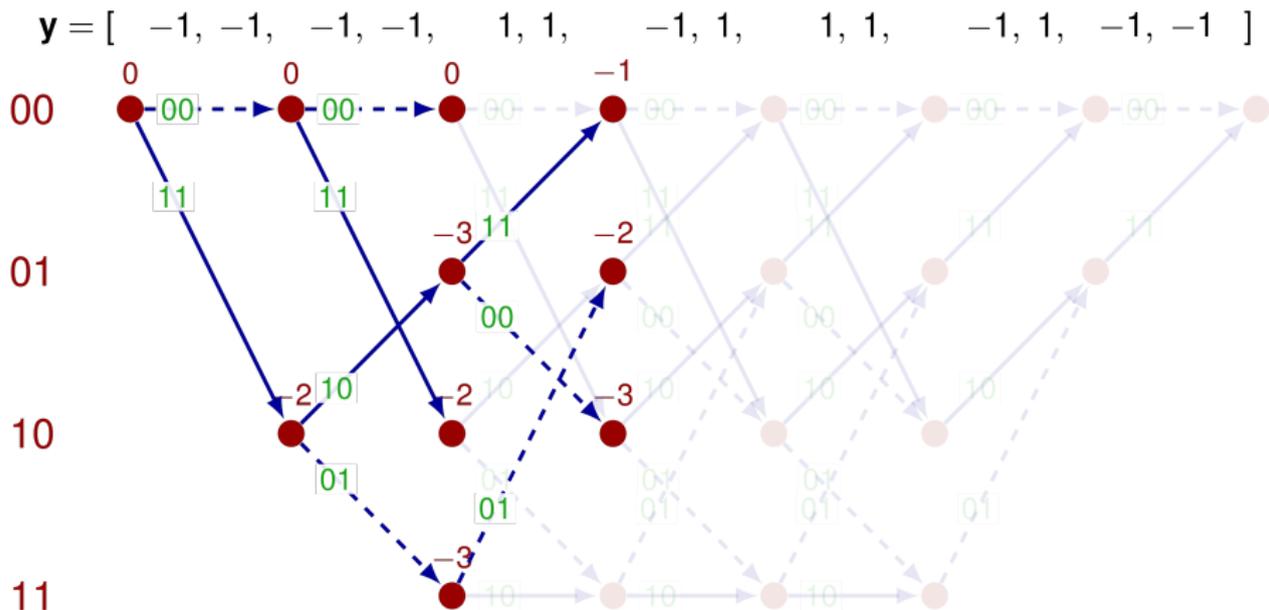
Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?

$$\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$$



Exemple de question...

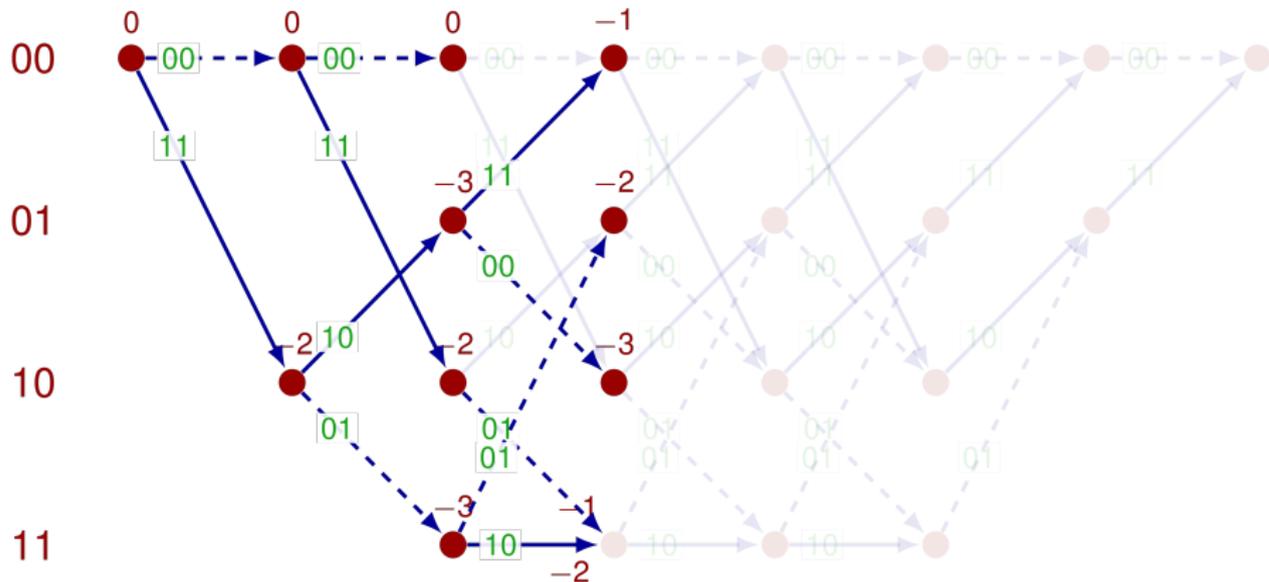
Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



