Résumé des fonctions réalisées à la couche liaison

1. Découpage du flot de bits en trames
2. Contrôle d'erreurs
3. Retransmission de trames erronées
   - Acquittements
   - Numéros de séquence
   - Stratégies ARQ
4. Établissement et terminaison de connexions

2. Contrôle d'erreurs

Problème

- Les protocoles de la couche physique ne sont pas parfaits mais subissent des erreurs bit
- Le taux d'erreur bit dépend surtout du média de transmission
  - Fibres optiques: ~ 10^{-12}
  - Canal radio: ~ 10^{-5}

La couche liaison implémente deux fonctions

- Détection d'erreurs (obligatoire)
  - Codes détecteurs
- Correction d'erreurs (seulement pour un service fiable)
  - Code correcteurs
  - Retransmission de trames

Critères d'efficacité d'un contrôle d'erreur

1. Capacité de détecter / corriger des erreurs multiples
2. Capacité de détecter / corriger des rafales d'erreurs d'une certaine longueur
3. Probabilité d'accepter une trame erronée comme correcte
4. Rendement:
   rapport entre les bits de données et la longueur totale des paquets

Codes simples: les parités

Méthode la plus simple de détection d'erreurs

- Ajouter un seul bit de parité à la fin d'un bloc de données
  - Parité paire :
    le nombre total de bits 1 (y compris la parité) est pair
  - Parité impaire :
    le nombre total de bits 1 (y compris la parité) est impair

➢ Code détecteur d'erreurs
  - Toutes les erreurs bit sur un nombre impair de bits sont détectables pour une longueur de données quelconque
  - En pratique, seulement la moitié des erreurs bit sont détectées
    - La plupart des erreurs bit apparaissent en rafale, donc un nombre pair d'erreurs est aussi probable qu'un nombre impair

Parités verticales et horizontales

- Permettent de construire un code correcteur simple
  - Arranger la séquence des bits en une matrice
  - Calculer une parité par ligne et par colonne

1	0	0	1	0	1	0	1
0	1	1	1	0	1	0	0
1	1	1	0	0	0	1	0
1	0	0	0	1	1	1	0
0	0	1	1	0	0	1	1
1	0	1	1	1	1	1	0

Parités horizontales

Parités verticales

Exemple

- Codez la séquence 010010110101 en utilisant la technique de parité croisée (verticale-horizontale) sachant que le mot à coder est divisé en blocs de 4 bits
- Introduisez une erreur lors de la transmission vers le système destinataire
- Corrigez l'erreur

Exercice 1

- Codez la séquence 111001001011010 en utilisant la technique de parité croisée (verticale-horizontale) sachant que le mot à coder est divisé en blocs de 5 bits
- Introduisez une erreur lors de la transmission vers le système destinataire
- Corrigez l'erreur

Exercice 2

- La séquence suivante est reçue par la couche 2 dans le système destinataire:
- 11101100101101110110
- Sachant que le codage au niveau de la station émettrice a été fait en utilisant la technique de parité croisée (verticale-horizontale) avec le mot à coder divisé en blocs de 4 bits, analysez la séquence par rapport aux erreurs éventuelles

Parités verticales et horizontales

- Permettent de corriger toutes les erreurs simples
- Permettent de détecter toutes les erreurs sur 2 ou 3 bits
- Permettent de détecteur toutes les rafales plus courtes *ou égales* que la longueur d'une ligne
- Déjà 4 erreurs bit peuvent passer sans être détectées

1	0	0	1	0	1	0	1
0	1	1	1	0	1	0	0
1	1	**0**	0	**1**	0	1	0
1	0	0	0	1	1	1	0
0	0	**0**	1	**1**	0	1	1
1	0	1	1	1	1	1	0

Parités horizontales

Parités verticales

HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

Codes correcteurs: Bases de la théorie de codage

- Distance de Hamming :
  nombre de bits différents des deux mots de code
  - Pour détecter *e* erreurs de bit il faut que le code ait une distance de Hamming minimale de *e*+1
  - Pour corriger *e* erreurs de bit le code doit avoir une distance de *2e+1*
- Combien de bits de contrôle faut il pour un code correcteur d'erreurs simples ?

