Chapitre 3 : Protection contre les erreurs

Z:\Polys\Internet de base (pour L2 Electronique)\3.ProtectionContreErreurs.fm - 7 janvier 2007 14:02

Plan

Bibliographie

- K. Lahèche, Les codes en informatique : codes détecteurs et correcteurs d'erreurs, Hermès, 1995.
- H. Nussbaumer, Téléinformatique - tome 1, Presses polytechniques romandes, 1983.
- C. Macchi, J-F. Guibert, Téléinformatique, Dunod, 1987.
- A.Tanenbaum, Réseaux, InterEditions, 1997.

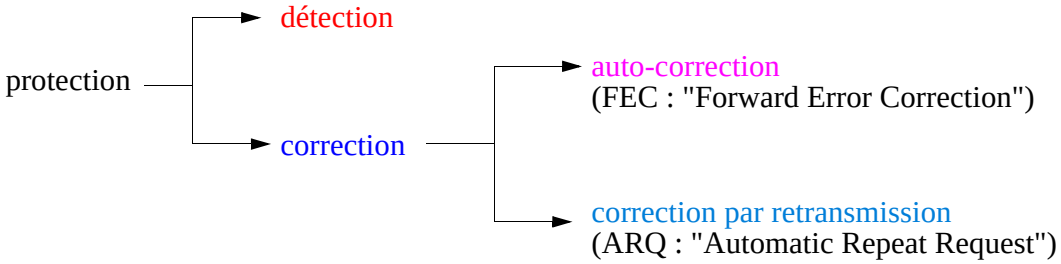
1. Introduction

Quelque soit la qualité des supports de communication et les performances des techniques de transmission utilisées, des perturbations vont se produire entraînant des erreurs sur les données transmises.

Dans ces conditions, la suite binaire reçue ne sera pas identique à la suite émise.

Mise en oeuvre de techniques de protection contre les erreurs de transmission

Stratégies de protection contre les erreurs de transmission :



1.1. Principe général pour la détection des erreurs de transmission :

- Un émetteur veut transmettre un message (suite binaire quelconque) à un récepteur.
- L'émetteur transforme le message initial à l'aide d'un procédé de calcul spécifique qui génère une certaine redondance des informations au sein du message codé.
- Le récepteur vérifie à l'aide du même procédé de calcul que le message reçu est bien le message envoyé grâce à ces redondances.

• Exemple : la technique de détection par répétition.
  . le message codé est un double exemplaire du message initial, le récepteur sait qu'il y a eu erreur si les exemplaires ne sont pas identiques.

• Note : certaines erreurs sont indétectables !
  . ex. : une même erreur sur les deux exemplaires simultanément.

1.2. Principe général pour la correction des erreurs de transmission :

- Après détection d'une erreur, la redondance est suffisante pour permettre de retrouver le message initial.

• Exemple : la technique de correction par répétition.
  . le message codé est un triple exemplaire du message initial, le récepteur suppose que le message initial correspond aux deux exemplaires qui sont identiques.

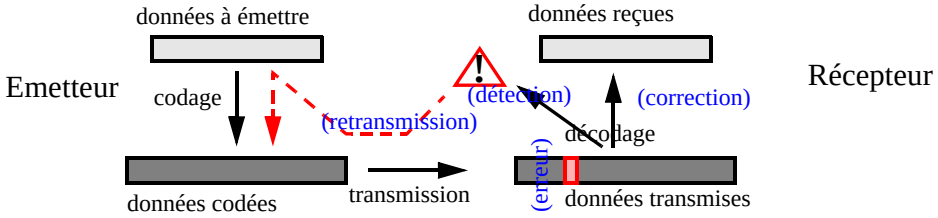
• Note : certaines erreurs détectées ne sont pas corrigibles !
  . ex. : une erreur différente sur au moins deux exemplaires.

• Note : certaines erreurs sont détectées et mal corrigées !
  . ex. : une même erreur sur deux exemplaires simultanément.