Exemple de question...

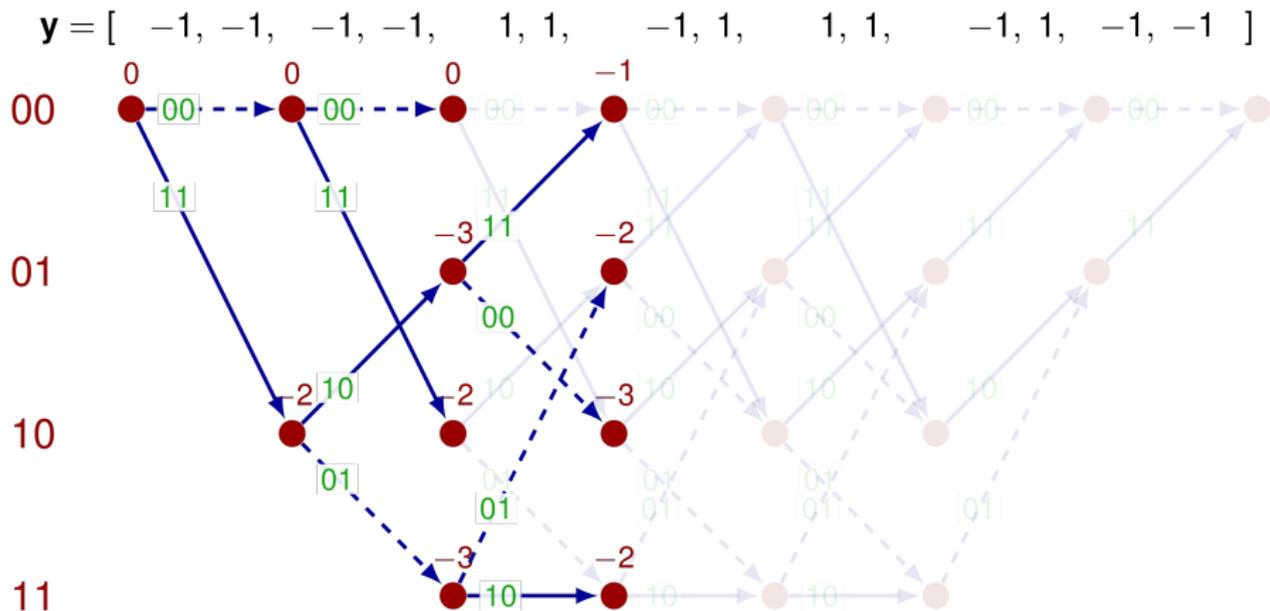
Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?