$$(m+r+1) \leq 2^r$$

Codes de Hamming

- Vecteur des $m=4$ bits de données: x^T
- Vecteur des $n=7$ bits du mot de code: y^T
  - Il y a donc $r=3$ bits de contrôle

Matrice de contrôle: H

- Dimensions de H: $r \times n = 3 \text{ x } 7$

$$H = \begin{vmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{vmatrix}$$

Codes de Hamming

- *Pour générer le mot code y, on multiplie H par y^T et on force le résultat à être* $[0 \quad 0 \quad 0]$
- $y = [{\color{red}C_1} \quad {\color{red}C_2} \quad C_3 \quad {\color{red}C_4} \quad C_5 \quad C_6 \quad C_7]$

r bits de contrôle

$$\begin{vmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{vmatrix} \begin{bmatrix} {\color{red}C_1} \\ {\color{red}C_2} \\ C_3 \\ {\color{red}C_4} \\ C_5 \\ C_6 \\ C_7 \end{bmatrix} = [0 \quad 0 \quad 0]$$

Codes de Hamming

- On obtient

$$C_4 + C_5 + C_6 + C_7 = 0$$

$$C_2 + C_3 + C_6 + C_7 = 0$$

$$C_1 + C_3 + C_5 + C_7 = 0$$

- Mettant en évidence (modulo-2) les bits de contrôle:

$$C_4 = C_5 + C_6 + C_7$$

$$C_2 = C_3 + C_6 + C_7$$

$$C_1 = C_3 + C_5 + C_7$$

Exemple de codage Hamming

- Le mot-information est 0001. Donc

$$C_3 = 0 \qquad C_5 = 0 \qquad C_6 = 0 \qquad C_7 = 1$$

- Les bits de contrôle sont:

$$C_4 = C_5 + C_6 + C_7 = 0 + 0 + 1 = 1$$

$$C_2 = C_3 + C_6 + C_7 = 0 + 0 + 1 = 1$$

$$C_1 = C_3 + C_5 + C_7 = 0 + 0 + 1 = 1$$

- Le mot-code est: $[1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1]$

Décodage

- Obtenir le syndrome *S* en multipliant la matrice de contrôle par le mot-code reçu y':

$$S = Hy'^T$$

$$\begin{bmatrix} s_1 & s_2 & s_3 \end{bmatrix} = \begin{vmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{vmatrix} \begin{bmatrix} {\color{red}C'_1} \\ {\color{red}C'_2} \\ C'_3 \\ {\color{red}C'_4} \\ C'_5 \\ C'_6 \\ C'_7 \end{bmatrix}$$

Décodage

- Si le syndrome est $[0 \quad 0 \quad 0]$, il n y'a pas eu d'erreur (on suppose que le nombre max d'erreurs est 1.
- Si le syndrome est

$$[s_1 \quad s_2 \quad s_3] \neq [0 \quad 0 \quad 0]$$

- Les bits $[s_1 \quad s_2 \quad s_3]$ pointent vers la position de l'erreur

Exemple de décodage

- Le mot-code transmis est: [1 1 0 1 0 0 1]
- On suppose qu'une erreur au bit 5:

  [1 1 0 1 1 0 1]

- Le syndrome est

$$\begin{vmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{vmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} = [1 \quad 0 \quad 1]$$

- C'est donc le 5$^{\text{ème}}$ bit qui est erroné (101 est égale à 5 en décimal)

Correction de rafales d'erreurs avec un code de Hamming

- Arrangement des mots de données en matrice
- Calcul des bits de contrôle par colonne

Sens de la transmission

1	0	0	1	0	1	0
0	1	1	1	0	1	0
1	1	1	0	0	0	1
1	0	0	0	1	1	1
0	0	1	1	0	0	1
1	0	1	1	1	1	1
0	1	0	0	0	1	1
1	0	1	1	0	1	0
0	0	1	0	1	0	0