1.3. Principe général pour la correction par retransmission des erreurs de transmission :

- Après détection d'une erreur, le récepteur demande à l'émetteur, implicitement (temporisateur) ou explicitement (nack), de retransmettre une nouvelle fois le message (codé).

• Exemple : de très nombreux protocoles de télécommunication : HDLC, X25, TCP, TP.



1.4. Conclusion

La correction par retransmission est préférée dans les réseaux où le taux de perte est faible et le délai de retransmission tolérable, car son surcoût est généralement plus faible que celui induit par les codes auto-correcteurs.

Estimation du surcoût (message de longueur moyenne = 1000 bits) :

- taux d'erreur typique = 10^{-9}, taux de retransmission $\Rightarrow$ 1000. $10^{-9} = 10^{-6}$
- surcoût typique d'un code auto-correcteur : qq octets par message $\Rightarrow$ 8/1000 = 10^{-2}

2. Les codes de protection contre les erreurs

2.1. Classification des codes

Deux grandes familles de codes :

• les codes en bloc (linéaires, cycliques ou non) : le codage/décodage d'un bloc dépend uniquement des informations de ce bloc.

• les codes en treillis (convolutifs, récursifs ou non) : le codage/décodage d'un bloc dépend des informations d'autres blocs (généralement de blocs précédemment transmis).

- un code convolutif s'applique sur une suite infinie de symboles et produit une suite infinie.

On préfère généralement le codage par bloc dans les applications téléinformatiques classiques :

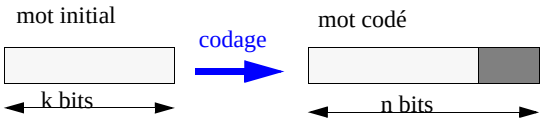
- le codage/décodage est plus simple et il y a moins de délai.

Par la suite, on ne va présenter que les codes par bloc :

- codes simples
- codes linéaires, de Hamming
- codes polynômiaux
- codes cycliques

2.2. Définitions générales

Un code (k, n) transforme (code) tout bloc initial de *k* bits d'information en un bloc codé de *n* bits. Le code introduit une redondance puisque n >= k. Le code est systématique si les *k* premiers bits du bloc codé sont égaux aux bits du bloc initial. Alors les *r* (r=n-k) derniers bits forment un champ de contrôle d'erreur. Le rendement d'un code (k, n) est : R = k/n



On appelle mot du code, la suite de *n* bits obtenue après un codage (k, n). Le nombre *n* de bits qui composent un mot du code est appelé la longueur du code. La dimension *k* étant la longueur initiale des mots.

Le poids de Hamming d'un mot est le nombre de bits à 1 qu'il contient. La distance de Hamming entre deux mots de même longueur est définie par le nombre de positions binaires qui diffèrent entre ces deux mots. On l'obtient par le poids de Hamming de la somme vectorielle (c-à-d. bit à bit) des 2 mots. La distance de Hamming d'un code est la distance minimum entre tous les mots du code.

- Exemple :
  - poids_de_Hamming(01001100) = 3
  - distance_de_Hamming(01001100, 01010101) = 3
  - distance_de_Hamming({01001100, 01010101, 00000000}) = min (3, 3, 4) = 3

La capacité de détection (de correction) d'un code est définie par les configurations erronées qu'il est capable de détecter (corriger). Une erreur simple (resp. double, ou d'ordre p) affecte une seule (resp. 2, ou p) position(s) binaire(s) d'un mot.

Pour qu'un code ait une capacité de détection (resp. correction) des erreurs d'ordre *e*, il faut que sa distance de Hamming soit supérieure à *1+e* (resp. *1 + 2e*).

- Exemple :
  - distance =3 $\Rightarrow$ capacité de détection ≤ 2, capacité de correction ≤ 1.

3. Exemples simples de codes par bloc

3.1. Le contrôle de parité

Parité paire (impaire) : le poids de Hamming des mots du code est pair (impair).

