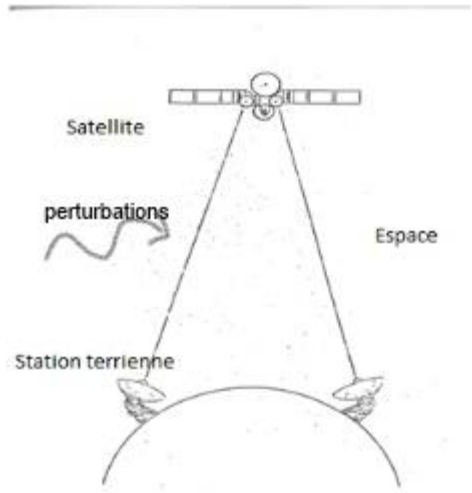
Communication satellitaire et code de Reed-Solomon

Présentation du problème



Modélisation des messages

- Message = suite de bits
- On regroupe les bits par octets
- On découpe le message en blocs qui sont codés puis envoyés au travers d'un canal bruité

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

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 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
  - On a donc $n = k + 2t$
- Objectif : implémenter RS(204,188,8)

Définition d'un octet

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

- Construction : soit $P_0 = X^8+X^4+X^3+X^2+1 \in GF(2)[X]$ un polynôme irréductible et de degré 8.

↳ L'ensemble des classes d'équivalence pour la congruence modulo P_0 est un corps à 256 éléments.

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 $inv(a) = a^{-1}$

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 : $C(X) = M(X) * X^{2t} + CK(X)$

Codage d'un message

- On remarque que :
  - Le code est systématique ie le message constitue les premiers octets du mot codé
  - g(X) divise tout mot du code

Implémentation python du codage

- Les messages, donc les polynômes sur GF(256), sont représentés par des numpy.array d'octets
- On code la somme en sommant les octets terme à terme
- On code le produit à l'aide de la définition du produit de polynômes
- On code la division euclidienne à l'aide de l'algorithme usuel

Principe du décodage

- On reçoit le mot $R = C + E$: C mot de code et E mot d'erreur
- On calcule les syndromes $S_i = R(\alpha^i)$
- Implémentation : algorithme de Horner pour l'évaluation des polynômes (complexité : $O(n)$)

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

Performances 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 complexité temporelle du codage et du décodage: $O(k^2)$: limite le débit d'octets.

Performances 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
- Décode parfaitement 8 erreurs et moins

Performances de la correction d'erreurs

- Si plus de 8 erreurs (tests sur 10 000 erreurs aléatoires):
  - 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 ne dilue pas beaucoup l'information
  - Il s'implémente en pratique avec des registres à décalage

  => ajoute peu de temps de calcul

↳ Il est donc particulièrement adapté à la communication satellitaire