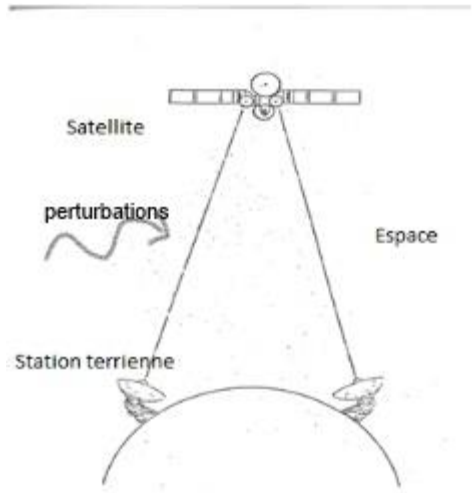
Communication satellitaire et code de Reed-Solomon

Sommaire

- Présentation du problème
- Modélisation
  - Messages
  - Canal binaire symétrique
- Code de Reed-Solomon
  - Codage
  - Principe du décodage
- Discussion des performances

Présentation du problème



Modélisation des messages

- Message = suite finie d'octets

Définition d'un octet

- Un octet est représenté par un élément du corps fini à 256 éléments GF(256)

Implémentation python des octets

- Un octet est représenté par un élément du type numpy.uint8, un entier non signé codé sur 8 bits
  - Par exemple 1010 1110 est représenté par 174

- La somme correspond à un ou exclusif bit à bit, réalisé par la fonction numpy.bitwise_xor

Implémentation python des octets

- Afin de réaliser la multiplication, on génère des tables table_exp et table_log et des fontions exp et log telles que :
  - $\exp(k) = \alpha^k$ avec α un générateur de $GF(256)^*$
  - $\log(a)$: logarithme discret de a en base α

  Alors, $a*b = \exp(\log(a)+\log(b))$

- On dispose égalment d'une fonction inv telle que $inv(a) = a^{-1}$

Implémentation des messages

- Message = suite finie d'octets => polynôme à coefficient dans GF(256)

Implémentation des messages

- Informatiquement : message = numpy.array de numpy.uint8 (tableau d'octets)
- Somme : somme terme à terme des tableaux
- Produit : on utilise la définition (produit de Cauchy)
- Division euclidienne : on implémente l'algorithme usuel

Modèle du canal bruité

- Canal binaire symétrique sans mémoire:
  - Opère sur des bits
  - A autant de chance de modifier un 0 qu'un 1
  - Probabilités de modifier chaque bit indépendantes

Implémentation du canal binaire symétrique

- Difficulté : il opère sur les bits et on a codé les octets

↳ Il suffit d'ajouter une puissance de 2

Principe des codes correcteurs

- Ajout de redondance pour détecter voire corriger les erreurs
- Exemple : numéro de sécurité sociale suivi d'une clé de contrôle pour détecter les erreurs de copie

Code de Reed-Solomon

- Code de Reed-Solomon = code par blocs
- Code de Reed-Solomon de paramètres n,k,t : RS(n,k,t):
  - Transforme un bloc de k octets en un bloc de n octets en ajoutant 2t octets
- Objectif : implémenter RS(204,188,8)

Codage d'un message

- On appelle générateur du code de Reed-Solomon et on note g(X) le polynôme à coefficients dans GF(256):

$$g(X) = \prod_{i=1}^{2t} (X - \alpha^i)$$

Où α est le générateur de GF(256)

Codage d'un message

- Soit M le message à coder, considéré comme un polynôme à coefficients dans $GF(256)$
- Contrôle de parité : $CK(X) = M(X) * X^{2t} \bmod g(X)$
- Mot codé : $C(X) = M(X) * X^{2t} + CK(X)$

↳ S'implémente facilement

Principe du décodage

- On reçoit le mot $R = C + E$: C mot de code et E mot d'erreur
- On calcule les syndromes $S_i = R(\alpha^i)$ $(i=1,\ldots,2t)$
- Implémentation : algorithme de Horner

Principe du décodage

- $S_i = R(\alpha^i) = C(\alpha^i) + E(\alpha^i) = E(\alpha^i)$ car g(X) divise C(X) et α^i annule g

- On a le système : $S_i = \sum_{k=1}^{\nu} Y_k X_k^i$ pour i = 1, 2,..., 2t

  $X_i = \alpha^{j_i}$: position de l'erreur i

  Y_i : valeur de l'erreur i

  ν : nombre d'erreurs commises

- On cherche le polynôme P ayant le moins de coefficients tel que $P(\alpha^i) = S_i$ pour i = 1,..., 2t

Principe du décodage

- Algorithme PGZ (Peterson, Gorenstein, Zierler):
  - Calcul des syndromes
  - Algorithme Euclidien : détermination de $\sigma(X)$ et de $\omega(X)$
  - Recherche de Chien : détermination de la localisation des erreurs
  - Algorithme de Forney : détermination de la valeur des erreurs

Choix des paramètres de RS(204,188,8)

- Taux d'information de RS(n,k,t): $\tau = k/n$
- Pour optimiser τ, il faut un grand k
- Mais, délais de codage et décodage non pris en compte

Choix des paramètres de RS(204,188,8)

