TRANSMISSION DE L'INFORMATION

SUPPORT DE COURS
E3i, 2001-2002
Université de Tours

Michel Crucianu

Ecole d'Ingénieurs en Informatique pour l'Industrie (E3I)
64, avenue Jean Portalis
37200 TOURS

Table des matières

1. Transmission de l'information : problèmes à résoudre

La fiabilité des transmissions : protection contre les erreurs (détection/correction, fiabilité des supports).
L'utilisation efficace du réseau : débit de transmission, compression.
La confidentialité des transmissions : cryptage.
Les contraintes temporelles : débits, délais, synchronisation.
L'interopérabilité : faire coopérer des équipements/protocoles hétérogènes.

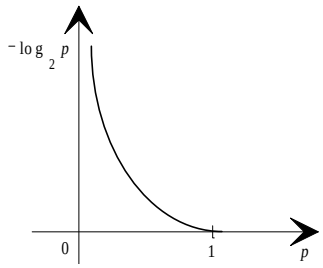
2. Eléments de théorie de l'information

2.1. Aspects qualitatifs et quantitatifs de l'information

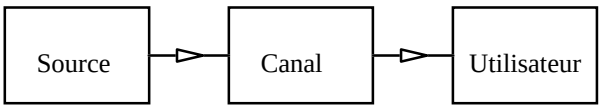
La quantité d'information qu'amène la connaissance d'un événement (l'indétermination que lève un message) est d'autant plus grande que l'événement en question est peu probable. Quand l'événement est certain, la quantité d'information qu'amène le message est bien entendu nulle. Exprimée en **bits**, la quantité d'information **au sens de Shannon** est :

$$Q = -\log_2 p (p \le 1),$$

p étant la probabilité de réalisation de l'événement.

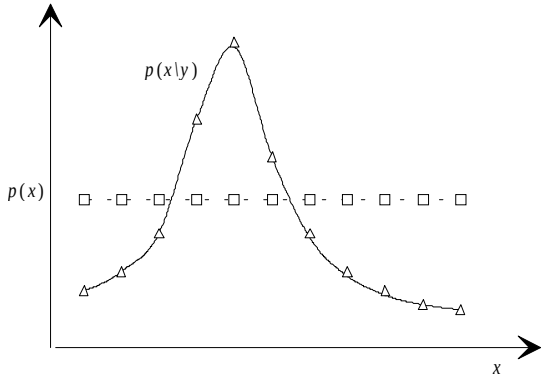


Un système de communication peut être représenté de la façon suivante :



Le **canal** (ou **voie**) transporte l'information entre une **source** (ou **émetteur**) et un **utilisateur** (ou **récepteur**). La source sélectionne et transmet des séquences x_i de **messages** (par exemple, des lettres) d'un alphabet X donné, à M éléments ; pour le récepteur, cette sélection s'effectue au hasard, en respectant toutefois une certaine distribution de probabilité (si le message n'était pas aléatoire du point de vue du récepteur, la communication serait inutile car le récepteur pourrait prédire le message à transmettre). Le récepteur peut recevoir un alphabet Y formé de N messages possibles y_i. Le canal est perturbé par un bruit qui affecte l'information transmise sur le canal. Pour le récepteur, l'action du bruit est aléatoire, sinon le bruit pourrait être entièrement prédit et donc éliminé. Le codeur et le décodeur permettent d'adapter la source et le récepteur aux caractéristiques du canal.

Au départ, l'utilisateur connaît les probabilités $p(x_k)$ pour les messages que la source peut émettre, appelées probabilités *a priori*, et les probabilités conditionnelles $p(x_k|y_i)$. Une fois reçu le message y_i, l'utilisateur a appris quelque chose de plus sur la source ; il peut utiliser $p(x_k|y_i)$ au lieu de $p(x_k)$. La figure suivante montre un exemple d'évolution $p(x_k) \rightarrow p(x_k|y_i)$ grâce à la réception de y_i :



Nous définissons l'**information mutuelle** entre les messages x_k et y_i comme

$$I(x_k;y_i)=\log_2\frac{p(x_k|y_i)}{p(x_k)}=[-\log_2\ p(x_k)]-[-\log_2\ p(x_k|y_i)],\ \begin{cases}>0 \text{ si } p(x_k)<p(x_k|y_i)\\ <0 \text{ si } p(x_k)>p(x_k|y_i)\end{cases}.$$

L'information mutuelle est une mesure du **transfert** d'information. En effet, avant la réception de y_i l'utilisateur connaît seulement la probabilité a priori $p(x_k)$, alors qu'après la réception de y_i ($p(y_i)=1$) l'utilisateur connaît $p(x_k,y_i)=p(y_i)\cdot p(x_k|y_i)=p(x_k|y_i)$ (probabilité *a posteriori*). On peut montrer facilement que l'information mutuelle est symétrique, $I(x_k;y_i)=I(y_i;x_k)$. On observe que $I(x_k;y_i)>0$ pour $p(x_k|y_i)>p(x_k)$ et $I(x_k;y_i)<0$ pour $p(x_k|y_i)<p(x_k)$. Cas particuliers pour $p(x_k)$ donnée :

1° Recevoir y_i informe avec certitude l'utilisateur que le message x_k a été émis

$p(x_k|y_i)=1\Rightarrow I(x_k;y_i)=-\log_2\ p(x_k)$, l'information mutuelle est maximale.

2° y_i est statistiquement indépendant des x_k

$p(x_k|y_i)\equiv p(x_k)\Rightarrow I(x_k;y_i)=0$, l'arrivée de y_i ne nous apprend rien sur x_k.

2.2. Self-information, entropie de la source

Supposons que le fait de recevoir y_i informe avec certitude l'utilisateur que le message x_k a été émis, $p(x_k|y_i)=1$, alors $I(x_k;y_i)=I(x_k)=\log_2\frac{1}{p(x_k)}=-\log_2\ p(x_k)$, que nous appelons self-information fournie par x_k. La self-information représente donc la quantité d'information contenue dans x_k. Nous pouvons vérifier que $I(x_k;y_i)\le I(x_k)$.

La valeur moyenne de la self-information contenue dans la source d'alphabet X sera alors

$$H(X)=-\sum_{k=1}^{M}p(x_k)\cdot\log_2\ p(x_k)=\sum_{k=1}^{M}p(x_k)\cdot I(x_k),$$

(par convention, si $p(x)=0$ alors $p(x)\cdot\log_2\ p(x)=0$)

appelée **entropie de la source**. On peut montrer que si la source peut émettre M messages, alors $H(X)\le\log_2 M$, l'égalité étant obtenue pour $p(x_k)=1/M$, $\forall k$. Le maximum de $H(X)$ est donc obtenu lorsque tous les messages que la source peut émettre sont équiprobables.

Valeurs moyennes de l'information mutuelle :

1° Moyenne sur X :

$$I(X;y_i)=\sum_{k=1}^{M}p(x_k|y_i)\cdot I(x_k;y_i).$$

On peut montrer que $I(X;y_i)\ge 0$ (même si, pour certains *k*, $I(x_k;y_i)<0$), l'égalité se produisant lorsque y_i est statistiquement indépendant des x_k ($p(x_k|y_i)\equiv p(x_k)$, l'arrivée de y_i ne nous apprend rien sur les x_k). On peut définir aussi l'entropie conditionnelle de la source :

$$H(X|y_i)=-\sum_{k=1}^{M}p(x_k|y_i)\cdot\log_2\ p(x_k|y_i)$$

et on peut montrer que $p(y_i)\cdot H(X|y_i)\le H(X)$, donc une fois y_i reçu ($p(y_i)=1$), l'entropie de la source vue par l'utilisateur ne peut que diminuer.

2° Moyenne sur X et Y :

$$I(X;Y)=\sum_{i=1}^{N}\sum_{k=1}^{M} p(x_k,y_i)\cdot I(x_k;y_i),$$

$$H(X|Y)=-\sum_{i=1}^{N}\sum_{k=1}^{M} p(x_k,y_i)\cdot \log_2\ p(x_k|y_i).$$

On peut montrer que $I(X;Y)\geq 0$, $H(X|Y)\leq H(X)$, $I(X;Y)=H(X)-H(X|Y)$ et donc $0\leq I(X;Y)<H(X)$. $H(X)$ est la quantité moyenne d'information contenue dans la source, $I(X;Y)$ est la quantité moyenne d'information transmise à travers le canal et $H(X|Y)$ (appelée **équivocation**) est la quantité moyenne d'information encore contenue dans la source après réception du message.

Cas extrêmes :

1° La transmission est (presque) sans erreur ($\forall i,\exists k,\ p(x_k|y_i)\cong 1$) : $I(X;Y)\cong H(X)$.

2° Les messages reçus sont indépendants des messages envoyés ($\forall k,\forall i,\ p(x_k|y_i)\equiv p(x_k)$) : $I(X;Y)=0$.

Propriétés utiles de l'entropie :

1° L'entropie dépend du niveau d'analyse du système : si un des messages est décomposé en deux messages (analyse plus fine) l'entropie augmente :

$$H'(X)=-p(x_{j1})\cdot \log_2\ p(x_{j1})-p(x_{j2})\cdot \log_2\ p(x_{j2})-\sum_{k\neq j} p(x_k)\cdot \log_2\ p(x_k),\ \text{avec}$$

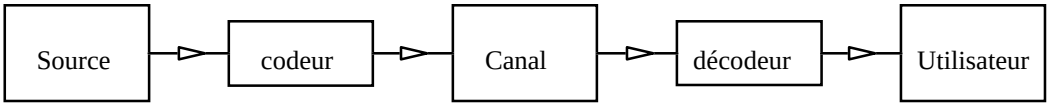
$$p(x_j)=p(x_{j1})+p(x_{j2}),\ \text{et alors}$$

$$H'(X)-H(X)=-p(x_{j1})\cdot \log_2\ p(x_{j1})-p(x_{j2})\cdot \log_2\ p(x_{j2})+p(x_j)\cdot \log_2\ p(x_j)=$$
$$=p(x_{j1})\cdot \log_2\frac{p(x_j)}{p(x_{j1})}+p(x_{j2})\cdot \log_2\frac{p(x_j)}{p(x_{j2})}>0.$$

2° L'entropie de la réunion de deux sources est supérieure à la somme pondérée des entropies des sources.

2.3. Canal de transmission idéalisé

Un canal est **discret** si les deux alphabets X et Y sont discrets — ce sera le cas dans ce qui suit. Nous avons considéré jusqu'ici que le canal pouvait transmettre tous les M symboles de l'alphabet X de la source, ce qui en général n'est pas le cas. Supposons que l'alphabet du canal, noté Z, est composé de D symboles, avec $D < M$. Il faut alors intercaler un codeur entre la source et le canal, codeur qui fera correspondre de façon biunivoque à chaque message x_k émis par la source une séquence z_k de longueur n_k de symboles appartenant à Z :



Nous avons ainsi adapté la source au canal. Soit $\overline{n}$ la longueur moyenne des séquences :

$$\overline{n}=\sum_{k=1}^{M} n_k\cdot p(x_k).$$

Dans le cas d'un canal idéal, nous désirons trouver un codeur qui assure une valeur minimale pour $\overline{n}$. L'entropie de la source est $H(X)$, qui sera aussi l'entropie du codeur **par messages** ; l'entropie du codeur **par symboles** sera alors $\frac{H(X)}{\overline{n}}$. L'entropie maximale du codeur est $\log_2 D$, donc l'efficacité du codeur est :

$$E = \frac{\frac{H(X)}{\overline{n}}}{\log_2 D}$$; $E < 1$ dans tous les cas utiles (voir plus loin), donc $\overline{n} \geq \frac{H(X)}{\log_2 D}$.

Le plus simple est d'effectuer le codage par des blocs de longueur fixe $\overline{n}$. Il faut que le nombre de codes distincts soit suffisant, c'est à dire $D^{\overline{n}} \geq M$, d'où $\overline{n} \geq \frac{\log_2 M}{\log_2 D}$. Dans le cas particulier où tous les messages sont équiprobables, $H(X) = \log_2 M$, ce qui nous mène à la même inégalité $\overline{n} \geq \frac{H(X)}{\log_2 D}$. Le signe égal est obtenu quand $M = D^{\overline{n}}$, et dans ce cas l'efficacité est de 1.

Le codage doit donc compte de la distribution de probabilité des messages : plus un message est probable, plus le code associé doit être court. Avec les codes de longueur variable se pose le problème de déterminer le début d'un mot de code. Les codes permettant une démarcation sans ambiguïté s'appellent **codes séparables**. Il a été montré que pour un code séparable optimal les inégalités suivantes sont vérifiées (attention, $\overline{n}$ n'est pas un entier) :

$$\frac{H(X)}{\log_2 D} \leq \overline{n} < \frac{H(X)}{\log_2 D} + 1.$$

La borne inférieure est valable pour tout code séparable (si le code n'est pas séparable, $\overline{n}$ peut être inférieur à la limite indiquée, et donc l'efficacité supérieure à 1 !), la borne supérieure est valable uniquement pour des codes judicieusement choisis (nous verrons un exemple dans le chapitre "Compression des informations").

Définissons une **extension d'ordre** n de la source initiale. Supposons que nous avons à transmettre n messages de X ; si au lieu de transmettre les messages au fur et à mesure de leur arrivée nous les emmagasinons dans un codeur, nous aurons alors à transmettre M^n messages u possibles (nous désignerons par $U = X^n$ cet ensemble), chaque message étant constitué d'une séquence de n symboles appartenant à X. Le résultat est une extension d'ordre n de la source. Le récepteur reçoit des messages v correspondants (nous noterons par $V = Y^n$ cet ensemble).

Dans l'hypothèse où les messages x émis par la source sont statistiquement indépendants, avec une distribution de probabilité $p(x)$ stationnaire (qui ne change pas dans le temps), la borne inférieure indiquée peut être atteinte **en limite**. Shannon a montré, en effet, en partant des inégalités précédentes et en utilisant des extensions d'ordre n de la source, qu'on peut obtenir (pour des codes bien choisis) :

$$\lim_{n \to \infty} \overline{n} = \frac{H(X)}{\log_2 D},$$

$\overline{n}$ étant la longueur moyenne de la séquence par message de la source. Ce résultat est appelé le **premier théorème fondamental de Shannon** et indique qu'il est effectivement possible de trouver un code séparable d'efficacité 1 (c'est un résultat **en limite**, qui suppose que le nombre de messages groupés avant codage tend vers l'infini).

2.4. Canal de transmission réel

Contrairement au canal idéalisé, un canal réel est affecté par le bruit. Le canal est caractérisé par une **matrice de transition** P (appelée aussi **matrice stochastique**) :

$$P = \begin{bmatrix} p(y_1|x_1) & \cdots & p(y_N|x_1) \\ \vdots & \ddots & \vdots \\ p(y_1|x_M) & \cdots & p(y_N|x_M) \end{bmatrix} \qquad \text{(rappel : } p(y|x) = \frac{p(x|y)\cdot p(y)}{p(x)}\text{).}$$

Le canal est **constant** (par opposition à un canal **à mémoire**) si sa matrice de transition est constante dans le temps (ne change pas d'un message au suivant). Nous considérerons ici uniquement des canaux **constants**.

Cas extrêmes :

1° Le canal n'est pas perturbé par le bruit. Dans ce cas, $M = N$ et $P \equiv I_N$ modulo une renumérotation des y ou des x (matrice identité d'ordre N).

2° Il n'y a aucune corrélation entre les messages d'entrée et les messages de sortie, c'est à dire $p(y_i|x_k) = p(y_i)$. Dans ce cas, toutes les lignes de la matrice sont identiques.