$$\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$$



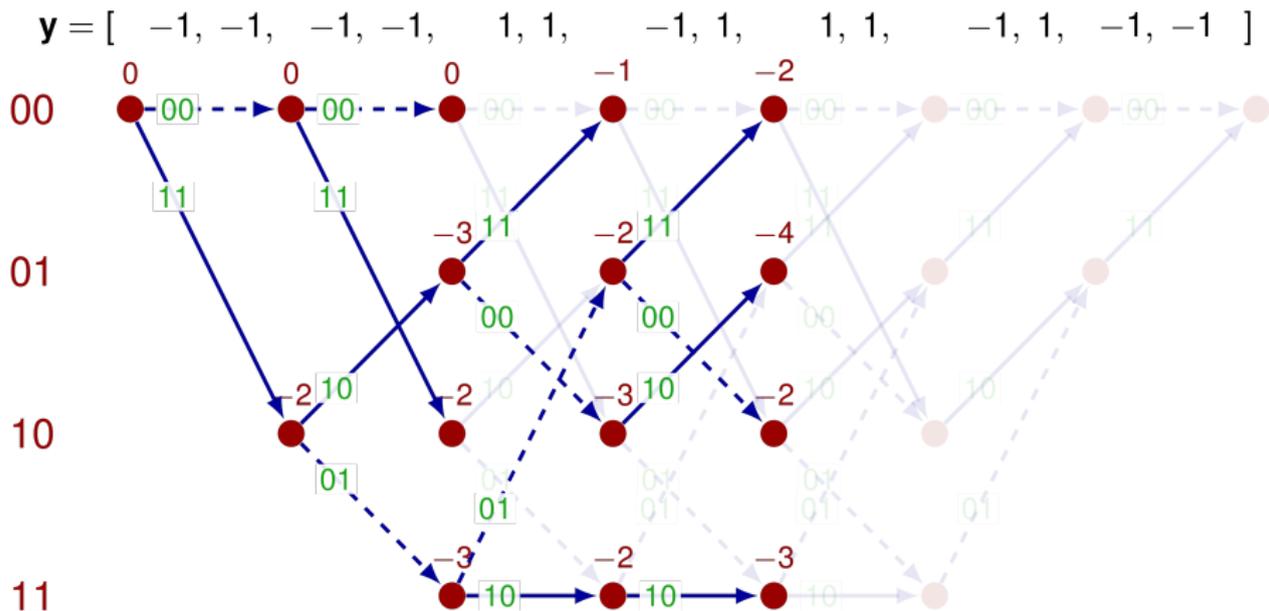
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



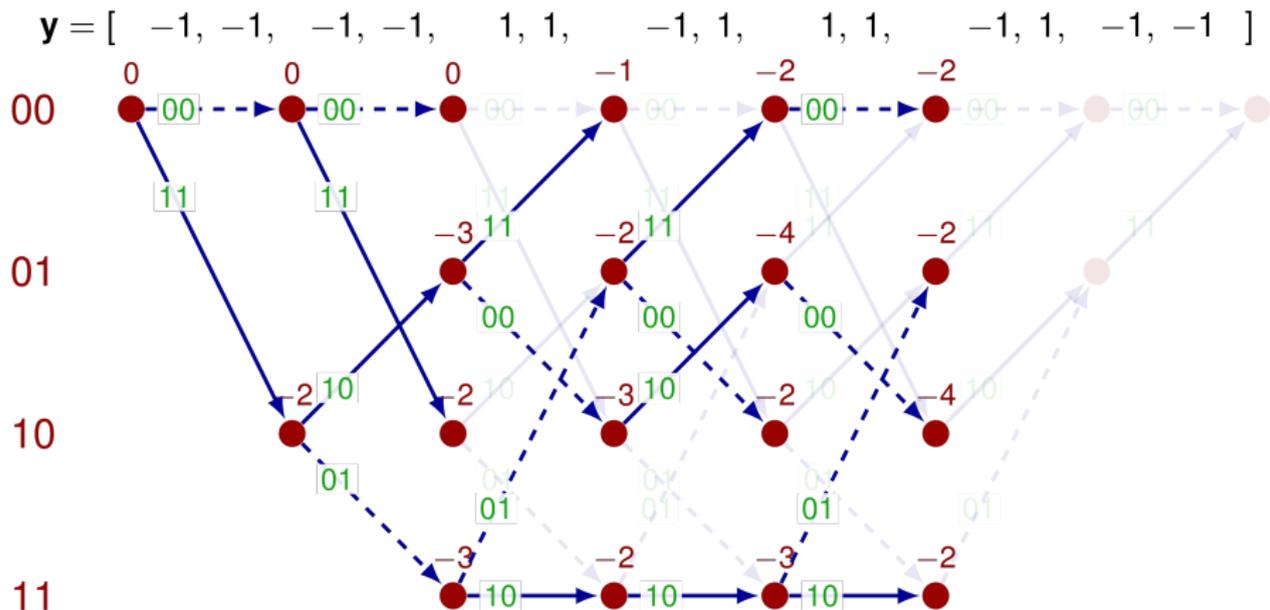
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