C'est un code systématique (k, k+1) dans lequel un bit (le bit de parité) est ajouté au mot initial pour assurer la parité. Son rendement est faible lorsque k est petit.

- Exemple : Transmission de caractères utilisant un code de représentation (le code ASCII sur 7 bits).

Lettre	Code ASCII	Mot codé (parité paire)	Mode codé (parité impaire)
E	1 0 1 0 0 0 1	1 0 1 0 0 0 1 1	1 0 1 0 0 0 1 0
V	0 1 1 0 1 0 1	0 1 1 0 1 0 1 0	0 1 1 0 1 0 1 1
A	1 0 0 0 0 0 1	1 0 0 0 0 0 1 0	1 0 0 0 0 0 1 1

Ce code est capable de détecter toutes les erreurs en nombre impair. Il ne détecte **pas** les erreurs en nombre pair !

3.2. Parité longitudinale et transversale

Association d'un double codage de la parité :

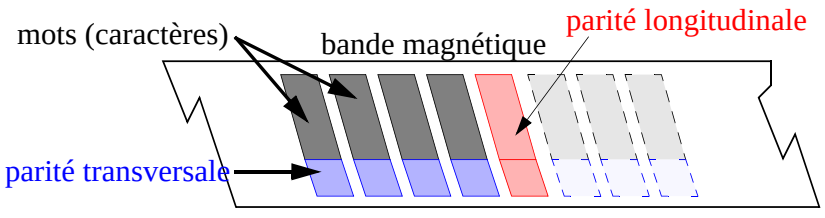
- LRC : "Longitudinal Redundancy Check" et VRC : "Vertical …"

Le bloc de données est disposé sous une forme matricielle (k=a.b). On applique la parité (uniquement paire) sur chaque ligne et chaque colonne. On obtient une matrice [a+1, b+1].

	VRC (parité paire)
	1 0 1 0 0 0 1 1
	0 1 1 0 1 0 1 0
	1 0 0 0 0 0 1 0

LRC =	0 1 0 0 1 0 1 1

Historique de la dénomination :



Le rendement est faible : a.b/ (a+1).(b+1).

Le délai de codage et décodage important : il faut recevoir tout le bloc.

Capacité de détection et d'autocorrection :

- Principe : Une erreur simple modifie simultanément la parité d'un ligne et d'une colonne.
- Correction : inverser le bit situé à l'intersection de la ligne et de la colonne ayant une parité incorrecte.

Exemple :

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

Attention : une erreur triple peut faire croire à une erreur simple et sa correction sera inadaptée !

Exemple :

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

4. Les codes linéaires

4.1. Définitions

Les codes linéaires sont des codes dont chaque mot du code (noté *c*) est obtenu après transformation linéaire des bits du mot initial (noté *i*).

Ces codes sont caractérisés par leur matrice $G_{(k, n)}$ (appelée matrice génératrice) telle que :

$i \cdot G = c$

Exemple : $\begin{bmatrix}1 & 0 & 1\end{bmatrix} \cdot \begin{bmatrix}1 & 0 & 1 & 1\\ 0 & 1 & 0 & 1\\ 1 & 1 & 0 & 1\end{bmatrix} = \begin{bmatrix}0 & 1 & 1 & 0\end{bmatrix}$

La matrice $H_{(n-k, n)}$ (appelée matrice de contrôle) permet de savoir si un mot reçu est un mot du code, en calculant son syndrome. Si le syndrome du mot est nul, ce mot appartient au code.

Syndrome du mot reçu c' : $c' . H^T = 0_{(n-k)} \Leftrightarrow c' \in C_{(k, n)}$

L'équation $G . H^T = 0$ définit la relation entre les deux matrices d'un code.