Nous avons remarqué que pour un canal $0 \le I(X;Y) < H(X)$, les cas extrêmes étant celui d'un canal (presque) sans bruit ($I(X;Y) \cong H(X)$) et celui d'un canal qui ne transmet pas l'information ($I(X;Y) = 0$). Pour un cas intermédiaire, la matrice de transition P étant donnée, il serait utile de déterminer le maximum de $I(X;Y)$ par rapport aux distributions de probabilités possibles pour les messages de la source ($p(x)$). **Afin d'améliorer la transmission, nous essayerons de privilégier ($p(x)$ élevé) les messages de la source qui ne sont pas affectés par les perturbations spécifiques au canal (telles qu'elles sont indiquées par P), et d'utiliser le moins possible les autres messages ($p(x) \cong 0$).**

Le problème peut être formulé de façon plus générale en faisant appel aux extensions d'ordre *n* de la source (et du récepteur). Le codeur transmet les M^n messages u possibles, chaque message étant constitué d'une séquence de *n* symboles appartenant à X. Le récepteur reçoit des messages *v* correspondants. Les distributions $p(u)$, $p(v)$ et $p(v|u)$ peuvent être facilement obtenues si les distributions $p(x)$, $p(y)$ et $p(y|x)$ sont connues et les symboles successifs sont indépendants :

$$p(u) = \prod_{i=1}^{n} p(x_i), \qquad p(v|u) = \prod_{i=1}^{n} p(y_i|x_i), \qquad p(v) = \sum_{u \in U} p(u)\cdot p(v|u).$$

Nous pouvons alors définir l'information mutuelle moyenne entre $U = X^n$ et $V = Y^n$:

$$I(X^n;Y^n) = \sum_{X^n}\sum_{Y^n} p(u,v)\cdot \log_2 \frac{p(v|u)}{p(v)}.$$

$I(X^n;Y^n)$ représente la quantité d'information fournie en moyenne par la séquence reçue v sur la séquence envoyée u. Alors $\frac{I(X^n;Y^n)}{n}$ représente la quantité d'information fournie, en moyenne, par un symbole de sortie y sur un symbole d'entrée x. La **capacité d'un canal** discret et constant est :

$$C \equiv \max_{p(u),n} \frac{I(X^n;Y^n)}{n}.$$

On peut montrer que $I(X^n;Y^n) \le \sum_{i=1}^{n} I(X_i;Y_i)$, où $I(X_i;Y_i)$ est l'information mutuelle moyenne associée aux symboles occupant la *i*-ème position des séquences u et v. L'égalité a lieu lorsque les symboles de la séquence d'entrée sont statistiquement indépendants les uns des autres (cas utile), ainsi que lorsque les symboles de la séquence de sortie sont statistiquement indépendants des symboles de la séquence d'entrée (cas sans intérêt).

$I(X^n;Y^n)$ sera donc maximum lorsque les x_i successifs (et par conséquence les y_i successifs) seront **statistiquement indépendants**. Dans ce cas, maximiser $I(X^n;Y^n)$ revient à maximiser les différents $I(X_i;Y_i)$, donc à choisir la distribution optimale $p(x_i)=p(x)$, quel que soit le rang du symbole d'entrée. Alors $I(X_i;Y_i)=I(X;Y)$, et donc $\frac{I(X^n;Y^n)}{n}=I(X;Y)$ quel que soit *n*, c'est à dire $C\equiv\max_{p(x)} I(X;Y)$.

Premier cas particulier : canal (presque) non bruité ; alors $I(X;Y)\cong H(X)$, et comme $H(X)$ est maximal lorsque les messages sont équiprobables, $p(x)$ recherchée est la distribution équiprobable.

Nous appellerons un canal discret et constant **uniforme à l'entrée** lorsque les lignes de la matrice de transition sont des permutations d'un même ensemble de valeurs. Pour un tel canal, la transmission est perturbée de la même manière quel que soit le symbole injecté à l'entrée. Pour un tel canal – la matrice de transition (et donc l'équivocation) étant donnée – la capacité est maximale lorsque l'entropie du récepteur est maximale, c'est à dire lorsque la probabilité des symboles d'entrée à été choisie de telle sorte que les symboles de sortie sont équiprobables. Un canal discret et constant est **uniforme à la sortie** lorsque les colonnes de la matrice de transition sont des permutations d'un même ensemble de valeurs. Pour un tel canal, des symboles d'entrée équiprobables entraînent des symboles de sortie équiprobables.

Deuxième cas particulier : canal uniforme à l'entrée et à la sortie. Pour un tel canal la capacité est donc maximale lorsque les symboles d'entrée sont équiprobables (et donc les symboles de sortie, voir plus haut). Cette capacité maximale sera :

$$C=\log_2 N+\sum_{j=1}^{N} p_j\cdot\log_2 p_j \left(\leq \log_2 N\right),\ p_j \text{ étant les éléments d'une ligne de la matrice } P.$$

Pour une source X nous pouvons définir le **taux** (ou débit d'informations) $R_T=\frac{H(X)}{T_S}$, T_S étant l'intervalle entre deux émissions de la source. Pour un canal qui reçoit les messages de X et délivre les messages de Y, canal caractérisé par sa matrice de transition P et sa capacité C, nous pouvons définir la **capacité par unité de temps** $C_T=\frac{C}{T_C}$, T_C étant l'intervalle entre deux émissions sur le canal. Le taux et la capacité par unité de temps sont mesurés en bits/seconde. Les résultats suivants ont été démontrés :

1° La probabilité moyenne d'erreur par symbole émis par la source, p_e, satisfait l'inégalité

$$(R_T-C_T)\cdot T_S\leq H(p_e)+p_e\cdot\log_2(M-1),$$

$H(p_e)=-p_e\cdot\log_2 p_e-(1-p_e)\cdot\log_2(1-p_e)$ étant l'**entropie d'erreur**. Si $R_T>C_T$ (taux source supérieur à la capacité par unité de temps du canal) alors pour p_e il existe une borne inférieure strictement positive (**théorème réciproque du codage**). Lorsqu'on émet une séquence de symboles, la probabilité d'erreur sur la séquence tend donc vers 1 lorsque la longueur de la séquence augmente.

2° Lorsque $R_T<C_T$, **en utilisant une procédure adéquate pour le codage et le décodage**, le récepteur peut récupérer les messages émis par la source avec une probabilité d'erreur arbitrairement petite (**second théorème fondamental de Shannon**). Explication : la probabilité d'erreur **par symbole du canal** est bornée, mais un codage **redondant** permet de réduire la probabilité d'erreur par symbole de la source (au prix d'une diminution de l'efficacité du codage...). Avant ce résultat on considérait que le bruit d'un canal introduisait une borne sur la probabilité d'erreur d'une transmission, quel que soit le taux de la source. Grâce à ce résultat nous savons que la seule borne est sur le taux de la source, la qualité de la restauration du message par le récepteur pouvant être arbitrairement élevée.

3. Codage des informations

3.1. Types de codage

Code binaire = correspondance entre un ensemble d'informations élémentaires (l'**alphabet**) et un ensemble de configurations binaires (les **mots de code**, de longueur fixe pour la plupart des codes employés). Dans la mesure où les données à représenter peuvent être mises sous la forme d'un texte, les informations élémentaires sont constituées par les 10 chiffres, les 26 lettres de l'alphabet, les signes graphiques et quelques caractères de contrôle. Le CCITT (Comité Consultatif International de Télégraphie et Téléphonie, actuel UIT-T) a normalisé plusieurs codes, parmi lesquels le code CCITT n° 2 — 5 bits, dérivé du code Baudot et utilisé sur le réseau Télex — et le code CCITT n° 5 — 7 bits, connu aussi comme ASCII 7 (*American Standard Code for Information Interchange*). D'autres codes, non normalisés par le CCITT, peuvent être utilisés par différents constructeurs informatiques sur leurs équipements, notamment EBCDIC (8 bits) par IBM.

Code CCITT n° 5 :

$b_6b_5b_4 \rightarrow$ $b_3b_2b_1b_0 \downarrow$	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	à (1)	P	\ (1)	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	# (1)	3	C	S	c	s
0100	EOT	DC4	$ (1)	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	° (1)	k	é (1)
1100	FF	FS	,	<	L	ç (1)	l	ù (1)
1101	CR	GS	–	=	M	§ (1)	m	è (1)
1110	SO	RS	.	>	N	^ (1)	n	~ (1)
1111	SI	US	/	?	O	_ (1)	o	DEL

(1) Caractères à usage national, ici français.

La signification des caractères spéciaux est donnée dans le tableau suivant :

Fonctions de mise en page	
BS	*Backspace* (retour en arrière)
HT	*Horizontal Tabulation*
LF	*Line Feed* (nouvelle ligne)
VT	*Vertical Tabulation*
FF	*Form Feed* (nouvelle page)
CR	*Carriage Return* (retour chariot)
Fonctions de contrôle de la transmission	
SOH	*Start Of Heading* (début d'en-tête)
STX	*Start of Text* (début de texte)
ETX	*End of Text*
EOT	*End Of Transmission*
ENQ	*Enquiry* (demande)
ACK	*Acknowledge* (Acquittement)
DLE	*Data Link Escape* (echappement liaison de données)
NAK	*Negative Acknowledge* (acquittement négatif)
SYN	*Synchronous Idle* (caractère de synchronisation)
ETB	*End of Transmission Block* (fin de bloc)

Fonctions de commande des périphériques	
DC1	*X-on* (mise en route)
DC2	
DC3	*X-off* (arrêt)
DC4	
Fonctions de séparation dans les fichiers	
US	*Unit Separator* (séparateur de sous-articles)
RS	*Record Separator* (séparateur d'article)
GS	*Group Separator* (séparateur de groupes)
FS	*File Separator* (séparateur de fichiers)
Autres caractères	
NUL	caractère vide (sans aucun effet)
BEL	*Bell* (sonnerie)
SO	*Shift Out* (hors code)
SI	*Shift In* (retour en code)
CAN	*Cancel* (annulation)
EM	*End of Medium* (fin de support)
SUB	*Substitution*
ESC	*Escape* (echappement)
DEL	*Delete*

Deux techniques permettent de donner à un ou plusieurs caractères une autre signification dans l'application qui reçoit un message les contenant :

1° S'il s'agit d'un seul caractère, en le faisant précéder par le caractère ESC.
2° En insérant la suite de caractères entre deux caractères de contrôle SO et SI.

3.2. Détection et correction des erreurs

Quelle que soit la qualité d'une ligne de transmission, la probabilité d'apparition d'erreurs est non nulle (ne serait-ce qu'à cause du bruit thermique). Pour certains types de transmissions des erreurs groupées peuvent apparaître — par exemple à cause de parasites électromagnétiques pour les transmissions sur paires torsadées non blindées ou de conditions atmosphériques pour les transmissions par satellite. Si pour certaines données, comme par exemple le texte d'un télégramme, les erreurs ponctuelles ne sont pas très gênantes (le texte reste compréhensible), pour d'autres, comme les transactions financières, elles sont inacceptables. Les erreurs groupées sont inacceptables quel que soit le type de données. La détection et la correction des erreurs est donc indispensable pour la transmission de données informatiques.

La détection et la correction des erreurs est fondée sur l'utilisation d'une information redondante transmise avec l'information utile. L'ajout de cette information redondante est obtenu par un recodage. Selon le degré de redondance des codes employés, un degré de détection et/ou de correction peut être atteint. Pour la correction, deux possibilités se présentent : les codes sont suffisamment redondants pour corriger, ou les codes permettent uniquement la détection et les informations erronées sont retransmises.

Remarque importante : on ne peut pas être sûr qu'un message reçu est correct — quel que soit le code employé, il ne peut pas détecter toutes les erreurs possibles. Par conséquence, les codes choisis dans une situation particulière sont ceux qui détectent le mieux et éventuellement corrigent les erreurs les plus fréquentes dans la situation en question.

Types de codes :

1° Codes **en blocs**. Si les informations utiles à transmettre sont découpées en blocs et à chaque bloc on associe (c'est à dire on transmet à la place du bloc) un mot de code qui ne dépend que du bloc en question, le code est appelé code en blocs.
2° Codes **continus**. Si l'information à expédier n'est pas divisible en blocs et la redondance est introduite de façon continue dans l'information utile, le code est appelé code continu (aussi **convolutionnel** ou **récurrent**).

3.2.1. Quelques définitions générales

Pour un code détecteur d'erreurs, l'**efficacité** de détection $\boldsymbol{e}$ (à ne pas confondre avec l'efficacité définie au chapitre précédent) est le rapport moyen entre le nombre de messages erronés reconnus comme tels et le nombre total de messages erronés. La proportion $1 - \boldsymbol{e}$ de messages sont donc erronés et reconnus comme corrects. L'efficacité $\boldsymbol{e}$ peut être définie par rapport à une classe d'erreurs (par exemple erreurs isolées, erreurs groupées).

Le **taux d'erreurs brut** τ est la proportion moyenne de messages erronés reçus (détectés ou non comme tels). Le taux d'erreurs brut dépend du taux d'erreurs par bit ; si la longueur moyenne des messages est de *n* bits et les erreurs sont indépendantes, avec une probabilité p par bit, alors $\tau = 1-(1-p)^n$

Le **taux d'erreurs résiduel** q est la proportion des messages qui restent erronés (après détection et correction des erreurs, par codes ou par retransmission). Pour qu'un message erroné soit reconnu comme correct, il est nécessaire : 1° qu'il soit erroné (probabilité τ) et 2° qu'il n'ait pas été détecté comme tel par le code (probabilité 1 – e) ou qu'erroné une première fois et détecté comme tel (probabilité $\tau \cdot e$) il soit erroné une seconde fois (erreur différente) et non détecté comme tel (probabilité $\tau^2 \cdot e \cdot (1-e)$), etc. Par conséquence

$$q = \tau \cdot (1-e) + \tau^2 \cdot e \cdot (1-e) + \ldots$$

Si e est proche de 1 et τ est très réduit, nous pouvons effectuer l'approximation $q = \tau \cdot (1-e)$. Connaissant donc la qualité de la voie physique (τ), et la qualité exigée pour la transmission (q) nous pouvons déterminer l'exigence sur le code (e) en fonction de la classe des erreurs les plus fréquentes.

Le **rendement de la transmission** ρ est le rapport entre le nombre de bits d'information utile correctement reçus (sauf erreurs résiduelles) et le nombre total de bits envoyés. Considérons le cas de la détection par code et correction par retransmission, pour des messages de *n* bits contenant *m* bits d'information utile. Pour obtenir l'information utile, il faut que le message reçu soit correct (probabilité 1 – τ), ou erroné, détecté comme tel (probabilité $\tau \cdot e$), retransmis et correctement reçu, ou... En négligeant les autres cas, le nombre moyen de bits nécessaires pour transmettre correctement les *m* bits d'information utile d'un message sera

$$\overline{n} = n \cdot (1-\tau) + 2n \cdot \tau \cdot e \cdot (1-\tau) \cong n \cdot (1+\tau), \qquad \text{(hypothèses : } e \cong 1 \text{ et } \tau^2 \cong 0\text{)}$$

et donc le rendement sera $\rho \cong \dfrac{m}{n \cdot (1+\tau)}$.

3.2.2. Codes en blocs

Considérons un code en blocs noté $C(n,m)$ qui associe à des blocs de m bits (ensemble **U**) des mots de code de n bits (ensemble **B**), avec $n > m$; n est la **longueur du code** et m est la **dimension du code**. Un message (ou mot de code) sera noté X_i, une erreur (considérée comme un message) e_j et une configuration d'erreur E_{ij} (erreur e_j appliquée au message X_i).

Propriétés des configurations d'erreurs :

1° Le nombre de configurations d'erreurs **détectées** par le code $C(n,m)$ est constant et ne dépend pas de la nature du code.