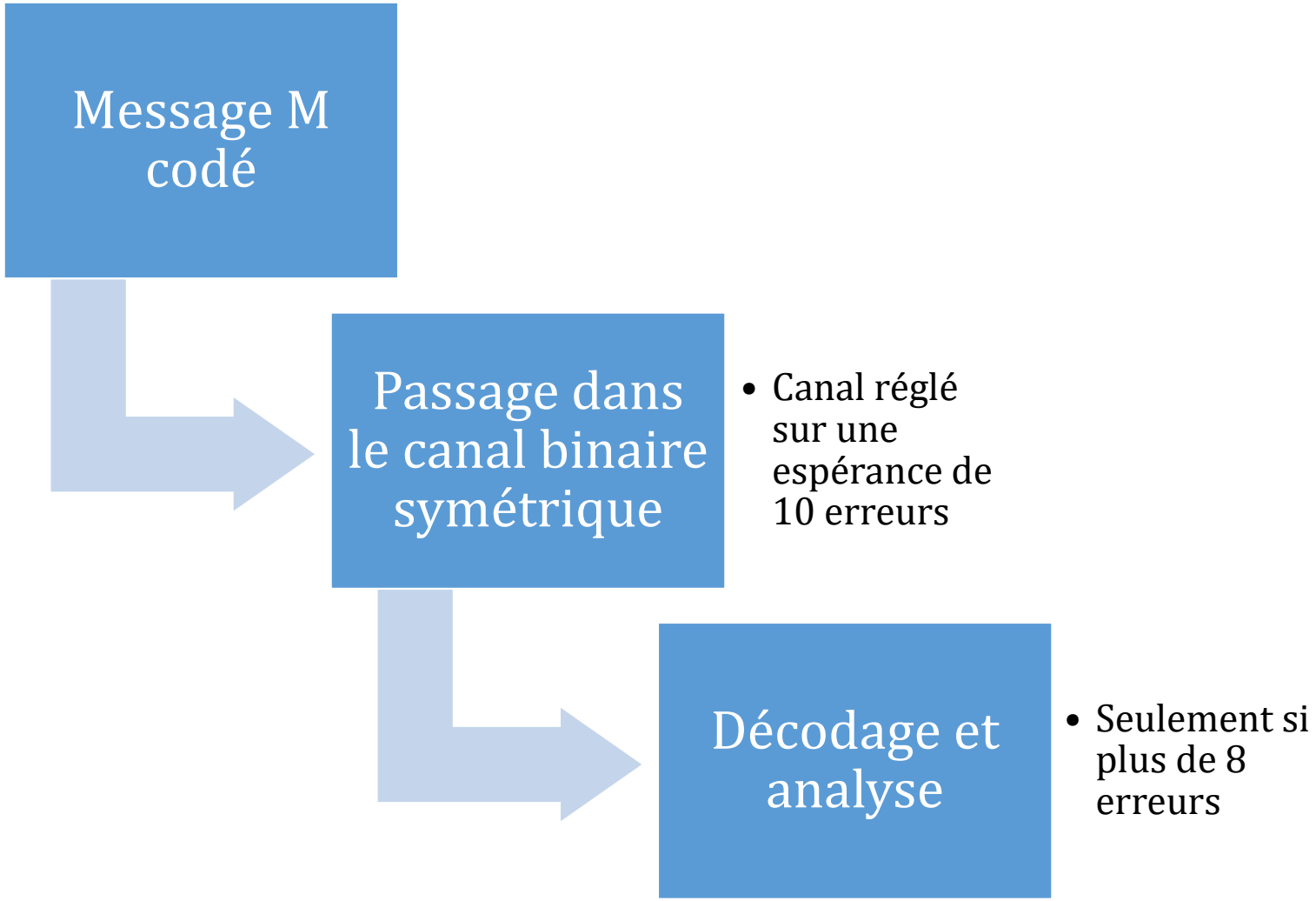
	RS(204,188)	RS(255,239)
Codage (s)	4	7
Syndromes (s)	0,08	0,1
Algorithme euclidien (s)	0,01	0,02
Chien search (s)	0,02	0,03
Algorithme de Forney (s)	0,001	0,002
Calcul de l'erreur (s)	0,004	0,007
Total décodage (s)	0,1	0,15
Total (s)	**4,1**	**7,15**

- k = 188 représente un compromis et maximise le débit d'information

Performances de la correction d'erreurs

- RS(204,188) traite des octets pas des bits => plusieurs erreurs sur des bits d'un même octet comptent pour 1 erreur
- corrige parfaitement 8 erreurs et moins

Performances de la correction d'erreurs



Performances de la correction d'erreurs

- Résultats pour 10 000 erreurs testées:
  - L'erreur introduite n'est jamais entièrement corrigée
  - Dans 10% des cas, l'erreur est partiellement corrigée
  - Des erreurs sont tout le temps ajoutées

Conclusion

- Le code RS(204,188) présente de nombreux avantages:
  - Il est performant pour la correction d'erreurs
  - Il permet un haut débit d'information
  - Il s'implémente efficacement avec des registres à décalage

  => Peu de contraintes à l'utilisation

↳ Il est donc particulièrement adapté à la communication satellitaire