Preuve : $c' . H^T = i . G . H^T = i . 0$

4.2. Propriétés

La distance de Hamming d'un code linéaire est égale au plus petit poids de Hamming non nul des mots du code. (idée de preuve : la somme vectorielle de 2 mots du code est un mot du code)

Si un code linéaire est systématique, sa matrice génératrice s'écrit :

- $G_{(k, n)} = [Id_{(k)}, P_{(k, n-k)}]$

alors sa matrice de contrôle s'écrit : $H_{(n-k, n)} = [P^T_{(n-k,k)}, Id_{(n-k)}]$

Exemples :

. Soit le code $C_{1(4, 6)}$ de matrice génératrice $G_1 = \begin{bmatrix} 1&0&0&0&1&0 \\ 0&1&0&0&0&1 \\ 0&0&1&0&1&0 \\ 0&0&0&1&0&1 \end{bmatrix}$.

- montrez que le mot de code *c* associé au mot initial *i*= $\begin{bmatrix} 1&0&1&1 \end{bmatrix}$ est le mot c = i.G_1 = $\begin{bmatrix} 1&0&1&1&0&1 \end{bmatrix}$.
- calculez sa matrice de contrôle H_1.

. Le code systématique $C_{2(7, 8)}$ associé à la parité paire a pour matrice de contrôle $H_2^T = \begin{bmatrix} 1\\1\\1\\1\\1\\1\\1\\1 \end{bmatrix}$.

Trouvez la matrice génératrice associé $G_{2(7, 8)}$.

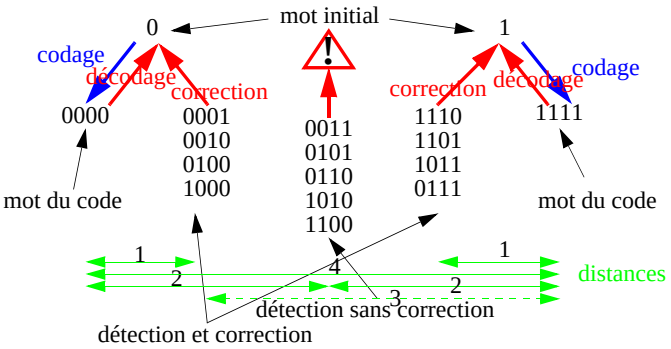
4.3. Correction

Correction par proximité : le procédé de correction transforme un mot reçu et détecté comme erroné dans le mot de code le plus proche au sens de la distance de Hamming.

- il peut exister plusieurs mots équidistants. On choisit un représentant.
- procédé de correction lourd : à base de listes.

Exemple :

Le code à répétition $C_{3(1, 4)}$, distance de Hamming de $C_3 = 4$.



4.4. Codes de Hamming

Famille de codes linéaires auto-correcteurs faciles à corriger.

Principe de construction d'un code de Hamming dense $C_{(2^m-m-1,\ 2^m-1)}$:

- Chaque colonne de sa matrice de contrôle $H_{(m,\ 2^m-1)}$ est non-nulle et différente.

Propriétés :

. leur distance minimale est égale à 3 (au moins)
. correction des erreurs simples et détection des erreurs doubles, au moins.

Auto-correction :

- le syndrome du mot reçu est identique à la colonne de la matrice de contrôle correspondant au bit à corriger.
- si l'on trie les colonnes de H suivant leur poids binaire croissant et que les poids de ses colonnes couvrent l'intervalle $[1,\ 2^m-1]$ alors la valeur binaire du syndrome est égale au numéro de bit erroné.

Un code de Hamming est compact si les 2^{n-k} syndromes différents sont utilisés pour identifier les $2^{n-k}-1$ erreurs simples (+ syndrome nul !).

Exemple :

- Code de Hamming $C_{4(4,\ 7)}$ tel que sa matrice de contrôle $H_{4(3,\ 7)} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$.

Calculez la matrice génératrice G_4 puis les syndromes de $c_1 = [0\ 0\ 1\ 0\ 1\ 1]$, de $c_2 = [0\ 0\ 1\ 0\ 1\ 1\ 0]$ et de $c_3 = [0\ 0\ 0\ 1\ 1\ 1\ 1]$.