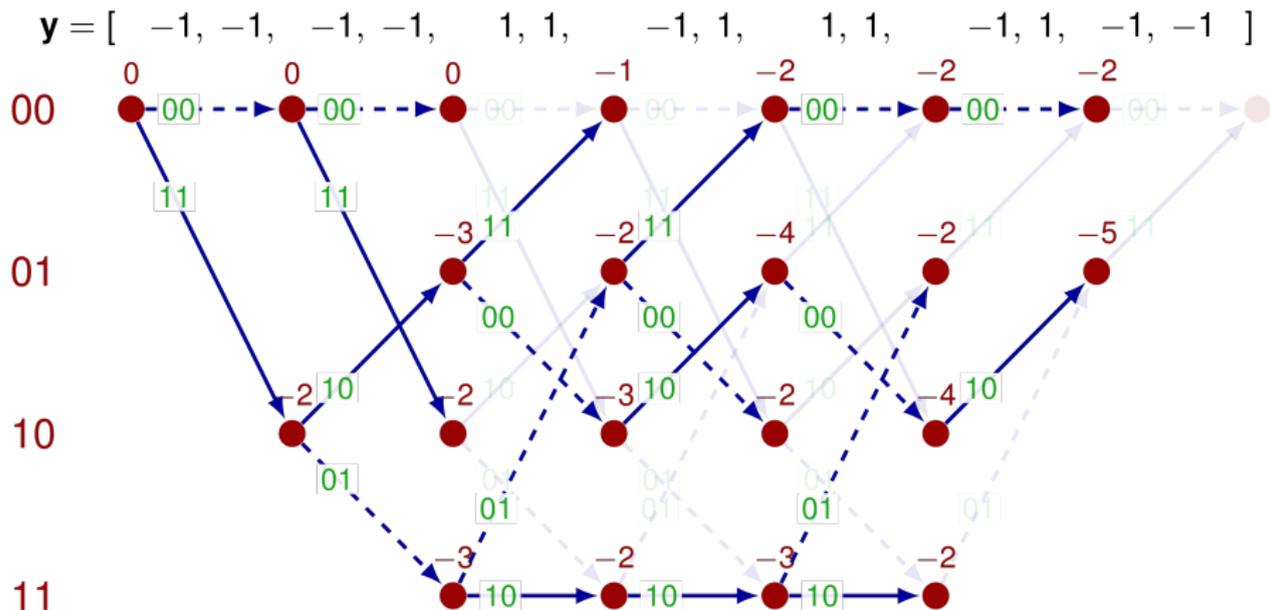
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