Quel que soit le code, une configuration d'erreur est détectée si et seulement si le message qu'elle produit ne fait pas partie du code. Le nombre d'erreurs possibles est $N = 2^n$ *et le nombre de messages auxquels ces erreurs peuvent être appliquées est* $M = 2^m$. *Il y a alors* $M \cdot N$ *configurations d'erreurs possibles (un même message sera produit par* M *configurations d'erreur différentes – une par mot de code) ; parmi celles-ci,* $(N-M) \cdot M$ *produisent des messages qui n'appartiennent pas au code et sont donc détectés comme erronés ;* M^2 *produisent des messages qui appartiennent au code et qui ne peuvent donc pas être détectés comme erronés (pour chaque mot de code, il existe une configuration d'erreur et une seule permettant d'obtenir n'importe quel autre mot de code). Le nombre de configurations d'erreurs détectées est donc constant et dépend uniquement de* m *et* n. *La proportion de configurations d'erreurs détectées est alors* $1 - \dfrac{M}{N}$, *quel que soit le code en blocs choisi.*

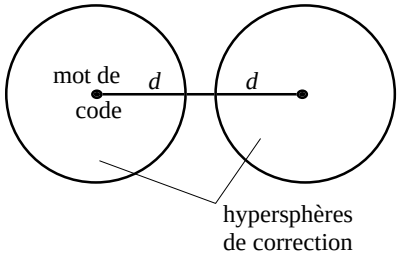
Illustration avec 00 → 0000, 01 → 0101, 10 → 1010, 11 → 1111. Note : ce code détecte les erreurs individuelles et les erreurs sur 2 bits consécutifs.

2° Le nombre de configurations d'erreur **corrigées** par le code $C(n,m)$ est constant et ne dépend pas de la nature du code.

Ces propriétés indiquent que si toutes les erreurs possibles sont équiprobables alors tous les codes en blocs se valent ; dans un cas concret, cependant, certaines erreurs sont sensiblement plus probables que d'autres et il faut

donc choisir les codes en blocs les plus adaptés. Dans ce qui suit nous étudierons des méthodes qui permettent de faire ce choix pour les classes d'erreurs les plus courantes.

Le **poids de Hamming** d'une configuration binaire est le nombre d'éléments "1" qu'elle contient. La **distance de Hamming** entre deux messages de même longueur est le poids de Hamming du message somme modulo 2 bit par bit. Cette notion de distance est très importante pour les codes en blocs : supposons que les M mots de code sont choisis de façon à ce que la distance **minimale** entre 2 mots soit $2d+1$, avec d entier ($n > m$, donc $N > M$) ; un tel code permettra de détecter toutes les erreurs d'ordre $2d$ (plus précisément, détecter comme erronés tous les messages se trouvant à une distance inférieure à $2d$ et supérieure à 0 d'un mot de code), et de corriger toutes les erreurs d'ordre inférieur ou égal à d (qui mettent le message reçu à une distance inférieure ou égale à d du mot de code émis) :



Si la distance minimale entre deux mots de code est $2d$, alors le code détecte toutes les erreurs d'ordre $2d-1$ et corrige toutes les erreurs d'ordre $d-1$.

Un code en blocs est appelé **systématique** (ou parfois **séparable**) si les m bits de chaque bloc d'information utile sont aussi les m premiers bits du mot de code (de longueur $n > m$) associé.

3.2.2.1. Codes linéaires

Comment construire des codes qui respectent une distance minimale entre les mots de code ? Comment effectuer la détection et éventuellement la correction des erreurs ?

Un code en blocs est dit **de groupe** si l'ensemble des mots de code est fermé par rapport à la somme modulo 2 bit par bit (par conséquence 0…0 est un mot de code). Nous pouvons alors constater qu'un code de groupe détecte toutes les erreurs (considérées comme des messages) n'appartenant pas au code et ne détecte aucune erreur appartenant au code.

Il est facile à montrer que l'ensemble $\mathbf{U}$ des M blocs d'information utile, de longueur m, forment un groupe avec la somme modulo 2 bit par bit (que nous noterons par "+"). Si les mots de code (ensemble $\mathbf{B}$) sont choisis de la façon suivante :

$$\text{si } \begin{cases} Y_1 \in \mathbf{U} \rightarrow X_1 \in \mathbf{B} \\ Y_2 \in \mathbf{U} \rightarrow X_2 \in \mathbf{B} \end{cases} \text{ alors } (Y_1 + Y_2) \in \mathbf{U} \rightarrow (X_1 + X_2) \in \mathbf{B},$$

le code est appelé **linéaire** (la correspondance entre les blocs d'information utile et les mots de code est une fonction linéaire).

Propriétés des codes linéaires :

1° Un code de groupe pour lequel la correspondance entre les blocs d'information utile et les mots de code (donnés) à été convenablement choisie est linéaire ; un code linéaire convenablement ordonné — c'est à dire pour lequel l'ordre des bits dans les mots de code et la correspondance entre les blocs d'information utile et les mots de code ont été convenablement choisies — est systématique (séparable).

La première affirmation est une conséquence immédiate de la définition des codes linéaires. La deuxième affirmation peut être vérifiée de la façon suivante : chaque mot de code X peut être écrit comme le produit entre la matrice T associée à la transformation linéaire qui définit le code et le bloc d'information utile Y

$$X = \begin{bmatrix} \alpha_{11} & \cdots & \alpha_{1m} \\ \vdots & \ddots & \vdots \\ \alpha_{m1} & \cdots & \alpha_{mm} \\ \vdots & \ddots & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nm} \end{bmatrix} \cdot Y .$$

Si Y ne contient qu'un seul 1, alors X est égal à une colonne de la matrice, donc toutes les colonnes de la matrice sont des mots de code. Ces colonnes sont aussi des vecteurs linéairement indépendants, sinon la transformation ne serait pas injective. Une sous-matrice carrée $m \times m$ de rang M peut alors être trouvée. Cette matrice peut être amenée à une forme diagonale (qui sera la matrice identité d'ordre m) en modifiant la transformation (donc la correspondance entre blocs d'information utile et mots de code) mais en laissant inchangé l'espace linéaire image (l'ensemble B des mots de code). Dans les mots de code ainsi obtenus, on peut alors ordonner les bits de façon que les m premiers soient ceux du bloc d'information utile et les $n-m$ suivants des bits de contrôle.

2° La distance minimum d'un code linéaire est égale au poids minimum des mots de code non nuls.

La distance entre deux mots de code est le poids de leur somme qui, pour les codes linéaires, est aussi un mot de code. La distance minimale correspond donc au mot de code non nul de poids minimal (le mot de code nul ne peut être que la somme entre un mot de code et lui-même — chaque mot de code est son propre inverse).

3° La détection des erreurs consiste en une multiplication entre le message reçu et une **matrice de vérification** (matrice qui n'est pas unique). Le résultat de la multiplication est un vecteur appelé **syndrome de l'erreur** ; un syndrome d'erreur nul signifie l'absence d'erreurs détectables par le code.

Une matrice de vérification pour un code linéaire $C(n,m)$ a $k = n - m$ lignes et n colonnes. Les lignes sont des vecteurs linéairement indépendants. Les lignes de la matrice forment une base dans le sous-espace linéaire de dimension k orthogonal au sous-espace linéaire engendré par les vecteurs de code (de cette façon, un syndrome d'erreur nul signifie l'absence d'erreurs détectables par le code). Cette condition d'orthogonalité permet d'obtenir une matrice de vérification associée à un code, comme dans l'exemple suivant :

Considérons le code linéaire $C(5,3)$ défini par la matrice

$$T^T = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}.$$

Une matrice de vérification aura 2 lignes et 5 colonnes. Considérons une ligne $\begin{bmatrix} u_1 & u_2 & u_3 & u_4 & u_5 \end{bmatrix}$ de la matrice de vérification, cette ligne doit être orthogonale à chaque ligne de T^T, ce qui donne 3 équations à 5 inconnues (la matrice de vérification n'est pas unique). Nous pouvons obtenir deux lignes linéairement indépendantes, par exemple :

$$H = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}.$$

Les composantes x_i de tout vecteur de code doivent donc satisfaire les relations :

$$\begin{cases} x_1 \oplus x_2 \oplus x_3 = 0 \\ x_2 \oplus x_4 \oplus x_5 = 0 \end{cases}$$

On peut constater qu'une erreur sur x_2 fait échouer les deux contrôles et, dans l'hypothèse que seules les erreurs qui portent sur 1 bit sont possibles, l'erreur sur x_2 est (la seule) corrigible.

Considérons un code linéaire $C(n,m)$, l'ensemble $C(Z)=\{Z+X, \forall X \in C(n,m)\}$ est appelé **classe** du mot Z. Chaque classe contient 2^m vecteurs (vecteur = configuration binaire). Pour un code linéaire, deux classes quelconques sont soit disjointes soit identiques ; tous les mots de code appartiennent à une même classe, $C(n,m)$. Le vecteur de poids minimum d'une classe est appelé **représentant** de la classe. Si le codeur de la source émet X et le décodeur du récepteur reçoit Z, l'erreur est $e = Z - X = Z + X$ et appartient donc à la même classe que Z. Le décodeur choisira alors le vecteur e de poids minimum dans la classe de Z, c'est à dire le représentant de la classe. Afin de simplifier le décodage, on peut construire un **tableau de codage complet** (appelé aussi **tableau standard**) pour le code : la première ligne contient tous les mots de code en commençant par le mot nul ; chaque ligne qui suit contient une autre classe et commence par le représentant de la classe.
Exemple :

Considérons le code linéaire $C(4,2)$ défini par la matrice

$$T^T = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}.$$

Le tableau standard sera (les blocs de données utiles ont été ajoutés sur la première ligne)

Bloc données utiles	00	10	01	11
Mots de code	0000	1011	0101	1110
Classe 1	1000	0011	1101	0110
Classe 2	**0100**	1111	**0001**	1010
Classe 3	0010	1001	0111	1100

Quand le décodeur reçoit par exemple 1111 *qui fait partie de la classe n°2, il décide que l'erreur correspond au représentant de la classe,* 0100*, et donc que le mot de code émis est* 1011*. On peut remarquer que pour la deuxième classe deux mots de poids minimal existent, le choix du premier comme représentant est arbitraire !*

La probabilité d'erreur au décodage est la probabilité pour que le vecteur d'erreur ne soit pas un représentant de classe. Si μ_i est la proportion de représentants de classe parmi les vecteurs de poids i et p est la probabilité d'erreur sur 1 bit, alors la probabilité pour que l'erreur soit un représentant de classe de poids i est $P_i = \mu_i \cdot p^i \cdot (1-p)^{n-i}$ (ici, $p^i \cdot (1-p)^{n-i}$ est la probabilité pour que i bits soient erronés et les autres $n-i$ corrects), n étant le nombre de bits d'un mot de code. La probabilité d'erreur au décodage sera alors :

$$P_{ed} = 1 - \sum_{i=1}^{n} \mu_i \cdot p^i \cdot (1-p)^{n-i} .$$

Remarque : si la distance d'un code linéaire est $2d+1$ ou $2d+2$, alors le code peut corriger d erreurs et donc tout vecteur de poids inférieur ou égal à d est le représentant (unique) d'une classe distincte. Un code est **parfait** si $\mu_i = 0, \forall i > d$, et **presque parfait** si $\mu_i = 0, \forall i > d+1$.

Quelques exemples de codes linéaires :

1° Le contrôle de parité simple : dans ce cas $n = m+1$, on ajoute un seul bit au bloc d'information utile afin d'obtenir le mot de code, et ce bit doit satisfaire l'équation $\overset{n}{\underset{j=1}{\oplus}} b_j = 0$ (parité paire) ou $\overset{n}{\underset{j=1}{\oplus}} b_j = 1$ (parité impaire). On peut facilement montrer qu'un tel code est linéaire. Ce code (parité paire ou impaire) **détecte toutes (et uniquement) les erreurs qui portent sur un nombre impair de bits** et ne peut en corriger aucune.

2° Les codes de Hamming : la matrice de vérification d'un code de Hamming est composée de k lignes et de $2^k - 1$ colonnes, les colonnes étant tous les vecteurs possibles à k éléments sauf le vecteur nul. Pour le code correspondant nous avons donc $n = 2^k - 1$ et $m = 2^k - 1 - k$. Ce code est de distance minimale 3, capable donc de détecter toutes les erreurs de poids 1 et 2 et de corriger les erreurs isolées (poids 1). En effet, soit X un mot de code, nous savons que $H \cdot X = 0$; le poids de X ne peut pas être 1 car cela

signifierait qu'une colonne de H serait 0 ; le poids de X ne peut pas être 2 car cela signifierait que deux colonnes de H sont identiques. Le poids minimal d'un mot de code non nul est donc supérieur ou égal à 3, et le code étant linéaire cela est aussi valable pour la distance minimale entre deux mots de code. Si une seule erreur est présente, son identité est facile à trouver : $H \cdot (X + e) = H \cdot e = c_j$, c_j étant une colonne de H, alors l'erreur s'est produite pour le bit *j*.

Un code linéaire peut être regardé comme un contrôle de parité généralisé. En effet, si on considère que le code est systématique, le bit de contrôle d'ordre $m+j$ est $x_{m+j} = \sum_{k=1}^{m} \alpha_{m+j,k} \cdot y_j$ et contrôle donc, dans le bloc d'information utile Y, la parité des cases pour lesquelles le coefficient α n'est pas nul.

3.2.2.2. Codes cycliques

Un code cyclique est un code linéaire fermé par rapport aux permutations circulaires (toute permutation circulaire de tout mot de code est un mot de code). Ils sont appelés aussi CRC (*Cyclic Redundancy Check*). Les codes cycliques sont bien adaptés à la détection des erreurs groupées (auxquelles nous ne nous étions pas encore intéressés) et sont faciles à implémenter, ce qui explique leur large utilisation. Propriétés des codes cycliques :

1° Les codes cycliques sont des codes linéaires et héritent donc toutes leurs propriétés.

2° Propriété fondamentale : les polynômes associés aux mots de code sont tous des multiples d'un **polynôme générateur** noté $g(x)$; réciproquement, tout polynôme (de degré inférieur à n) multiple de $g(x)$ correspond à un mot de code.

A un mot binaire quelconque nous associons un polynôme à coefficients dans $\{0,1\}$. Par exemple à $a_{m-1} \ldots a_1 a_0$ (m bits) nous associons $q(x) = a_{m-1}x^{m-1} + \ldots + a_2 x^2 + a_1 x + a_0$ (de degré maximum $(m-1)$). L'opération employée pour les coefficients est le OU-exclusif. Soit le code cyclique $C(n,m)$, alors son polynôme générateur est de degré $k = n - m$.

C (comme ensemble de polynômes associés aux mots de code) comporte un élément de degré minimal k non nul, que nous noterons $g(x)$; cet élément est unique (sinon on pourrait trouver un élément de degré inférieur). On peut montrer que les $m = n - k$ éléments $g(x), xg(x), x^2 g(x), \ldots, x^{m-1} g(x)$ sont linéairement indépendants et forment une base du code (linéaire) C. Tout mot de code est une combinaison linéaire des vecteurs de la base et donc divisible par $g(x)$. Réciproquement, si $f(x)$ est un multiple de $g(x)$ alors $f(x) = q(x) \cdot g(x)$, ce qui signifie que $f(x)$ est une combinaison linéaire des vecteurs de la base et donc fait partie du code.

3° Le polynôme générateur divise $(x^n + 1)$; ceci implique que le terme de degré 0 a le coefficient 1.

Pour le polynôme associé, l'opération de permutation circulaire d'ordre p d'un mot de code correspond à une multiplication par x^p modulo $(x^n + 1)$. Ainsi, la permutation circulaire de $f_1(x) \in C$ donne $f_2(x) \in C$, avec $x^p \cdot f_1(x) = (x^n + 1) \cdot q(x) + f_2(x)$. Si $f_1(x)$ est de degré $n-1$ et $p = 1$, alors $x \cdot f_1(x) = x^n + 1 + f_2(x)$. Comme $g(x)$ divise $f_1(x)$ et $f_2(x)$, $g(x)$ sera obligatoirement un diviseur de $(x^n + 1)$. Comme $(x^n + 1)$ n'est pas divisible par $x^\alpha, \alpha > 0$, le terme de degré 0 du polynôme générateur ne peut pas être 0.

4° Propriété importante : tout code cyclique généré à partir d'un polynôme $g(x)$ de degré k détecte tout paquet comprenant jusqu'à k erreurs.