Le code est systématique, la transposée de H_4 : $H_4^T = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ donc $G_4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$

$c_1 . H_4^T = [0\ 0\ 0] \rightarrow i_1 = [0\ 0\ 1\ 0]$

$c_2 . H_4^T = [0\ 0\ 1] \rightarrow$ erreur sur le dernier bit $\rightarrow i_2 = [0\ 0\ 1\ 0]$

$c_3 . H_4^T = [1\ 0\ 0] \rightarrow$ erreur sur le $5^{ème}$ bit $\rightarrow i_3 = [0\ 0\ 0\ 1]$

4.5. Conclusion

La méthode matricielle impose de travailler sur des mots de taille fixe, ce qui est un inconvénient majeur pour les réseaux informatiques où les messages sont de taille variable.

5. Les codes polynômiaux

5.1. Présentation

Notation : tout vecteur peut être présenté sous une forme polynômiale :

- $U = <u_0,u_1,u_2,...,u_n> \Leftrightarrow U(x) = u_0+u_1.x+u_2.x^2+...+u_n.x^n$

Attention : les opérations sont binaires (construits sur le corps Z/2Z) : 1.x + 1.x = 0.x !

- Somme polynômiale :
  - somme deux à deux de chaque coefficient (somme vectorielle)
  - exemple : $<x^3+1> + <x^4+x^2+1> = <x^4+x^3+x^2>$
- Produit polynômial :
  - produit vectoriel
  - exemple : $<x^3+1> . <x+1> = <x^4+x^3+x+1>$
- Division polynômiale
  - exemple : $<x^4+x^3+x^2+1> / <x+1> = <x^3+ x+1>$
- Modulo : $<x^4+x^3+x^2+x+1> // <x+1> = 1$

Définition : Un code polynômial est un code linéaire dont chacun des mots du code est un multiple d'un polynôme générateur (noté g(x)).

$\Leftrightarrow$ les lignes de la matrice génératrice sont engendrées par le polynôme générateur.

- Le degré du polynôme est égal à la longueur du champ de contrôle d'erreur.

5.2. Exemples

Soit le polynôme générateur : g(x) = x+1

. on s'intéresse à un code $C_{5(2,3)}$ (les calculs se feront modulo x^3+1),

. 1 et x sont pris comme coefficients multiplicateurs,

. - première ligne : $(g(x).1)//(x^3+1) = x+1$

. - deuxième : $(g(x).x)//(x^3+1) = x^2+x$

. ce qui donne la matrice génératrice suivante : $G_{5(2,3)} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$

Exemples de codes polynômiaux :

- L'avis V41 du CCITT conseille l'utilisation de codes polynômiaux (de longueurs n = 260, 500, 980 ou 3860 bits) avec le polynôme générateur :
  . $g(x) = x^{16} + x^{12} + x^5 + 1$.
- Le polynôme CRC-16 est utilisé par le protocole HDLC :
  . $g(x) = x^{16} + x^{15} + x^2 + 1$.
- Le polynôme suivant est utilisé par Ethernet :
  . $g(x) = x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+1$.

5.3. Principe du codage

Le mot de code m(x) d'un code polynômial (k, n) de polynôme générateur g(x) associé au mot initial i(x) est défini par :

m(x) = i(x). x^{n-k} + r(x), où r(x) est le reste de la division de i(x).x^{n-k} par le polynôme générateur g(x) (noté : r(x) = i(x).x^{n-k}//g(x)).

Remarques :

(i) Les r = n-k bits de r(x) (de degré ≤ n-k-1) forment les bits du champ de contrôle.
(ii) Les bits de poids fort (de degré > n-k-1) forment le mot initial (→ code systématique)
(iii) L'opération de codage effectuée à l'émission est ramenée à une division polynômiale, qui peut être réalisée simplement (électroniquement).