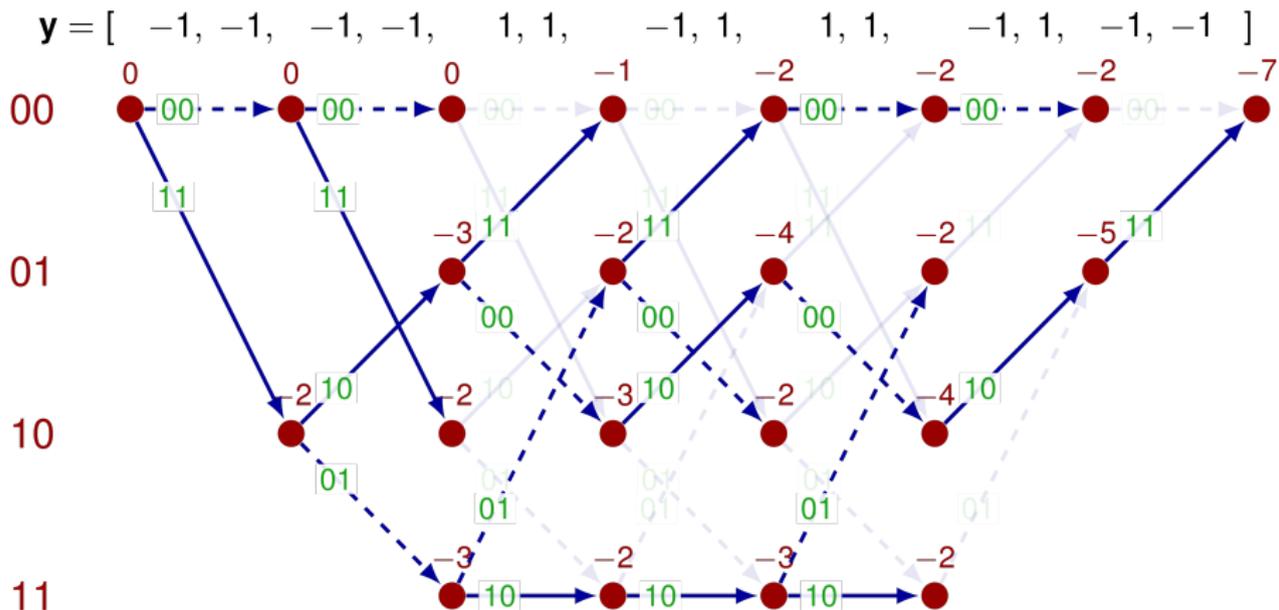
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



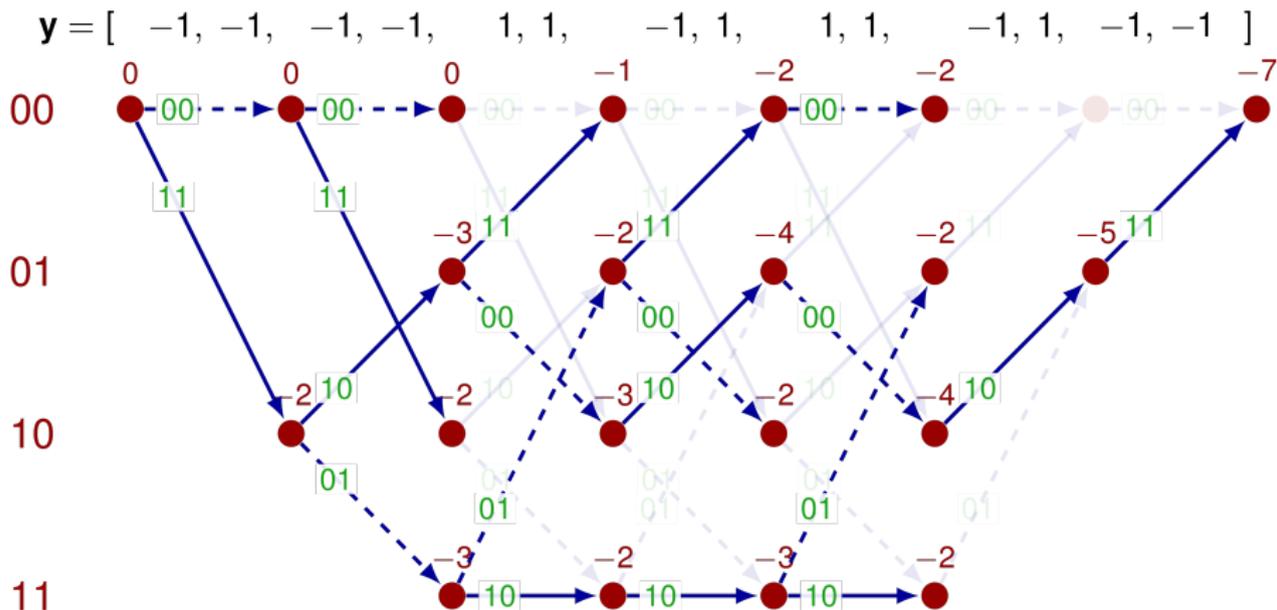
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