En effet, le polynôme associé à un paquet de k erreurs contient au maximum k termes et peut toujours être mis sous la forme $x^\alpha \cdot e(x), \alpha \geq 0$ où $e(x)$ est de degré $(k-1)$, donc indivisible par $g(x)$. D'autre part, $x^\alpha, \alpha > 0$, et $g(x)$ ne peuvent avoir aucun diviseur commun, car le terme de degré 0 de $g(x)$ a le coefficient 1. Le polynôme associé à un paquet de k erreurs ne peut donc pas être divisible par $g(x)$.

Considérons un code $C(n,m)$ généré par le polynôme $g(x)$ de degré $k=n-m$. En choisissant convenablement la correspondance entre les blocs d'information utile et les mots de code nous pouvons séparer les m bits d'information utile des k bits de redondance : soit $u(x)$ le bloc d'information utile à coder, on forme d'abord $x^k \cdot u(x)$ et ensuite on divise $x^k \cdot u(x)$ par $g(x)$, $x^k \cdot u(x) = g(x) \cdot q(x) + r(x)$. Le mot de code correspondant est $x^k \cdot u(x) + r(x)$ qui appartient bien au code puisque divisible par $g(x)$ (en effet, l'opération entre les coefficients étant le OU-exclusif, $x^k \cdot u(x) + r(x) = x^k \cdot u(x) - r(x) = g(x) \cdot q(x)$).

A la réception on divise le message reçu par $g(x)$ et on détecte des erreurs si le reste n'est pas nul.

Exemple :

Considérons le code $C(3,2)$ *généré par le polynôme* $g(x) = x+1$, *diviseur de* x^3+1. *Les blocs d'information utile peuvent être* 00, 01, 10 *et* 11. *Les mots de code correspondants seront :*

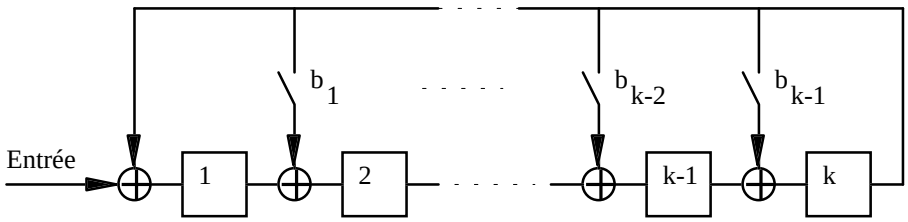
$00 \rightarrow u(x) = 0 \rightarrow x \cdot u(x) = 0 \rightarrow r(x) = 0 \rightarrow$ *mot de code* 000 ,

$01 \rightarrow u(x) = 1 \rightarrow x \cdot u(x) = x \rightarrow r(x) = 1 \rightarrow$ *mot de code* 011 ,

$10 \rightarrow u(x) = x \rightarrow x \cdot u(x) = x^2 \rightarrow r(x) = 1 \rightarrow$ *mot de code* 101 ,

$11 \rightarrow u(x) = x+1 \rightarrow x \cdot u(x) = x^2 + x \rightarrow r(x) = 0 \rightarrow$ *mot de code* 110 .

L'élément essentiel à la fois pour l'émission et la réception est donc le diviseur par $g(x)$, qui possède le schéma de principe suivant :



Registre à décalage et circuits OU-exclusif

avec $g(x) = x^k + b_{k-1}x^{k-1} + \ldots + b_1 x + b_0$ (b_0 doit être 1 pour que $g(x)$ soit diviseur de (x^n+1)). L'interrupteur i est fermé si et seulement si $b_i = 1$. Les cellules du registre sont initialisées à 0. Le dividende ($x^k \cdot u(x)$ ou le mot reçu, selon qu'il s'agit de l'émission ou de la réception) est ensuite présenté à l'entrée, poids fort en tête, et est chargé (partiellement) dans le registre à décalage pendant k pas. La division commence au pas suivant et après $m = n-k$ pas le reste de la division est contenu dans le registre.

Les codes cycliques les plus employés sont générés par les polynômes suivants :

Code à 16 bits de redondance par bloc, normalisé par le CCITT dans V41 (avec n = 260, 500, 980, 3860) :

$$g(x) = x^{16} + x^{12} + x^5 + 1.$$

Code employé par l'ARPA, 24 bits de redondance par bloc :

$$g(x) = x^{24} + x^{23} + x^{17} + x^{16} + x^{15} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^5 + x^3 + 1.$$

Les codes **polynomiaux** sont une généralisation des codes cycliques : le polynôme générateur n'est plus un diviseur de (x^n+1).

3.2.3. Codes continus

Si l'information à expédier n'est pas divisible en blocs et la redondance est introduite de façon continue dans l'information utile, le code est appelé code continu (aussi **convolutionnel** ou **récurrent**). Quand sur n bits envoyés on compte m bits d'information utile et $k = n-m$ bits de redondance, le code est dit n/m. Les bits de redondance peuvent être identifiables ou non, donc les codes sont séparables ou non. Les codes continus sont particulièrement adaptés aux erreurs en paquets séparés par des séquences correctes.

Les codes de Hagelbarger sont des codes continus séparables. Pour le plus simple, qui est un code 2/1, chaque bit d'information utile est suivi par un bit de redondance. Soit $u_1 u_2 \ldots u_k \ldots$ la suite de bits

d'information utile et $r_1 r_2 \ldots r_k \ldots$ les bits de redondance, la suite émise sera $u_1 r_1 u_2 r_2 \ldots u_k r_k \ldots$, la relation de récurrence qui produit les bits de redondance étant $r_k = u_{k-2} \oplus u_{k-4}$ (d'autres relations de récurrence peuvent aussi être employées). A la réception, les erreurs sont détectées en vérifiant la relation de récurrence. Pour ce code, un paquet de 4 erreurs peut être corrigé s'il est suivi par au minimum 13 bits corrects. De façon générale, en choisissant correctement le nombre de bits (2 seulement dans notre exemple) qui séparent les bits de redondance des bits d'information utile correspondants, un code 2/1 peut corriger tout paquet de e erreurs à condition que les paquets soient séparés par des séquences correctes de longueur minimale de $3e+1$.

Selon une technique similaire, nous pouvons construire un code continu $n/(n-1)$ avec $n > 3$ capable de corriger un paquet de e erreurs si ce paquet est suivi par $e \cdot n \cdot (n-1) - 1$ bits corrects.

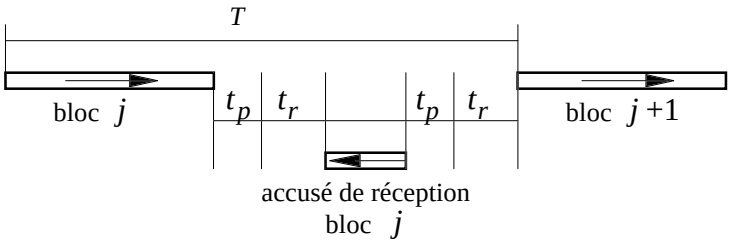
3.2.4. Correction par retransmission

Dans le cas où les erreurs peuvent être détectées mais ne peuvent pas être corrigées par la technique de codage, la retransmission devient impérative. Les types de retransmission sont :

1° Retransmission avec arrêt et attente : l'émetteur transmet le bloc j et attend un accusé de réception. Si celui-ci est négatif il émet le bloc à nouveau, sinon il continue avec le bloc $j+1$. Si aucun accusé de réception n'arrive avant expiration d'un certain délai, le bloc j est émis à nouveau.
Le temps total T nécessaire pour la transmission d'un bloc sera

$$T = \frac{ni + nc + nar}{D} + 2 \cdot (t_p + t_r),$$

où ni est le nombre de bits d'information du bloc, nc le nombre de bits de contrôle et nar le nombre de bits de l'accusé de réception, D le débit binaire de la ligne (bits/s), t_p le temps de propagation sur la ligne et t_r le temps de retournement (pour une liaison sémi-duplex).



Si p est le taux d'erreur sur un bit et si les erreurs sont indépendantes, la probabilité qu'un bloc soit transmis correctement est $P_c = (1-p)^{ni+nc}$. Si toutes les erreurs sont détectées, la durée moyenne de transmission d'un bloc, en considérant les retransmissions éventuelles, sera

$$\overline{T} = \sum_{i=1}^{\infty} i \cdot T \cdot P_c \cdot (1 - P_c)^{i-1} = \frac{T}{P_c}.$$

Le **débit efficace** sera défini comme $D_{ef} = \frac{ni}{\overline{T}}$, et donc $D_{ef} = \frac{ni}{T} \cdot (1-p)^{ni+nc}$. Ce débit passe par un maximum lorsque la longueur des blocs varie ; plus le taux d'erreurs p est réduit, plus la valeur du maximum est élevée. Par exemple, pour D = 10 Mbits/s, $ni + nc + nar \cong ni$, $t_r \cong 0$, $t_p = 100\mu s$ et $p = 10^{-5}$ on obtient le débit maximum D_{ef} = 7,54 Mbits/s pour ni = 10000 bits, donc un rendement maximum de transmission $\frac{D_{ef}}{D}$ = 75,4 %. Ce rendement descend à 30,65 % pour $p = 10^{-4}$!

2° Retransmission continue : l'émetteur transmet des blocs successifs sans attendre après chaque bloc un accusé de réception. Les accusés de réception arrivent sur une voie de retour. Quand l'émetteur reçoit un accusé de réception négatif, le bloc erroné est retransmis, **ainsi que tous les blocs qui le suivent**.

Supposons que l'émetteur peut transmettre k blocs sans avoir d'accusé de réception. La durée de transmission d'un bloc est $\frac{ni+nc}{D}$ et l'acquittement doit normalement arriver au bout de $\frac{nar}{D}+2t_p$ (nous avons supposé que la liaison était duplex et donc le temps de retournement nul). Si $\frac{nar}{D}+2t_p \le (k-1)\cdot\frac{ni+nc}{D}$ et le taux d'erreurs est nul alors il n'y a aucune attente, les blocs sont émis sans interruption. Considérons k choisi de telle sorte que l'émetteur reçoit l'accusé de réception pour le bloc $j+1$ pendant la transmission du bloc $j+k$; si l'accusé de réception est négatif, tous les k blocs (de $j+1$ à $j+k$) seront retransmis. La durée moyenne de transmission d'un bloc sera alors :

$$\overline{T}=\sum_{i=0}^{\infty}(i\cdot k+1)\cdot\frac{ni+nc}{D}\cdot P_c\cdot(1-P_c)^i=\frac{ni+nc}{D}\cdot\left(1+k\cdot\frac{1-P_c}{P_c}\right),$$

et le débit effectif $D_{ef}=\frac{D\cdot ni}{ni+nc}\cdot\left(1+k\cdot\frac{1-P_c}{P_c}\right)^{-1}$.

3° Retransmission sélective : la transmission est continue, comme dans le cas précèdent, mais uniquement les blocs erronés (pour lesquels un accusé de réception négatif a été reçu) sont retransmis.
Avec k choisi de telle sorte que l'émetteur reçoive l'accusé de réception pour le bloc $j+1$ pendant la transmission du bloc $j+k$, comme dans le cas précédent, nous obtenons :

$$\overline{T}=\sum_{i=0}^{\infty}(i+1)\cdot\frac{ni+nc}{D}\cdot P_c\cdot(1-P_c)^i=\frac{ni+nc}{D\cdot P_c}\qquad\text{et}$$

$$D_{ef}=\frac{D\cdot ni}{ni+nc}\cdot(1-p)^{ni+nc}.$$

Cette technique est la plus complexe à mettre en œuvre mais la plus performante et permet d'atteindre un rendement de 86% sur des liaisons satellite (temps de propagation de 300 ms et $p=10^{-4}$).

4. Compression des informations

4.1. Problèmes et classifications

La compression a pour but la réduction du volume des informations ; cette réduction peut être effectuée :

1° sans perte d'informations (réduction de la redondance), permettant de reconstituer de façon exacte le fichier original. Dans certains cas la redondance est très réduite ; de façon générale, l'évaluation de la redondance est très difficile et dépend du contexte. Aussi, la répartition de la redondance n'est pas uniforme dans un fichier ; la compression élimine cette redondance non uniforme et l'utilisation **ultérieure** d'un codage détecteur/correcteur d'erreurs ajoute une information redondante uniforme.

2° avec perte d'informations, utilisée principalement pour les images et les sons, permettant de reconstituer de façon approximative le fichier original. En tirant profit des imperfections des organes sensoriels humains, des taux de compression élevés peuvent être obtenus sans une détérioration sensible du résultat de la reconstitution.

L'algorithme de compression peut avoir à la base :

1° une analyse statistique générale, applicable à tout type de fichier et permettant d'évaluer théoriquement l'efficacité d'un algorithme (algorithme de Huffman, de Shannon-Fano). La complexité d'une analyse statistique d'ordre élevé limite l'efficacité de ces algorithmes.

2° des heuristiques spécifiques aux différents types de fichiers (algorithme relativement général LZW, méthode TCD pour les images, méthode PASC pour les sons). Les connaissances à priori sur le type d'informations que contient un fichier permettent l'utilisation d'heuristiques qui assurent une efficacité élevée mais pour un domaine en général restreint.

Le fichier compressé peut être :

1° décompactable par un programme externe.

2° autodécompactable (le programme de décompression fait partie du fichier). La taille du programme s'ajoute à la taille du fichier — donc perte en efficacité, mais l'algorithme de décompression peut être dédié au fichier et donc permettre un gain en efficacité.

4.2. Propriétés générales des compacteurs

Tout compacteur doit posséder un décompacteur associé. Quand la compression se fait sans perte d'informations le compacteur correspond à une fonction bijective dont l'inverse doit être calculable. Si la compression a lieu avec perte d'informations, la fonction définie par le compacteur n'est pas bijective et donc n'admet pas d'inverse ; à partir d'un fichier comprimé, le décompacteur permet de retrouver non pas le fichier original mais un fichier prototype pour un ensemble de fichiers non comprimés.

Pour tout compacteur il existe au moins un fichier non compactable. En effet, s'il existait un compacteur *C* pour lequel tout fichier était compactable, alors il suffirait d'itérer un nombre suffisant de fois l'algorithme *C* pour obtenir 1 bit, quel que soit le fichier de départ, ce qui est absurde.

Existe-t-il un compacteur C_O qui est meilleur que tout autre compacteur C sur tous les fichiers F, c'est à dire tel que $\forall F, \forall C, |C_O(F)| \leq |C(F)|$ ($|\cdot|$ symbolise la taille du fichier) ? Non, car pour tout fichier F il existe le compacteur C_F associé à la fonction de caractérisation de F et pour lequel $|C_F(F)| = 1$, compacteur défini par :

$$C_F(f) = \begin{cases} 1 & \text{si} \quad f = F \\ 0 \mid f & \text{sinon} \end{cases},$$

$\mid$ étant le symbole de concaténation. Il faut toutefois remarquer que le compacteur C_F ne présente aucun intérêt : non seulement il n'a aucun effet sur les autres fichiers, mais sa description est équivalente à celle de F. Il faut donc faire intervenir dans la notion de compactage optimal non seulement la spécification du fichier, mais aussi la spécification de l'algorithme de compression ! Plus un domaine est restreint, plus les compacteurs développés à partir de la connaissance du domaine (compacteurs heuristiques) sont efficaces dans ce domaine et inefficaces dans d'autres ; ces compacteurs incorporent une partie de la description du domaine que les fichiers comprimés n'ont donc plus à véhiculer.

4.3. Quelques techniques primitives

Ces techniques ont été les premières développées et sont encore utilisées (en conjonction avec d'autres techniques) dans des cas très particuliers.

4.3.1. Codage des répétitions

Certains fichiers (tableaux en mode texte, images) présentent des successions d'octets identiques. Nous pouvons choisir un caractère de contrôle # et coder une suite de *k* octets identiques par # octet *k*. Par exemple, la chaîne de caractères "|_19_ _ _ _ _ _ _|_ _ _ _ _" (17 octets) devient "|_19#_7|#_5" (11 octets). Ce codage n'est utile donc qu'à partir de 4 répétitions. Si on ne peut pas trouver un caractère de contrôle non employé dans le fichier on peut en choisir un qui est peu présent et chaque fois qu'on le rencontre on écrit un de plus.