Exemple:

Code polynômial $C^6_{(4,5)}$ de polynôme générateur $g^6(x)$= x+1.

. i(x) = x^3 + x + 1
. i(x). x = x^4 + x^2 + x
. r(x) = (x^4 + x^2 + x) // (x+1) = 1
. m(x) = x^4 + x^2 + x + 1

5.4. Principe du décodage

A la réception, chaque mot reçu m'(x) est divisé par le polynôme générateur g(x).

- Un reste non-nul indique qu'il y a eu erreur lors de la transmission.
- Syndrome de m'(x) : m'(x)//g(x) ≠ 0 ⇒ Erreur !

Attention : la réciproque est fausse ! Si le reste est nul cela ne veut pas dire qu'il n'y a pas eu d'erreurs → subsistance d'erreurs dites résiduelles.

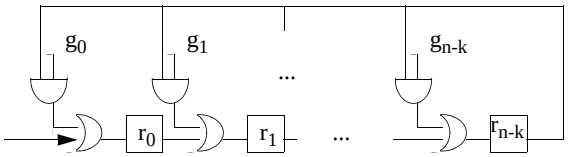
Exemple :

. m'(x) = x^4 + x^2 + x + 1
. S(m'(x)) = m'(x) // g(x) = (x^4 + x^2 + x + 1) // (x+1) = 0 → pas d'erreur détectée

5.5. Schéma électronique d'un diviseur polynômial

La multiplication est réalisée par un *ET logique*, l'addition par un *OU exclusif*, plus des registres à décalage.

Le diviseur : $g(x) = g_0+g_1.x+g_2.x^2+...+g_{n-k}.x^{n-k}$



Procédé :

(i) les registres r_i sont mis à zéros
(ii) les bits du mot à diviser sont insérés en entrée (k étapes), bits de poids fort en tête.
(iii) les registres r_i contiennent alors le reste, qu'on extrait (n-k étapes).

De nombreuses optimisations sont possibles :

- lorsque $g_i=0$ on supprime simplement la connexion et la porte ET !
- phase spécifique d'initialisation, etc.

6. Les codes cycliques

6.1. Définitions

Un code cyclique (k, n) est un code linéaire (k, n) tel que toute permutation circulaire d'un mot du code est encore un mot du code.

Exemple :

- Un code cyclique (1, 2) possède les mots de code suivants : {01, 10} ou {00, 11}, mais pas {01, 11}.
- Un code cyclique (1, 3) possède les mots de code suivants : {000, 111}.

Un code cyclique (k, n) est un code polynômial dont le polynôme générateur divise $x^n + 1$.

- Les dispositifs de codage et de décodage sont identiques à ceux des codes polynômiaux.

Les codes cycliques (k, n) dont le polynôme générateur est primitif et tel que $n=2^{n-k} + 1$, sont des codes de Hamming.

- On appelle période du polynôme H(x) le plus petit entier u tel que H(x) divise x^u+1.
- Un polynôme est irréductible s'il ne possède aucun diviseur de degré supérieur à zéro.
- Si la période d'un polynôme irréductible est égale à n-k alors le polynôme est dit primitif.

6.2. Capacité de détection

- Capacité de détection des erreurs :

Un code cyclique (k, n) dont le polynôme générateur au moins 2 coefficients non-nuls (donc il ne divise pas x^i, i<n) permet de détecter toutes les erreurs simples.

- idée de preuve : $(C(x) + x^i) \,//\, g(x) \neq 0$

Si le polynôme générateur d'un code cyclique (k, n) a un facteur irréductible de trois termes (il ne divise ni x^i ni $1 + x^{j-i}$, i<j<n), le code permet de détecter les erreurs doubles.

. par exemple : $g(x) = x^3 + x + 1$

Pour qu'un code polynômial détecte toutes les erreurs d'ordre impair, il suffit que son polynôme générateur ait (x+1) comme facteur.

. Par exemple : le code de polynôme générateur (x+1), qui est équivalent à la parité.