Exemple de question...

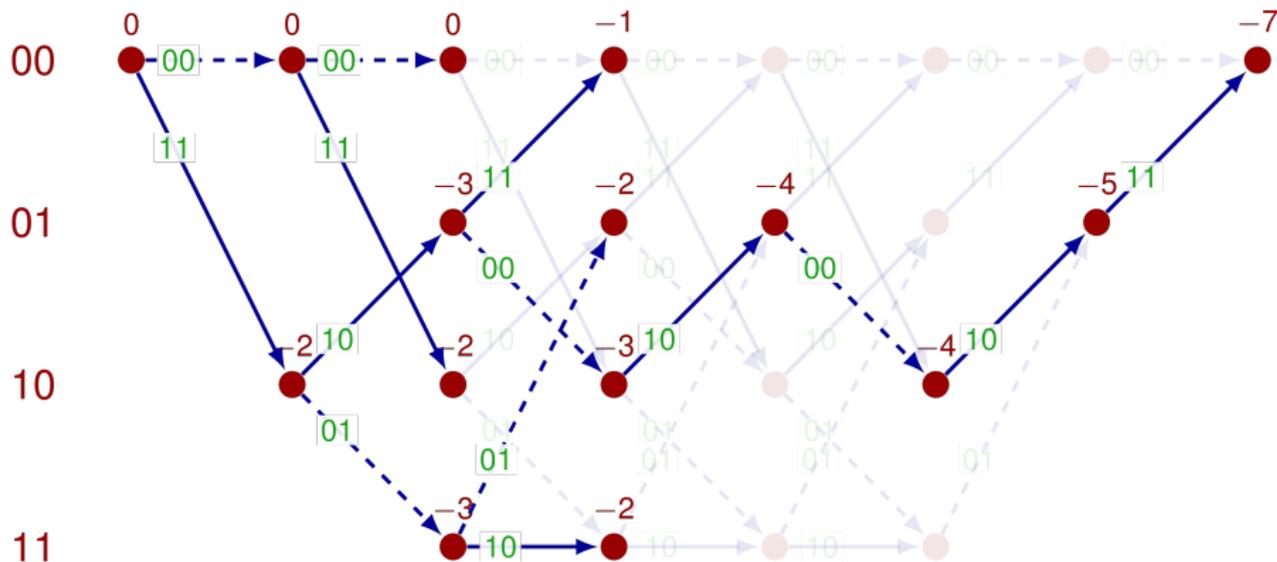
Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



Exemple de question...