4.3.2. Codage topologique

Ce codage est appliqué en général dans les fichiers où un octet *O* (ou autre groupe de bits) est sensiblement dominant. On lit le fichier par blocs de *n* octets et on utilise un octet (octet **topologique**) dans lequel les bits 1 désignent les positions de *O*. Par exemple, la chaîne "|_19_ _ _ _ _ _ _|_ _ _ _" (16 octets) devient : "(01001111)|19(11101111)|" (6 octets), les bits des octets topologiques étant écrits entre parenthèses. Ce codage amène une réduction de la taille dès que la fréquence de l'octet *O* dans le fichier est supérieure à 1/8. La taille du fichier comprimé est $T' = T \cdot (1 - f + 1/8)$, f étant la fréquence de *O*.

4.3.3. Codage relatif

Ce codage est employé quand les valeurs des codes (octets ou groupes d'octets) successifs sont comprises dans un intervalle de faible amplitude par rapport à la valeur moyenne. Par exemple, la suite d'octets "10101101 10101000 10101110 10101001" (32 bits) peut être écrite comme "5 10101 4 101 000 110 001" (28 bits), 5 (codé sur 3 bits) étant la longueur en bits du champ fixe (les 5 bits de poids fort) et 4 (codé sur un octet) le nombre de codes successifs. Une suite de résultats de mesures ou de paramètres de contrôle permet l'utilisation efficace d'un tel codage.

Une variante du codage relatif correspond au cas où les différences entre deux valeurs successives sont réduites (en valeur absolue) par rapport à ces valeurs. Il est alors utile de coder la valeur de départ et ensuite

uniquement les différences, sur un nombre de bits réduit. Par exemple, la séquence de 4 octets antérieure devient avec un tel codage "10101101 1011 0001 1100" (20 bits), les valeurs des différences étant représentées par complément à 2.

4.4. Algorithmes statistiques

L'idée de départ des algorithmes statistiques est de réduire le nombre de bits nécessaires à la représentation des symboles les plus fréquents dans un fichier (voir aussi *§ Canaux de transmission*).

4.4.1. Algorithme de Huffman

L'algorithme de Huffman (1952), sans perte d'informations, a comme point de départ une analyse statistique d'ordre 0 (calcul des fréquences des symboles) d'un fichier. Le nombre de bits utilisé pour le codage d'un symbole est d'autant plus réduit que le symbole est fréquent dans le fichier. L'algorithme :

1° Evaluer les fréquences d'occurrence des symboles du fichier.

2° Classer les symboles en ordre décroissant des fréquences d'apparition.

3° Regrouper de façon séquentielle les paires de symboles de plus faible probabilité, en reclassant symboles et groupes si nécessaire.

4° Calculer les codes avec retour en arrière en ajoutant, dans chaque point de regroupement, un 0 à une branche et un 1 à l'autre branche.

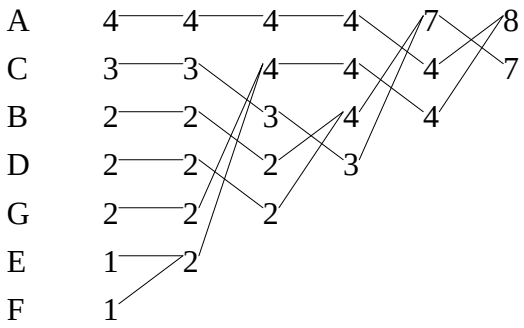
Exemple :

Chaîne à comprimer : "ABCFGABDDACEACG" (45 bits), représentée avec 3 bits/lettre.

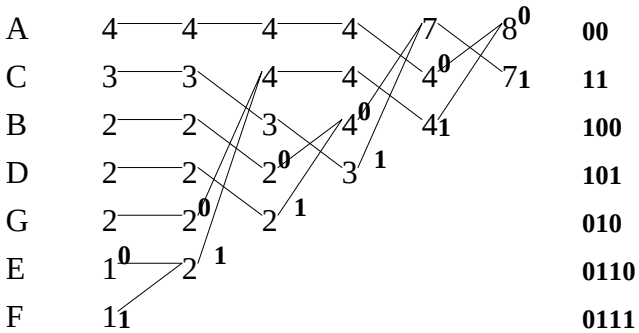
Classement des symboles :

Symbole	Fréquence
A	4
C	3
B	2
D	2
G	2
E	1
F	1

Regroupement séquentiel avec reclassement :



Codage avec retour en arrière :



Taille des données compactées : 40 bits.

Les codes binaires obtenus par l'algorithme sont **séparables** : il y a une seule façon de lire les codes dans une suite binaire. Les codes employés doivent en revanche être connus par le décompacteur, donc une table de correspondances ou une table de fréquences (permettant au décompacteur de retrouver les codes) doit être

sauvegardée dans l'en-tête du fichier comprimé. Pour qu'il occupe un minimum de place, différentes techniques peuvent être employés pour le codages de l'en-tête :

1° Une taille *T* unique — suffisante pour la fréquence la plus élevée — est employée pour coder les fréquences et un codage topologique indique quelles sont les fréquences non nulles (bit *i* à 1 signifie fréquence *i* non nulle). L'en-tête a donc la forme suivante :

256 bits topologiques $\left| T \right| f_{i_1} \left| f_{i_2} \ldots \right| f_{i_p}$, p étant le nombre de fréquences non nulles.

Un codage relatif peut éventuellement être employé ensuite pour ces fréquences non nulles.

2° Si les fréquences sont élevées, il est plus efficace de stocker dans l'en-tête directement les codes. Pour chaque code on indique d'abord sa taille, sur un nombre de bits fixé. Un codage topologique peut être employé, comme pour les fréquences, afin de n'inclure que les codes correspondant à des octets présents.

L'entropie telle que nous l'avons définie (entropie d'ordre 0) permet d'évaluer le taux de compression pour une classe donnée de fichiers avec l'algorithme de Huffman. Si les symboles sont représentés par des octets, l'entropie correspondant à un fichier est

$$E_0 = -\sum_{i=1}^{256} p_i \cdot \log_2 p_i \text{ , avec } \sum_{i=1}^{256} p_i = 1, \ p_j \text{ étant la probabilité de présence de l'octet } j.$$

L'entropie est maximale (et égale à 8) si et seulement si les octets de 0 à 255 sont tous présents et équiprobables. L'entropie est nulle si et seulement si un seul octet se répète dans tout le fichier.

A partir de l'entropie d'ordre 0 nous définissons la **redondance entropique d'ordre 0 par symbole**, qui est la différence entre la valeur maximale de l'entropie et sa valeur réelle : $R_0 = \log_2 N - E_0$, N étant le nombre maximal de symboles utilisables (en général 256, correspondant aux 8 bits du code ASCII 8). Par exemple, comme dans la plupart des textes le nombre de symboles ne dépasse pas 64, nous obtenons :

$$R_0 \geq 8 + \sum_{k=0}^{63} \frac{1}{64} \cdot \log_2 \frac{1}{64} = 8 - 6 = 2 \text{ bits de redondance par symbole.}$$

Nous avons égalité si tous les 64 symboles sont équiprobables (entropie maximale), ce qui n'est pratiquement jamais le cas ; l'entropie d'un fichier est en général inférieure, et donc la redondance supérieure. Il faut remarquer qu'une redondance **d'ordre 0** presque nulle ne signifie pas que le fichier ne peut plus être comprimé par une méthode différente de celle de Huffman (à partir d'une analyse statistique d'ordre supérieur ou d'une heuristique).

En considérant comme symboles codants des groupes de k bits, avec k > 8, l'entropie maximale augmente en général beaucoup plus vite ($\sim 2^k$) que l'entropie réelle. La redondance et donc le taux de compression possible seront alors plus élevés. L'entropie d'ordre 0 pour les symboles à k bits est :

$$E_0^k = -\sum_{i=1}^{2^k} p_i \cdot \log_2 p_i \text{ , avec } \sum_{i=1}^{2^k} p_i = 1, \ p_j \text{ étant la probabilité de présence du symbole codant } j.$$

Une recherche peut être faite par l'algorithme de compression afin de trouver une valeur optimale pour k (en général entre 4 et 16, par multiples de 4) pour un fichier donné. Malheureusement, la taille de l'en-tête augmente en général de façon exponentielle avec la taille des symboles codants considérés.

Le codage de Huffman est optimal au sens suivant : quels que soient les éléments binaires séparables eb_i associés aux probabilités p_i, l'inégalité suivante est vérifiée

$$\sum_i p_i \cdot \left| eb_i \right| \geq \sum_i p_i \cdot \left| h_i \right|,$$

h_i étant le code de Huffman associé à la probabilité p_i.

4.4.2. Algorithmes dynamiques

Un désavantage de l'algorithme de base est qu'il faut ajouter au fichier comprimé un en-tête décrivant les codes. Un autre désavantage est qu'il faut lire deux fois le fichier à comprimer : une première fois pour faire les statistiques permettant de construire les codes et une deuxième fois pour remplacer les symboles par leurs codes. Enfin, un désavantage est que le codage prend en compte les statistiques sur le fichier entier, qui peut ne pas être homogène.

Afin de tenter de résoudre les deux premiers problèmes, nous pouvons effectuer à l'avance des statistiques sur différents types de fichiers et construire des codes spécifiques pour chaque type. Chaque nouveau

fichier sera ensuite comprimé avec une seule lecture en utilisant ces codes définis au préalable, connus à la fois par le compacteur et le décompacteur — les en-têtes ne sont plus nécessaires. Bien sûr, le codage ne sera pas optimal (au sens mentionné plus haut) sur chacun des fichiers nouveaux, mais le résultat sera le plus souvent de bonne qualité.

Une autre possibilité est d'employer des algorithmes dynamiques, qui modifient le codage au cours du traitement afin de mieux l'adapter aux caractéristiques statistiques **locales** du fichier.

4.4.2.1. Algorithme sans en-tête

La table des fréquences n'est pas construite lors d'une première lecture du fichier mais est vide au départ et actualisée après la lecture de chaque symbole. Le code émis pour un même symbole dépendra donc des octets lus auparavant et ne sera pas le même au cours de la compression. Au départ, les codes proviennent d'une table de référence obtenue par des statistiques préalables sur d'autres fichiers de même type et connue à la fois par le compacteur et le décompacteur. Après la lecture d'un symbole du fichier, le code correspondant est émis et **ensuite** sa fréquence est augmentée de 1 ; le classement est alors mis à jour (le symbole est placé **immédiatement après** les autres symboles ayant la même fréquence) ; enfin, les codes sont **attribués** à nouveau (au lieu d'être **calculés** à nouveau) selon ce nouveau classement. A la décompression on procède de la même façon : la table des fréquences et la table des codes associée sont réactualisées après la lecture de chaque code (nous avons remarqué que les codes étaient séparables), permettant de récupérer correctement les symboles de départ.

4.4.2.2. Algorithme avec en-tête

Cet algorithme utilise une seule lecture du fichier, et la table des fréquences mais **aussi la table des codes** sont au départ vides. Comme pour l'algorithme précédent, la table des fréquences est mise à jour après la lecture de chaque symbole (à fréquence égale, les symboles sont classés toujours en **ordre lexicographique**) ; les codes sont en revanche **calculés** à nouveau au lieu d'être attribués à nouveau, comme auparavant. La table de codes attachée au fichier comprimé est la dernière table de codes (correspondant à la distribution des fréquences après la lecture du dernier symbole). Pour récupérer le fichier initial, il faudra donc commencer par la fin du fichier comprimé. Dans ces conditions, pour que les codes restent séparables il faudra les écrire à l'envers.

Pour les fichiers qui sont loin d'être homogènes, l'algorithme fournit de meilleurs résultats que l'algorithme de Huffman de base.

4.4.3. Algorithmes d'ordre supérieur à 0

Pour l'analyse statistique d'ordre 0, les symboles sont considérés indépendamment les uns des autres et seules les fréquences d'apparition des symboles individuels sont calculées. Il est évident toutefois que la redondance se manifeste non seulement au niveau des symboles individuels mais aussi au niveau des séquences de symboles.

Les méthodes d'ordre k tiennent compte des k antécédents d'un symbole afin de calculer un code plus efficace. Pour $k = 1$, c'est la probabilité conditionnelle d'occurrence d'un symbole en connaissant le symbole précédent qui est utilisée. L'entropie **d'ordre 1** est :

$$E_1 = -\sum_{i,j} p_{ij} \cdot \log_2 p_{i|j}$$

p_{ij} étant la probabilité d'apparition de la séquence *ji* et $p_{i|j}$ la probabilité conditionnelle.

L'algorithme de codage est similaire à l'algorithme de Huffman : après une première lecture du fichier, une table de fréquences $f_{i|j}$ est remplie et une table de codes de Huffman est créée pour chaque antécédent ; le fichier comprimé est obtenu après une deuxième lecture du fichier initial.

Par exemple, considérons la chaîne à comprimer suivante : "ACABBDDBAAA". Les fréquences conditionnelles et les codes seront alors :

Antécédent	$f_{i\|j}$	Code
sachant A	f(A) = 2	0
	f(B) = 1	10
	f(C) = 1	11
sachant B	f(A) = 1	1
	f(B) = 1	00
	f(D) = 1	01
sachant C	f(A) = 1	0
sachant D	f(B) = 1	0
	f(D) = 1	1

En codant le premier caractère du fichier, A, sur 2 bits, la taille de la chaîne compactée est de 16 bits, par rapport à 20 bits par algorithme de Huffman de base.

Malheureusement, plus l'ordre de l'algorithme augmente, plus les en-têtes sont de taille importante : $O\left(2^{(k+1)\cdot n}\right)$ (n est le nombre de bits utilisés pour les symboles dans le fichier initial et k est l'ordre de l'algorithme). Pour cette raison, ainsi que pour maintenir la rapidité des calculs dans des limites acceptables, k ne dépassera pas 2.

Enfin, des variantes dynamiques similaires existent pour les algorithmes d'ordre supérieur à 0.

4.5. Algorithmes de type dictionnaire

L'idée de départ est de construire une liste (dictionnaire) des séquences redondantes du fichier à comprimer et de remplacer chaque séquence par son adresse dans la liste.

4.5.1. **Algorithme LZW** (Lempel et Ziv 1977, Welch 1984)

Le dictionnaire est un tableau dans lequel sont rangées des séquences de symboles de taille variable. Ce tableau est construit au cours du traitement du fichier et permet de remplacer toute séquence qui s'y trouve par son adresse lors d'une occurrence ultérieure. Le dictionnaire n'est pas sauvegardé avec le fichier comprimé car il peut être **reconstruit** par le décompacteur.

L'algorithme de compression :

1° lire le premier octet ***c*** *du fichier à comprimer ;*
2° tant que non EOF
s := *octet suivant ;*
seq := c | s *;*
si ***seq*** *est dans le dictionnaire, alors faire* ***c := seq*** *;*
sinon
écrire ***adresse(c)*** *;*
ajouter ***seq*** *dans le dictionnaire ;*
c := s *;*
3° écrire ***adresse(c)***.

On considère que tout octet est identique à son adresse, donc dans le dictionnaire nous écrivons uniquement les séquences d'au minimum 2 octets.

Le nombre de bits utilisés pour coder une adresse dans la table ne sera pas constant mais augmentera au cours du traitement (avec la taille de la table). Chaque fois qu'une augmentation est nécessaire, elle est signalée au décompacteur en écrivant un indicateur (par exemple, pour passer de n à n+1 bits, le code $2^n - 1$) dans le fichier comprimé. Un problème se manifeste pour n = 8 : à la première apparition d'une adresse 255 qui ne joue par le rôle d'indicateur il faut augmenter d'abord la taille des adresses à 9 bits (écriture de 11111111) et ensuite seulement écrire l'adresse 255, sur 9 bits. De façon similaire, un certain nombre d'adresses (de préférence supérieures à 255) peuvent ainsi être réservées à la communication entre le compacteur et le décompacteur. Le décompacteur reconstruit le dictionnaire au cours du traitement, mais doit connaître d'avance les adresses réservées. L'algorithme de décompression (en faisant l'hypothèse que le fichier traité a été comprimé en utilisant l'algorithme LZW) est :

1° lire le premier code ***c*** *du fichier à décomprimer ;*
2° ***seqp := séquence(c)*** *;*
3° écrire ***seqp*** *;*
4° tant que non EOF
    s := *code suivant ;*
    seqc := séquence(s) *;*
    écrire ***seqc*** *;*
    tmp := *premier octet de* ***seqc*** *;*
    ajouter ***seqp*** **|** ***tmp*** *dans le dictionnaire ;*
    seqp := seqc *;*

Le changement de taille des adresses (codes) n'est pas explicité dans cette présentation de l'algorithme.