- Capacité de détection des paquets d'erreurs :

Un code cyclique (k, n) permet de détecter toutes les erreurs d'ordre $l \leq n-k$ (c'est-à-dire inférieur au degré du polynôme générateur).

Et la probabilité de ne pas détecter les erreurs d'ordre l>n-k est très faible et égale à : $2^{-(n-k)}$

6.3. Codes correcteurs performants

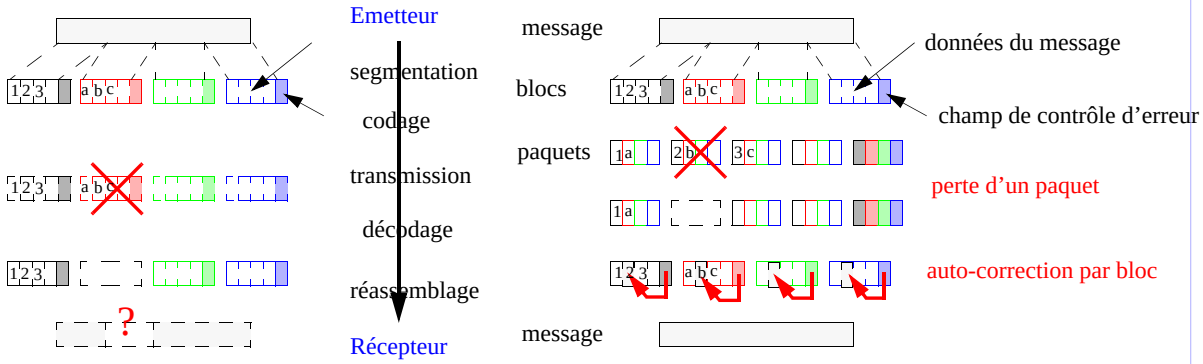
Les codes BCH (Bose-Chaudhuri-Hocquenghem) sont ceux qui ont la plus grande capacité de correction d'erreurs indépendantes pour une redondance et une longueur données. Leur rendement n'est pas très élevé. Ce sont une extension des codes cycliques, ils sont non pas construit sur un alphabet binaire mais un alphabet composé d'un grand ensemble de symboles.

Les codes RS (Reed-Solomon) sont des codes correcteurs très puissants. Ils peuvent être présentés comme des codes BCH dans lequel chaque bit des mots du code est remplacé par un entier défini modulo 2^v (avec $n=2^v-1$). La distance d'un code RS(m, n) est égale à n-m+1.

7. Autres techniques de correction d'erreurs

7.1. Entrelacement

On répartie les bits consécutifs des blocs formant le message à transmettre dans des paquets différents. Ainsi, un groupe d'erreurs (affectant un paquet) affecte uniquement quelques bits dans plusieurs blocs. La technique de protection contre les erreurs appliquée aux blocs est à même de corriger indépendamment ces quelques bits erronés dans chacun de ces blocs.

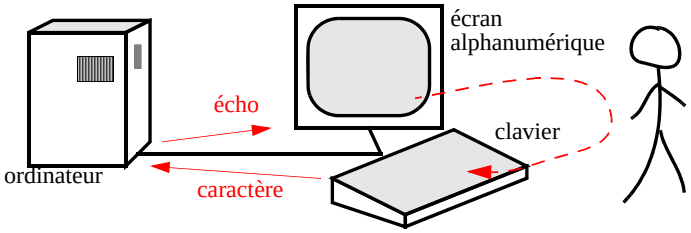


8. Techniques de contrôle d'erreur par retransmission

8.1. Techniques de détection des erreurs

L'écho permet de s'assurer que l'ordinateur a reçu correctement les informations émises.

Exemple : technique utilisée sur les liaisons (asynchrones) entre les ordinateurs et les périphériques par caractères !



Nécessite un opérateur humain : très mauvais rendement et délai important.