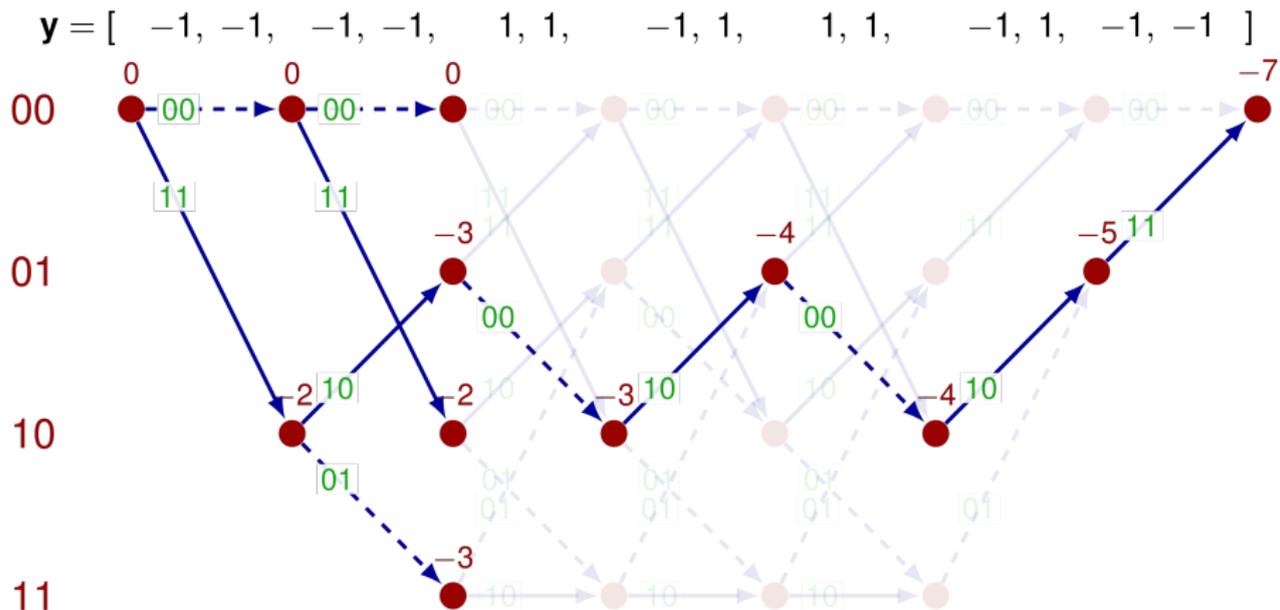
Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?

$$\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$$



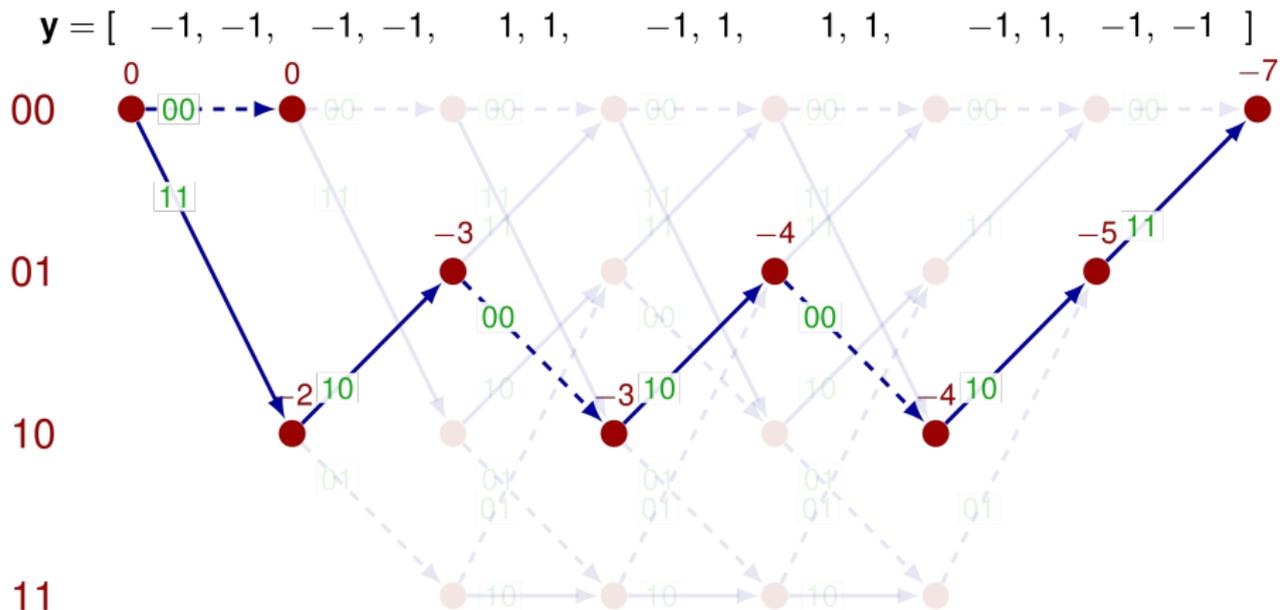
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