Exemple : Montrer le fonctionnement sur la chaîne "L'algorithme LZW et l'algorithme de ..."

4.5.2. Algorithme de Storer et Szymanski

Cet algorithme est une version de l'algorithme LZW qui utilise un tampon de taille fixe à la place du dictionnaire. Les octets du fichier à comprimer sont lus un à un dans le tampon ; tant qu'il n'y a pas de répétition de séquence d'octets, chaque octet est écrit dans le fichier comprimé avec un bit 1 comme en-tête ; dès qu'une séquence qui se répète est trouvée, la séquence maximale est recherchée et ensuite écrite dans le fichier comprimé comme un couple (position dans le tampon, longueur séquence) avec un bit 0 comme en-tête. Quand le tampon est plein, son contenu est décalé d'une position vers la gauche à chaque fois qu'un nouvel octet est introduit à l'extrémité droite ; tant que le caractère ou la séquence courante se répète, le contenu du tampon est décalé d'une position vers la gauche.

Cette version détecte plus rapidement les séquences répétitives maximales que l'algorithme LZW. Le nombre de bits utilisés pour représenter la position dans le tampon dépendra bien sûr de la taille du tampon (par exemple, 13 bits pour 8 Ko) mais on peut le faire varier selon la position en utilisant une technique simple. La longueur des séquences dépendra souvent du type de fichier et sera codée sur le nombre de bits correspondant ; sachant qu'une compression effective peut être obtenue à partir de séquences répétitives de 3 octets, le code 0 peut représenter déjà 3.

Exemple : Montrer le fonctionnement sur la chaîne "L'algorithme LZW et l'algorithme de ..."

Les algorithmes de type dictionnaire sont un moyen simple — du point de vue de la complexité des calculs — de prendre en compte des statistiques d'ordre plus élevé que l'algorithme de Huffman.

4.6. Algorithmes mixtes

Les techniques mixtes utilisent conjointement des algorithmes statistiques (en général des versions dynamiques de Huffman) et des algorithmes de type dictionnaire. Deux solutions sont les plus utilisées :

1° Appliquer LZW sur un fichier déjà comprimé par un algorithme de Huffman. En effet, l'algorithme de Huffman prend en compte les fréquences relatives occurrence des symboles individuels, alors que LZW prend en compte la répétition de séquences de symboles.
2° Appliquer l'algorithme de Huffman sur une partie des adresses obtenues après une compression par LZW (ou sur une partie des codes de position obtenus par la version de Storer et Szymanski). Les parties de poids faible des adresses peuvent être difficilement comprimées (car en général assez aléatoires). En revanche, les parties de poids fort permettent en général une compression assez importante.

4.7. Compression avec perte d'informations

Les techniques de compression présentées jusqu'ici permettent de récupérer la forme originale des fichiers traités. Si les fichiers exécutables ou les fichiers de données informatiques exigent de telles techniques, les fichiers d'images ou de sons admettent souvent une légère perte d'informations. Cette perte d'informations est le prix à payer pour obtenir des taux de compression très importants, exigés par la transmission (ex. vidéoconférence) ou

le stockage (ex. mini-disques, DCC) de quantités importantes de données. Heureusement, les systèmes sensoriels humains sont insensibles à la disparition de certaines informations dans le signal d'entrée.

4.7.1. Compression des images

Une image est représentée par une collection (matrice 2D) de pixels, chaque pixel étant décrit par un entier unique (plages de gris, ou adresse dans une table de sélection de couleurs) ou par plusieurs entiers (composantes RVB, ou luminance et chrominances).

Quelques caractéristiques des images et de la vision humaine (connaissance du domaine) : l'œil humain ne perçoit pas les 16 millions de nuances permises par le codage usuel des couleurs sur 24 bits ; plus encore, les variations fines et denses de nuance à l'intérieur d'une région de la même couleur ne sont pas perçues ; enfin, un léger lissage des contours dans l'image améliore le confort visuel. Ces constatations permettent d'éliminer une partie des informations présentes dans l'image initiale en gardant son intelligibilité et même en améliorant le confort de celui qui la regarde.

La réduction du contenu informationnel se fait par un filtrage de l'image, filtrage qui élimine du spectre les fréquences trop hautes ou à faible amplitude. Ce filtrage peut s'effectuer directement sur l'image, en utilisant la transformation de Walsh-Hadamard (lissage direct), ou sur le spectre de l'image obtenu par une transformation de Fourrier ou transformation cosinus discrète.

La transformation de Walsh-Hadamard utilise les matrices d'Hadamard engendrées par la relation de récurrence suivante :

$$H_{2n} = \frac{1}{\sqrt{2}} \cdot \begin{bmatrix} H_n & H_n \\ H_n & -H_n \end{bmatrix} \text{ avec la condition initiale } H_2 = \frac{1}{\sqrt{2}} \cdot \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

La transformation d'un vecteur V_k de taille 2^k consiste en une multiplication avec la matrice Hadamard correspondante : $Wh(V_k) = H_{2^k} \cdot V_k$. Etant donnée une image de taille $2^k \times 2^k$, la transformation est calculée d'abord sur les lignes et ensuite sur les colonnes de la nouvelle matrice. La transformation élimine les hautes fréquences de l'image (lissage direct) et donc augmente la redondance, améliorant ainsi de façon significative le taux de compression qui peut être obtenu par une technique classique.

Pour une image de taille $2^n \times 2^n$ et de pixels $f(j,k)$, la transformée de Fourrier discrète est :

$$F(u,v) = \frac{1}{2^n} \cdot \sum_{j=0}^{2^n-1} \sum_{k=0}^{2^n-1} f(j,k) \cdot e^{-\frac{2\pi}{2^n} \cdot i \cdot (u \cdot j + v \cdot k)}, \quad \text{avec } 0 \le u, v < 2^n.$$

et la transformée cosinus discrète (TCD) :

$$F(u,v) = \frac{4}{2^{n+1}-1} \cdot \sum_{j=0}^{2^n-1} \sum_{k=0}^{2^n-1} \breve{f}(j,k) \cdot \cos\left[\frac{2\pi u j}{2^{n+1}-1}\right] \cdot \cos\left[\frac{2\pi v k}{2^{n+1}-1}\right],$$

avec $$\breve{f}(j,k) = \begin{cases} f(j,k) \text{ si } j \cdot k \neq 0 \\ 0{,}5 \cdot f(j,k) \text{ si } j = 0 \text{ ou } k = 0, j \neq k \\ 0{,}25 \cdot f(j,k) \text{ si } j = k = 0 \end{cases}.$$

Seuls sont retenus les coefficients dont l'amplitude dépasse un certain seuil.

4.7.1.1. Méthode JPEG

La norme JPEG (*Joint Photographic Expert Group*) a été adoptée en 1992 et utilise trois étapes dans la compression :

1° Une transformation cosinus discrète appliquée sur des blocs de 8×8 pixels sur chaque composante de l'image (luminance Y, chrominance U, chrominance V) et donne 64 coefficients de fréquence par composante.

2° Les coefficients TCD sont divisés par des valeurs présentes dans une table de quantification et les valeurs sont arrondies à des entiers. C'est dans cette étape que certaines informations sont éliminées.

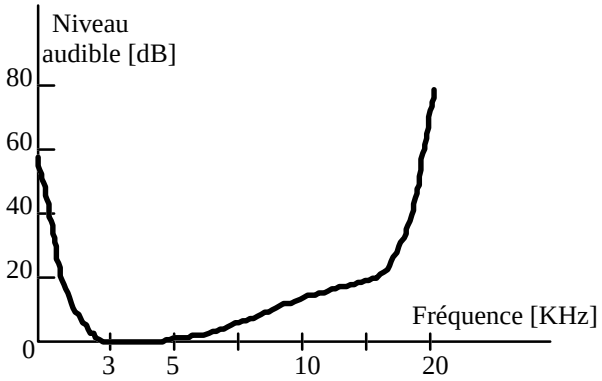
3° Les coefficients obtenus à l'étape antérieure sont comprimés par un algorithme de Huffman dynamique (adaptatif) à partir de tables de codes connues (afin d'éviter le stockage d'un en-tête pour chaque bloc).

Les taux de compression varient entre 20% et 2400% (en général 1000% ÷ 1600%).

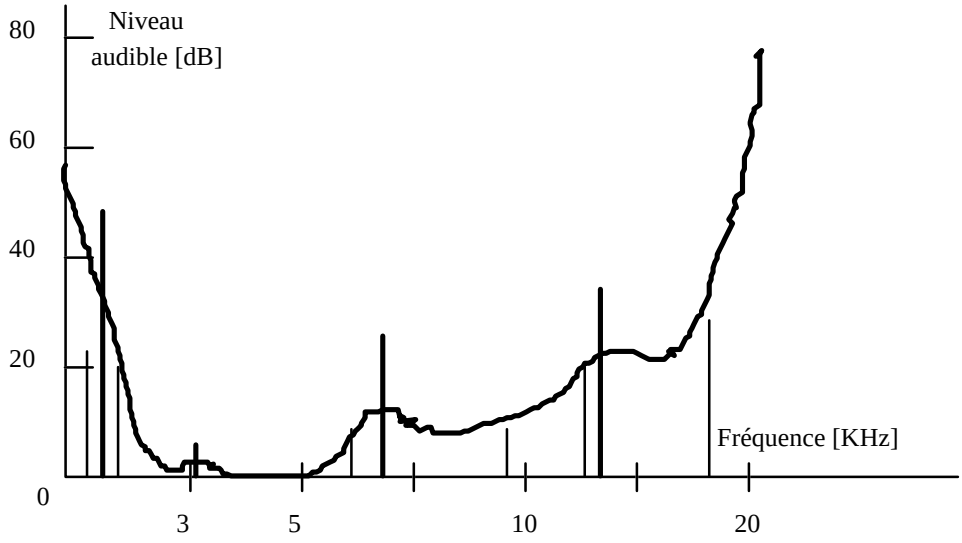
Pour des séquences d'images animées, une première technique consiste à comprimer avec JPEG chaque image de la séquence (M-JPEG). Des techniques plus évoluées ont été définies dans le cadre de MPEG ; quelques images de la séquence sont codées avec JPEG, pour les autres images on trouve les zones modifiées par rapport à la dernière image codée JPEG et les corrélations de mouvement entre ces zones ; des compressions de 15 à 20 fois sont obtenues.

4.7.2. Compression des sons

Les sons sont représentés par une suite d'échantillons obtenus à une certaine cadence (par exemple, pour les CD audio la fréquence d'échantillonnage est de 44 KHz et les échantillons sont codés sur 16 bits). Le système auditif humain est en général très sensible aux distorsions du signal sonore, mais sa courbe de sensibilité est proche de celle d'un filtre passe-bande :



De plus, si deux sons de fréquences relativement proches sont émis simultanément, nous n'entendons que le plus fort des deux sons. Cela correspond à un ajustement dynamique de la courbe de sensibilité, comme dans l'exemple :



4.7.2.1. Méthode PASC

Cette méthode (*Precision Adaptive Sub-band Coding*), employée pour les DCC, utilise les observations que nous avons faites. La compression se déroule selon les étapes suivantes :

1° Le signal subit une division spectrale sur 32 bandes de 750 Hz de largeur et sur chacune on calcule le niveau moyen.

2° En utilisant les connaissances sur les courbes dynamiques d'audibilité, on détermine dans quelles bandes le signal peut être éliminé.

3° Dans chaque bande retenue, le signal est codé en utilisant un nombre variable de bits répartis entre les sous-bandes selon la différence entre le niveau de la sous-bande et le niveau moyen dans la bande.

Le taux de compression est de 3 à 4 fois.

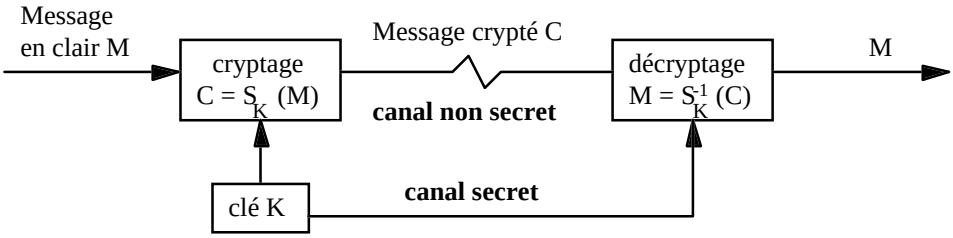
5. Cryptage des informations

5.1. Problèmes et classification

Différentes raisons rendent aujourd'hui incontournable le cryptage des informations stockées ou transmises : la facilité avec laquelle on peut accéder à des données à travers les réseaux, l'interception des communications, la nécessité de pouvoir authentifier les documents électroniques dans les transactions d'affaires.

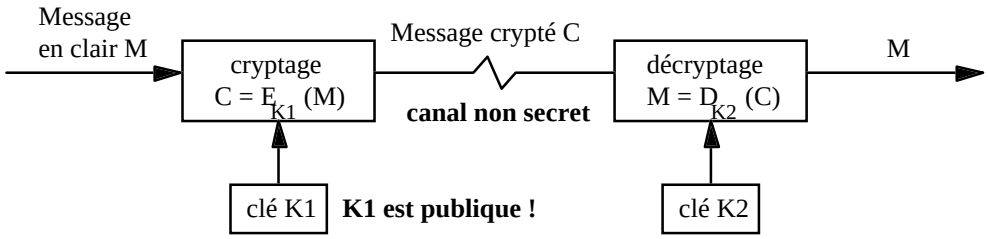
Deux grandes familles de cryptosystèmes :

1° Cryptosystèmes conventionnels. Les deux interlocuteurs doivent connaître la clé permettant de crypter et de décrypter les messages échangés à travers un canal de communication non secret (réseau de transmission). Cette clé doit être échangée à travers un canal de communication secret (comme un porteur privé), ce qui pose le plus souvent des problèmes de coût et de délai de transmission.



Différentes techniques, résistant plus ou moins bien à la cryptanalyse, peuvent être employées afin de coder un message à partir de la clé. Le cryptosystème conventionnel le plus utilisé actuellement, réputé sûr, est DES (*Data Encryption Standard* développé par IBM il y a 20 ans). DES est employé entre autres dans le système d'exploitation UNIX pour la protection des mots de passe.

2° Cryptosystèmes à clé publique. Le cryptage et le décryptage utilisent deux clés distinctes ; la clé de cryptage est publiée, en revanche la clé de décryptage est connue uniquement par le **destinataire** du message. La clé de décryptage ne peut pas être calculée, du moins **en l'état actuel des connaissances mathématiques**, à partir de la clé de cryptage en un temps raisonnable.



En employant des techniques similaires, les cryptosystèmes à **distribution de clé publique** laissent deux interlocuteurs aboutir à une clé commune (utilisable par la suite dans un cryptosystème conventionnel), tout en échangeant sur un canal non secret des informations ne permettant pas à une tierce personne de retrouver la clé. Les cryptosystèmes à clé publique permettent aussi de mettre en œuvre facilement des systèmes d'authentification. Les cryptosystèmes à clé publique les plus utilisés actuellement sont RSA (Rivest, Shamir, Adleman) et Diffie-Hellman.

La sûreté d'un système de cryptage repose sur :

1° La qualité du brouillage opéré par le cryptage. Si le brouillage est trop faible, des techniques de cryptanalyse (parfois même une simple analyse en fréquence) permettent de reconstituer le message en clair, sans connaître la clé ou l'algorithme de cryptage. Le cryptage opère de préférence sur des blocs de taille différente de l'octet et fait appel à des transformations fortement non linéaires.