8.2. Technique de masquage des pertes par répétition

Les informations à transmettre sont répétés plusieurs fois. Au moins un des exemplaires (redondants) sera correctement reçu. La répétition peut être périodique ou le délai entre deux répétitions peut s'accroître exponentiellement (x 2). Le nombre de répétitions est bornée. Par ex. RPC ou RIP au-dessus d'UDP. Le rendement n'est pas bon.

8.3. Technique de correction par retransmission

8.3.1 Principe de fonctionnement

- L'émetteur envoie les paquets de données un à un. Les paquets sont formés des données et muni d'un champ de contrôle d'erreur. Il conserve une copie des données envoyées.
- Le récepteur détecte les erreurs grâce au champ de contrôle d'erreur (code polynômial).
- Le récepteur informe l'émetteur de la bonne (resp. mauvaise) réception en lui retournant un paquet spécifique :
  - . acquittement positif (resp. négatif) souvent appelé ACK (resp. NACK).
- Dans le cas d'un acquittement négatif, l'émetteur doit réémettre le paquet erroné, sinon il peut émettre le prochain paquet.

→ Protocole "Send and wait"

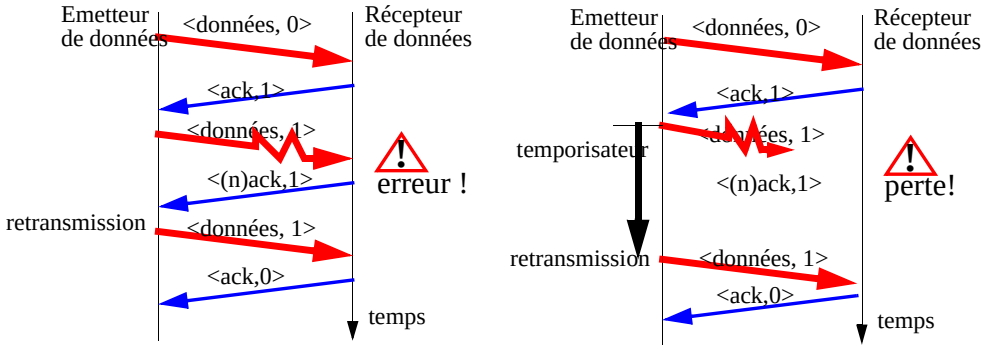
Un temporisateur bornant la durée d'attente des acquittements est nécessaire pour assurer la correction du mécanisme lors des pertes de paquets de données ou des acquittements. Le nombre de retransmission d'un même paquet est généralement borné.

L'identification des paquets (de données et d'acquittement) est nécessaire pour assurer la correction du mécanisme si l'on suppose que les paquets peuvent être retardés : au moins numérotation modulo 2.

→ Protocole "du bit alterné"

8.3.2 Exemple de fonctionnement

Fonctionne à l'alternat : Emetteur → Récepteur



Communication bidirectionnelle : x 2 !

Son rendement très faible :

- lorsque le temps de transmission T_t est faible vis-à-vis du temps de propagation T_p.
  - . car la liaison est inutilisée lorsque l'émetteur attend l'acquittement.
  - . $R = T_t / (2T_p + T_t)$

→ Optimisation : mécanisme de la fenêtre coulissante ("sliding window") !

9. Conclusion

Mécanismes de base de protection contre les erreurs :

- détection
- autocorrection
- retransmission

→ Amélioration de la fiabilité de la communication, mais ... (bien qu'avec très une faible probabilité) :

- certaines configurations d'erreur ne sont pas détectées,
- certaines configurations d'erreur ne peuvent pas être corrigées,
- certaines configurations d'erreur sont mal corrigées !

Ces mécanismes sont présents au sein de nombreuses couches (protocoles), parmi lesquelles on peut citer :

- Liaison de données (ex. Ethernet, Token Ring, FDDI, HDLC, etc)
- Réseau (ex. IP, etc)
- Transport (ex. TP4, TCP, UDP, etc)