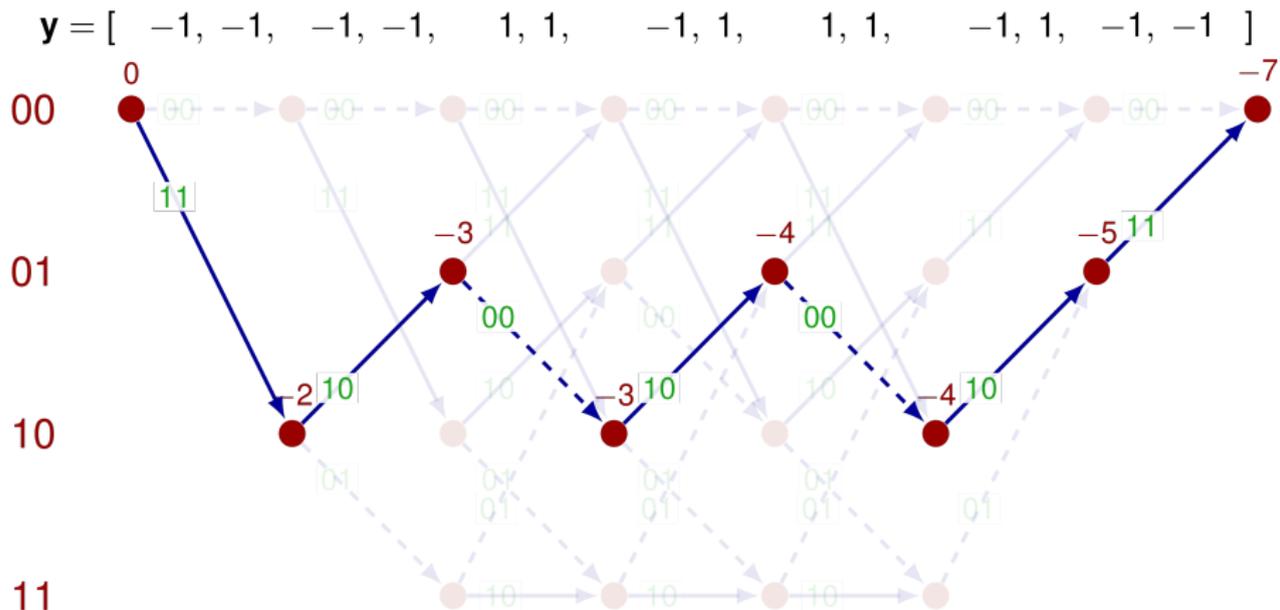
Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



Exemple de question...

Soit le signal observé $\mathbf{y} = [-1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1]$, sachant que $\mathbf{u} = [1, 1, 0, 1, 0]$, reçoit-on le message sans erreur ?



Dernier QCM

Comment avez-vous trouvé ce cours ?

- A Très difficile
- B Difficile
- C Moyen
- D Simple
- E Très simple