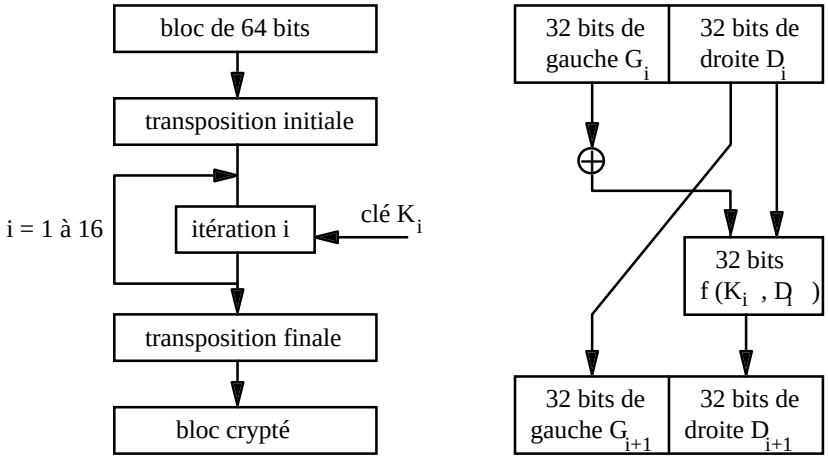
2° Le nombre de solutions possibles. Si la cryptanalyse est impuissante mais l'algorithme de cryptage est connu, il ne reste que le choix entre essayer toutes les solutions possibles (cryptosystèmes conventionnels) ou tenter de calculer la clé de décryptage à partir de la clé de cryptage (cryptosystèmes à clé publique). Ces calculs doivent donc être suffisamment complexes pour que le temps de calcul soit prohibitif, même avec la puissance de calcul maximale disponible.

5.2. Système DES

DES utilise transpositions, opérations OU-exclusif et substitutions afin de produire un cryptage qui résiste à la cryptanalyse. DES traite des blocs de 64 bits (8 octets) et emploie actuellement des clés de 64 ou 128 bits. Le

fichier à traiter est découpé en blocs de 64 bits, chaque bloc est crypté et les blocs obtenus sont réunis pour obtenir le fichier crypté. Algorithme de cryptage d'un bloc (voir [Marsault, 1992] pour implémentation en C) :

1° La permutation suivante est appliquée aux bits du bloc :
$(b_1, b_2, \ldots, b_{64}) \rightarrow (b_{58}, b_{50}, b_{42}, \ldots, b_k, b_{k-8}, \ldots)$.
Le bloc est ensuite découpé en 2 sous-blocs de 32 bits chacun, G_1 et D_1 ; $i := 1$.
Les opérations 2° à 6° sont itérées 16 fois ($i = 1-16$).



2° Les 32 bits de D_i sont mélangés et répétés pour obtenir 48 bits :
$(b'_1, b'_2, \ldots, b'_{32}) \rightarrow (b'_{32}, b'_1, b'_2, b'_3, b'_4, b'_5, b'_4, b'_5, b'_6, b'_7, b'_8, b'_9, b'_8, b'_9, \ldots)$.

3° La clé K_i est calculée à partir de la clé d'origine K et un OU-exclusif est effectué avec les 48 bits obtenus à l'étape précédente.

4° Le bloc 48 bits résultant est découpé en 8 sous-blocs de 6 bits chacun. Chaque sous-bloc de 6 bits permet de calculer une adresse dans une table de substitution (les 2 bits de poids fort indiquent le numéro de ligne et les 4 bits de poids faible le numéro de colonne) et est remplacé par les 4 bits se trouvant à cette adresse dans la table. On obtient donc 32 bits.

5° Le résultat de l'étape précédente subit une permutation :
$(b_1, b_2, \ldots, b_{32}) \rightarrow (b_{16,7,20,21,29,12,28,17,1,15,23,26,5,18,31,10,2,8,24,14,32,27,3,9,19,13,30,6,22,11,4,25})$.

6° $D_{i+1} := (\text{résultat étape } 5°) \oplus G_i$
$G_{i+1} := D_i$.

7° Le résultat de la dernière itération subit la permutation inverse à la permutation initiale.

L'algorithme de décryptage est **identique** à l'algorithme de cryptage et utilise les mêmes clés, mais en partant de K_{16}. Une technique de cryptanalyse récente, la **cryptanalyse différentielle** développée par Biham et Shamir, a montré certaines limites de DES et a invalidé plusieurs algorithmes inspirés par DES (comme FEAL-4, NewDES, GDES). La cryptanalyse différentielle part de deux hypothèses : 1° l'analyste peut avoir accès à un nombre élevé de textes en clair et aux textes cryptés correspondants, 2° la table de substitution employée pour l'étape 4° de l'algorithme est connue. Augmenter la longueur de la clé ne résout pas nécessairement le problème : il a été ainsi montré qu'une clé de longueur inférieure ou égale à 768 bits ne résiste pas à la cryptanalyse différentielle. Afin de faire face à cette technique, DES est actuellement employé en 3 passes successives, avec des clés de 112 ou 168 bits. D'autres versions de DES (comme Khufu, Blowfish), faisant appel à des tables de substitution multiples définies à partir de la clé, ont été développées dernièrement et sont censées résister aussi à la cryptanalyse différentielle.

5.3. **Système RSA** (Rivest, Shamir, Adleman)

Ce système est fondé sur le problème NP-complet de la décomposition en facteurs premiers d'un nombre. Si le nombre en question est choisi de taille importante (plusieurs centaines de chiffres), le temps de calcul est prohibitif. Afin de construire la clé publique et la clé secrète, le destinataire choisit au hasard deux grands nombres premiers distincts, *p* et *q*, de tailles sensiblement différentes et ayant la forme (2*x* + 1), avec *x* premier et tel que *x*-1 possède de grands facteurs premiers. Le destinataire calcule $n = pq$; le nombre d'entiers inférieurs à n et premiers avec lui, noté $\varphi(n)$, sera alors égal à (*p*-1)(*q*-1). Il choisit ensuite un grand nombre *E* premier avec

$\varphi(n)$. Il calcule *D*, l'inverse modulo $\varphi(n)$ de *E* et vérifie que *D* >> $\log_2 n$ et que *D* et *n* sont premiers entre eux. La clé publique — permettant le cryptage — sera la paire (*E*, *n*) et la clé secrète du destinataire — permettant le décryptage — sera *D* (*p* et *q* sont aussi gardés secrets par le destinataire). Pour qui ne connaît pas *p* et *q*, le calcul de la clé secrète à partir de la clé publique nécessite une décomposition de *n* en facteurs premiers, ce qui demande un temps de calcul prohibitif.

Afin de crypter le message, celui-ci est décomposé en blocs *Bi* sur *m* bits, tel que chaque nombre *Bi* soit dans l'intervalle $[0, n-1]$ et $2^{m+1} > n > 2^m - 1$. Chaque *Bi* est crypté par l'opération suivante : *Ci* = $Bi^E \bmod n$; chaque *Ci* est représenté sur $m+1$ bits. La suite des *Ci* est transmise sur le canal non secret. Pour décrypter le message reçu, le destinataire applique sur chaque bloc *Ci* l'opération inverse, qui est *Bi* = $Ci^D \bmod n$ (le destinataire est le seul à connaître *D*, la clé de décryptage).

Exemple simple (les valeurs utilisées ici sont trop petites pour assurer la protection !) :
Message en clair : 10001010111101001001011110100010…
Clé publique : E = 11, n = 11023.
Clé destinataire (secrète) : D = 5891 (p = 73, q = 151).

Taille des blocs : 13 bits car $2^{14} > 11023 > 2^{13} - 1$.
Le premier bloc de 13 bits est B1 = 1000101011110 = 4446, donc le bloc crypté est C1 = $B1^E \bmod n$ *= 9121 (à représenter sur 14 bits !)*

Le bloc décrypté par le destinataire est $C1^D \bmod n$ *= 4446 = B1.*
A partir de p, q, E et n, le calcul de D a été effectué de la façon suivante : n = pq, $\varphi(n) = (p-1)\cdot(q-1) = 10800$, $D = 11^{-1} \bmod 10800 = 5891$ *(10800 × **6** + 1 = 64801 = 11 × 5891).*

Remarques importantes :

1° Les nombres premiers connus n'étant pas très nombreux, il est important de ne pas utiliser trop souvent la même clé afin de décourager la cryptanalyse.

2° Si un algorithme rapide de factorisation était découvert, RSA serait définitivement abandonné.

Le développement récent des techniques de factorisation, des circuits spécialisés et du piratage du temps de calcul font que la longueur conseillée pour la clé de décryptage soit supérieure à 1000 bits pour 1995 et à 1500 bits à l'horizon 2000. Par exemple, la version actuelle du logiciel du domaine public PGP (*Pretty Good Privacy*) peut employer des clés de longueur maximale de 1280 bits.

5.3.1. Authentification

Un document électronique est jugé authentique si sa signature et son intégrité peuvent être vérifiées. Ces vérifications sont possibles grâce à des techniques de cryptage, notamment RSA qui est particulièrement adaptée. Afin de permettre la vérification de l'intégrité d'un document, son **créateur** calcule à partir du document un code redondant (détection des erreurs) de longueur suffisante. Il indique ensuite son identité à l'aide d'un code d'identification et crypte à l'aide de sa clé secrète *D* (qu'il est le seul à connaître) le code redondant et son code d'identification. Ce cryptage tient lieu de signature : en effet, tous ceux qui sont en possession de la clé publique (*E*, *n*) correspondante peuvent vérifier que le document est intègre et qu'il a bien été créé par le propriétaire de la clé et du code d'identification.

Pour que le document soit en plus confidentiel il faudra employer — sur tout le document et en utilisant la clé publique du **destinataire** — un cryptage préalable ou ultérieur au cryptage d'authentification.

5.3.2. Distribution de clé publique

La complexité algorithmique de la décomposition en facteurs premiers fournit un moyen pour générer une clé commune à deux utilisateurs (à utiliser avec des cryptosystèmes conventionnels, comme DES) en véhiculant entre les deux utilisateurs uniquement des informations non confidentielles, sur un canal non secret.

Pour cela, les deux utilisateurs choisissent un grand nombre premier *p* et un entier $a < p$ qu'ils échangent publiquement. Ensuite, chaque utilisateur choisit un entier *X* appartenant à [1, *p*-1] et calcule $Y = a^X \bmod p$. Les valeurs *Y1* et *Y2*, calculées par les deux utilisateurs, sont ensuite échangées publiquement. Chaque utilisateur peut alors calculer la clé commune $c = a^{X1 \cdot X2} \bmod p$ car :

$$c = Y1^{X2} \bmod p = Y2^{X1} \bmod p .$$

Une tierce personne connaissant uniquement *p*, *a*, *Y1* et *Y2* doit calculer un logarithme modulo *p* afin de retrouver un *X* et donc la clé, ce qui est prohibitif en termes de temps de calcul (étant données les hypothèses sur *p*).

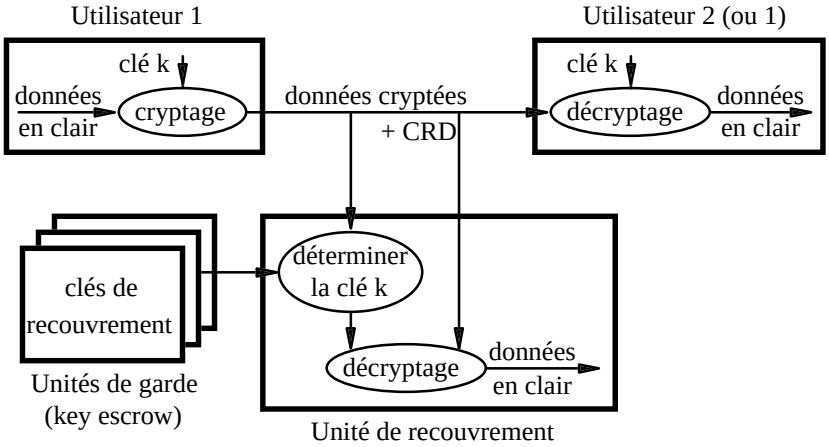
5.4. *Key Escrow*

L'utilisation du cryptage pour la transmission et la sauvegarde d'informations peut poser différents problèmes, notamment :

1° En cas de perte de sa clé, le destinataire ou le propriétaire ne dispose plus d'aucun moyen d'accès aux informations.

2° L'accès aux informations par un tiers autorisé est impossible sans l'accord du destinataire/propriétaire des informations. Le tiers autorisé peut être la direction de la société dans laquelle travaille le destinataire/propriétaire (en cas d'absence ou de mauvaise volonté) ou l'autorité judiciaire (en cas d'enquête).

La solution est d'associer à chaque système de cryptage une facilité de décryptage utilisable uniquement par des personnes ou organisations autorisées. Des solutions en nombre très important ont été proposées, mais elles respectent toutes le schéma suivant :



Un Champ de Recouvrement des Données (CRD) est transmis ou stocké avec les données cryptées. Ce champ contient des informations suffisantes pour le décryptage par un agent autorisé (l'Unité de recouvrement), comme par exemple la clé *k* cryptée à l'aide de la clé publique (*E*, *n*) qui correspond à une clé secrète *D* de l'Unité de garde.

Différentes précautions peuvent être prises afin d'améliorer la protection des utilisateurs face aux éventuels abus des Unités de garde ou des Unités de recouvrement, notamment :

1° Unités de garde multiples et indépendantes. Chaque unité détient un seul fragment de la clé et ne peut la communiquer qu'à l'unité de recouvrement (l'agent autorisé).

2° Grain fin pour les clés : le cryptage n'est pas effectué avec des clés à longue durée de vie, mais avec des clés de session. Les clés à longue durée de vie restent connues uniquement aux utilisateurs, les unités de garde permettent uniquement de récupérer les clés de session.

Le système *key escrow* le plus connu est celui proposé par l'administration des Etats Unis sous le nom de *Escrow Encryption Standard*, qui fait appel à un algorithme de cryptage confidentiel (*Skipjack*) implémenté dans des circuits intégrés spécialisés, inviolables (*Clipper/Capstone*). Il est toutefois peu probable que ce système puisse s'imposer car il tient compte uniquement des intérêts de l'administration...

Remarque : rien ne peut garantir que les informations transmises ou stockées à l'aide d'un système *key escrow* ne sont pas cryptées au préalable avec des clés connues uniquement aux utilisateurs !

Pour plus de détails voir (entre autres) les articles publiés à ce sujet dans [CACM 3/39, 1996] ainsi que le document `http://www.cosc.georgetown.edu/ndenning/crypto/appendix.html`.

6. Supports de transmission

6.1. Canal de transmission

6.1.1. Caractéristiques du signal

Les informations transmises sont représentées par des signaux. Afin de comprendre les limitations que des phénomènes physiques imposent sur la transmission des signaux, il est nécessaire de connaître, par une analyse mathématique, les caractéristiques des signaux.

Une fonction périodique quelconque $g(t)$ peut être écrite comme une somme de fonctions périodiques sinusoïdales (*série de Fourrier*), appelées *composantes harmoniques* :

$$g(t) = \frac{1}{2}c + \sum_{n=1}^{\infty} a_n \sin(2\pi nft) + \sum_{n=1}^{\infty} b_n \cos(2\pi nft)$$

où $f = 1/T$ représente la *fréquence fondamentale* du signal $g(t)$ et a_n et b_n sont les amplitudes des composantes harmoniques d'ordre *n*. La fréquence d'une harmonique est donc un multiple de la fréquence fondamentale. Un signal de durée *T* finie peut être analysé comme un signal périodique (de période *T*) en considérant que le signal reproduit périodiquement (*T*-2*T*, 2*T*-3*T*,...) la forme qu'il a dans l'intervalle 0-*T*.