Bits de données

Rafale d'erreurs

Bits de contrôle par colonne

Efficacité des codes correcteurs

- Rendement d'un code : $R = m/(m+r)$
- Rendement des codes de Hamming :

$$R = \frac{n-r}{n} = 1 - \frac{\log_2(n+1)}{n}$$

- Exemple :
  - Mot de données sur 4 bits
  - 3 bits de contrôle sont nécessaires
  - Rendement R = 57%

Code détecteurs d'erreurs

- Codes polynomiaux ou codes cycliques (CRC)
  - Permettent de contrôler des mots d'une longueur variable
  - Réalisation simple et efficace en matériel
  - Très bonnes capacités de détection d'erreurs

Principe des codes polynomiaux

- Un bloc de *m* bits est vu comme un polynôme M(x) de degré *m-1* avec des coefficients binaires
  - Exemple : 110001 → $M(x) = x^5+x^4+x^0$
- Polynôme générateur : *G(x)* de degré *r*
- Le mot de code correspond à un polynôme T(x) de degré m-1+r avec T(x) / G(x) = 0

Un code polynomial est un code dans lequel tous les mots de code, représentés par des polynômes T(x), sont des multiples d'un polynôme générateur G(x)

Calcul d'un mot de code

1. Ajouter r zéros après le bit de poids faible du bloc Il contient ainsi $m + r$ bits, correspondant au polynôme $x^r M(x)$
2. Division modulo 2 du polynôme $x^r M(x)$ par $G(x)$
3. Soustraire modulo 2 le reste de la division de la chaîne de bits correspondant au polynôme $x^r M(x)$
4. Le résultat de cette opération est la trame à transmettre

Exemple

- Trame: 1101011011
- Générateur: 10011

  → $G(x) = x^4 + x + 1$

```
11010110110000 | 10011
10011          |-----------
-----          | 1100001010
 10011
 10011
 -----
  00001
  00000
  -----
   00010
   00000
   -----
    00101
    00000
    -----
     01011
     00000
     -----
      10110
      10011
      -----
       01010
       00000
       -----
        10100
        10011
        -----
         01110
         00000
         -----
          1110   Reste
```

➢ Trame transmises:

1101011011111**0**

Original: Jürgen Ehrensberger. Modifications: Marcos Rubinstein

Détection d'erreurs

- Le récepteur calcule $T(x) / G(x)$
- Il accepte la trame comme correcte si le reste est 0

- En cas d'erreurs bit, la trame reçue correspond à un polynôme $T'(x) = T(x) + E(x)$
- Une erreur est détectée si le reste de $E(x)/G(x)$ n'est pas nul

Capacité de détection d'erreurs

1. Toute erreur simple est détectée
   – Si le polynôme générateur G(x) comporte plus d'un coefficient non nul
2. Les erreurs doubles sont toutes détectées
   – Si le polynôme générateur *G(x)* ne divise pas $x^k + 1$, où k peut prendre n'importe quelle valeur comprise entre 1 et n-1
3. Une erreur comportant un nombre impair d'erreurs est détectée
   – Si le polynôme générateur comporte (x + 1) en facteur
4. Toutes les rafales d'erreurs de longueur r sont détectées
   – Pour un polynôme générateur G(x) de degré r
5. Les rafales d'erreurs d'une longueur supérieure à r sont détectées avec un probabilité de $1-(1/2^{r-1})$

Polynômes générateurs normalisés

- CRC-12 : $x^{12} + x^{11} + x^{3} + x^{2} + x^{1} + 1$
- CRC-16 : $x^{16} + x^{15} + x^{2} + 1$
- CRC-CCITT : $x^{16} + x^{12} + x^{5} + 1$
- CRC-32 : $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x^{1} + 1$
- Le CRC-32 est utilisé pour Ethernet
  - Détecte toutes les rafales d'erreurs de 32 bits
  - Probabilité qu'une rafale plus longue ne soit pas détectée: $P = 4{,}6 \cdot 10^{-10}$