Les amplitudes des composantes harmoniques peuvent être calculées selon :

$$a_n = \frac{2}{T}\int_0^T g(t)\sin(2\pi nft)dt\,,$$

$$b_n = \frac{2}{T}\int_0^T g(t)\cos(2\pi nft)dt\,,$$

$$c = \frac{2}{T}\int_0^T g(t)dt$$

(en effet, $\int_0^T \sin(2\pi kft)\sin(2\pi nft)dt = \begin{cases} 0 \text{ si } k \neq n \\ T/2 \text{ si } k = n \end{cases}$, etc.).

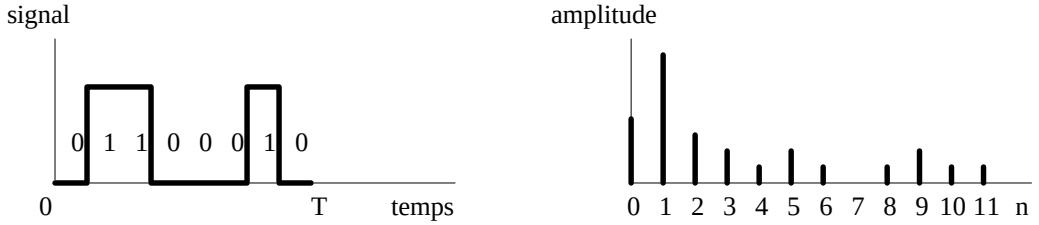
La suite des amplitudes des composantes harmoniques forme le *spectre* du signal. En général, c'est le spectre de puissance qui est employé : ses composantes sont $c_n = \sqrt{a_n^2 + b_n^2}$ et leur carré est proportionnel aux quantités d'énergie transmises par les signaux aux fréquences correspondantes.

Par exemple, l'analyse du signal qui représente la séquence binaire 0110 0010 (code ASCII 8 bits du caractère *b*) donne les coefficients suivants :

$$a_n = \frac{1}{n\pi}\left[\cos\left(\frac{n\pi}{4}\right) - \cos\left(\frac{3n\pi}{4}\right) + \cos\left(\frac{6n\pi}{4}\right) - \cos\left(\frac{7n\pi}{4}\right)\right],$$

$$b_n = \frac{1}{n\pi}\left[-\sin\left(\frac{n\pi}{4}\right) + \sin\left(\frac{3n\pi}{4}\right) - \sin\left(\frac{6n\pi}{4}\right) + \sin\left(\frac{7n\pi}{4}\right)\right], \qquad c = \frac{3}{4}.$$

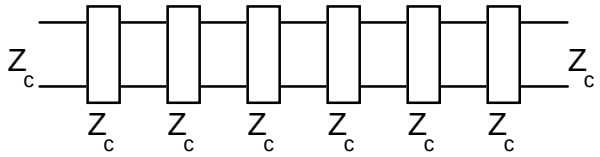
Les valeurs correspondantes pour c_n sont indiquées dans la figure suivante :



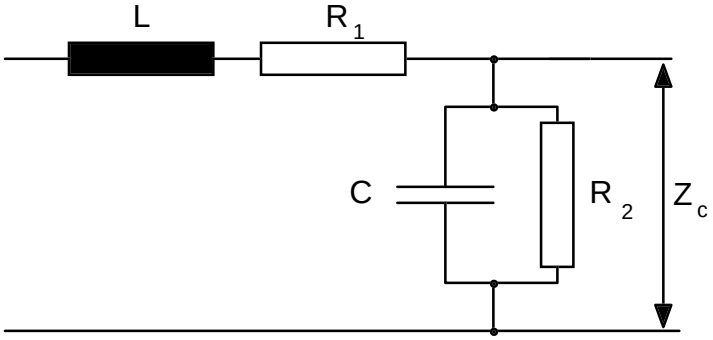
6.1.2. Caractéristiques du canal

Pour un canal qui transmet des signaux électriques, une caractéristique importante est l'*impédance caractéristique de la ligne*. En effet, la ligne peut être modélisée comme une suite de cellules élémentaires (une cellule par unité de longueur), à chaque cellule correspondant un circuit équivalent :

ligne = suite de cellules élémentaires



Cellule élémentaire - circuit équivalent :



La *résistance* R_1 de la ligne est proportionnelle à la longueur et la résistivité du matériel conducteur utilisé, et inversement proportionnelle à sa section. L'*inductance* L est proportionnelle à la longueur et dépend du diamètre des conducteurs, de leur espacement et de leur configuration. La *capacité* C de la ligne est proportionnelle à sa longueur et à la permittivité du milieu isolant et inversement proportionnelle à l'espacement entre les conducteurs. La *perditance* R_2 est due à l'imperfection du milieu isolant et est inversement proportionnelle à la longueur de la ligne ; la perditance est en partie dépendante de la fréquence du signal véhiculé.

Une ligne doit être refermée sur son impédance caractéristique et ne pas subir de variation locale d'impédance. Dans le cas contraire, des ondes réfléchies apparaissent, produisant une instabilité des signaux et une pollution électromagnétique de l'environnement.

Ce modèle de ligne permet de comprendre pourquoi tout signal transmis subit deux types de modifications qui dépendent de la longueur de la ligne et de la fréquence du signal :

1° un affaiblissement ;
2° un déphasage.

L'*affaiblissement linéique* (défini donc pour une certaine fréquence) est mesuré en décibels/unité de longueur (dB/km) :

$$A_l = 10 \log_{10} \frac{\text{énergie du signal initial}}{\text{énergie du signal après une unité de longueur (km)}}.$$

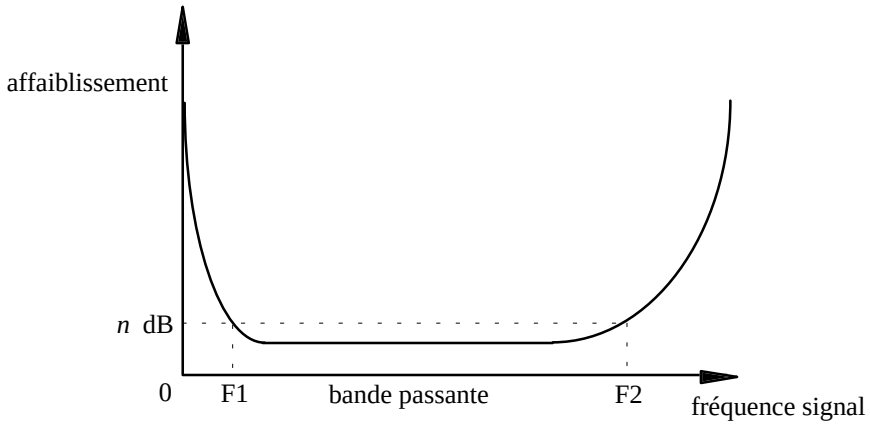
Pour faire face à ce problème, dans le cas d'une transmission analogique le signal est *amplifié* et dans le cas d'une transmission numérique le signal est *régénéré*.

Une autre caractéristique importante du canal est la vitesse de propagation du signal (peut dépendre de la fréquence du signal). La vitesse de propagation usuelle des signaux électriques sur câble bifilaire ou coaxial est de 150 000 à 220 000 km/s.

Enfin, la sensibilité aux parasites impose des contraintes fortes sur la longueur d'une ligne de transmission et sur l'environnement dans lequel celle-ci peut être employée. Types de parasites : le *bruit blanc*, dû à l'agitation thermique, peu gênant pour la transmission de données (permanent mais de faible amplitude, de moyenne nulle et de densité spectrale constante) ; *bruits impulsifs*, dus à des phénomènes de diaphonie, décharges électriques, commutation mécanique, etc., provoquent souvent des erreurs de transmission (forte intensité et durée brève, mais suffisante pour couvrir un long segment de données sur une ligne à haut débit). La sensibilité aux parasites électromagnétiques (bruits impulsifs) est importante pour un câble bifilaire sans blindage, se réduit sensiblement par l'utilisation d'un blindage ou d'un câble coaxial et est pratiquement nulle pour la fibre optique.

6.1.3. Bande passante, rapidité de modulation et capacité de transmission

L'affaiblissement du signal sur une ligne est en général considéré négligeable pour les fréquences comprises entre deux fréquences-limite — définissant la *bande passante H = F2 – F1*, mesurée en Hz, de la ligne — et augmente rapidement en dehors de ces limites :



La bande passante se définit par rapport à un affaiblissement admissible, souvent $A = 3\text{dB}$ (ceci correspond donc à une baisse admissible de 2 fois de l'énergie du signal, donc à une baisse de 1,4 fois de son amplitude).

Toutes les composantes harmoniques d'un signal ne subissent ni le même affaiblissement, ni le même déphasage, donc le signal reconstruit à l'arrivée n'a pas la même forme que le signal émis. La valeur de la bande passante *H* de la ligne impose donc une limite sur la rapidité à laquelle sont effectués les changements d'état significatifs du signal — appelée *rapidité de modulation* (*R*) ou *vitesse de signalisation* et mesurée en *bauds* — représentant l'information à transmettre (plus ces changements sont rapides, plus la bande passante exigée du canal pour que le signal émis puisse être reconstitué à la réception est large). Relation de Nyquist :

$$R = 2H\ .$$

Souvent, le nombre d'états significatifs du signal (appelé aussi *valence*) est $V > 2$; dans ce cas, le *débit binaire* (en bit/s) de la ligne est donné par :

$$D_{\max} = 2H \log_2 V\ .$$

La quantité de bruit présent sur une ligne de transmission peut être quantifiée en utilisant le rapport entre l'énergie utile du signal et l'énergie du bruit, S/B (rapport exprimé en général comme $10 \log_{10} S/B$, en dB). Pour une ligne de transmission sensible au bruit, Shannon montre que la valence maximale utilisable dépend du rapport signal/bruit selon :

$$V = \sqrt{1 + \frac{S}{B}}\ ,$$

et donc le débit résultant, appelé *capacité de transmission* (*C*, en bit/s), est :

$$C = H \log_2 \left(1 + \frac{S}{B}\right).$$

Par exemple, pour le réseau téléphonique commuté (RTC), la bande passante à 3 dB est de 3,1 kHz (300 Hz à 3400 Hz) et le rapport signal/bruit de 30 dB (donc S/B = 1000) ; la capacité de transmission est alors ~ 28 000 bit/s.

Il faut remarquer que le rapport signal/bruit, la bande passante et par conséquence la capacité de transmission dépendent de la longueur de la ligne.

6.2. Transmission sur supports filaires en cuivre

Les supports en cuivre employés sont la paire torsadée et le câble coaxial.

6.2.1. Supports bifilaires (symétriques)

La *paire torsadée* est le support de transmission le plus ancien et encore le plus largement utilisé, principalement pour les services téléphoniques. La paire torsadée est composée de deux conducteurs en cuivre, isolés l'un de

l'autre, et enroulés de façon hélicoïdale autour de l'axe longitudinale. L'affaiblissement croit rapidement avec la longueur du support.

Le débit binaire accessible dépend de la qualité du câble et de sa longueur ; il peut varier entre quelques dizaines de Kbit/s sur quelques dizains de km, quelques Mbit/s sur quelques km et plusieurs centaines de Mbit/s pour quelques centaines de mètres. La sensibilité aux parasites d'origine électromagnétique est relativement importante mais peut être réduite si le câble est blindé. Enfin, le taux d'erreur est de l'ordre de 10^{-5}. Le rayonnement électromagnétique d'un câble non blindé permet l'écoute de la communication.

L'importance de l'infrastructure en paire torsadée au niveau de la boucle locale des réseaux de télécommunication et du câblage des immeubles, ainsi que l'évolution des techniques de transmission sur des paires torsadées (ADSL, Gigabit Ethernet), font que son remplacement généralisé, par d'autres supports, ne soit pas envisagé à court terme.

6.2.2. Support coaxial

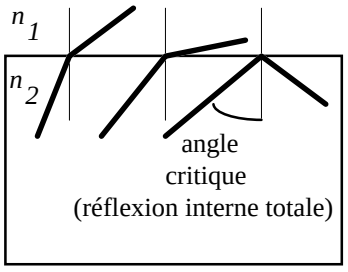
Plus cher que la paire torsadée, le câble coaxial est encore largement utilisé pour des artères à moyen débit des réseaux de transport, ainsi que pour les réseaux de télédiffusion. Deux types de câbles sont les plus utilisés, le câble à impédance caractéristique de 50 ohms (notamment pour les réseaux locaux) et le câble à impédance de 75 ohms (télédiffusion, artères internes aux réseaux téléphoniques interurbains et internationaux).

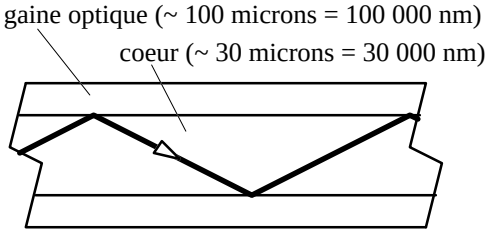
La bande passante peut atteindre 400 MHz sur plusieurs dizaines de km. Le débit binaire typiquement employé est de 10 Mbit/s (réseaux Ethernet) sur des distances inférieures à 1km et peut monter jusqu'à plusieurs centaines de Mbit/s sur des distances très courtes.

La sensibilité aux parasites ainsi que l'affaiblissement sont réduits par rapport à la paire torsadée (mais le prix est significativement plus élevé). Le taux d'erreur est de l'ordre de 10^{-7}. Le câble coaxial est progressivement remplacé par la fibre optique.

6.3. Transmission par fibre optique

Les signaux binaires sont transmis sous la forme d'impulsions lumineuses, à travers un guide d'onde en fibre de verre. Afin de maintenir les rayons lumineux à l'intérieur du fibre optique, le phénomène de réflexion totale est employé :





L'indice de réfraction de la gaine (n_1) doit être inférieur à celui du cœur (n_2). L'angle critique est donné par la formule

$$\theta_c = \arcsin \frac{n_1}{n_2}$$

(dans l'équation de la réfraction $n_1 \sin\theta_1 = n_2 \sin\theta_2$, $\theta_2 = \theta_c$ quand $\theta_1 = 90°$).

Afin de subir uniquement des réflexions totales dans la fibre, un rayon lumineux en provenance d'une source (diode électroluminescente pour des longueurs d'onde 800÷900 nm, diode laser pour 800÷1300 nm) doit atteindre le bout de la fibre sous un angle d'incidence inférieur à

$$\theta_A = \arcsin \frac{\sqrt{n_2^2 - n_1^2}}{n_0}$$, n_0 étant l'indice de réfraction de l'air.

Tous les rayons qui dépassent l'angle critique subissent une réflexion totale ; ce sont donc en général plusieurs rayons, correspondant au même signal, qui se propagent à l'intérieur de la fibre optique — fibre *multimode* à saut d'indice (dessins précédents) ou à gradient d'indice. Quand le diamètre du cœur de la fibre est tellement réduit (comparable à la longueur d'onde de la lumière utilisée) qu'un seul rayon peut se propager, la fibre est appelée *monomode*.

Exercice : *calculer la rapidité de modulation maximale admise pour la transmission sur une fibre optique multimode à saut d'indice sachant que la longueur du câble est de 1000 m, la vitesse de propagation de la lumière dans la fibre est de 200 000 km/s et le rapport entre indices de réfraction est de 1,02.*

Les débits binaires varient entre plusieurs centaines de Mbit/s (fibre multimode, plusieurs km) et environ 10 Gbit/s (fibre monomode, jusqu'à 100 km). L'affaiblissement est très réduit, 0,2-0,8 dB/km, donc les transmissions sans répéteurs sur des distances de 100 à 200 km sont courantes. L'étude de l'affaiblissement dans une fibre de silice fait apparaître 3 minima : 1dB/km pour une longueur d'onde de 850 nm, 0,35 dB/km pour 1300 nm et 0,2 dB/km pour 1550 nm.

La fibre optique est insensible aux parasites d'origine électromagnétique et assure un taux d'erreur très bas, de l'ordre de 10^{-12}. Aussi, la fibre optique ne produit pas de rayonnement électromagnétique, ce qui contribue à la confidentialité des transmissions.

Grâce à ses avantages, la fibre optique non seulement remplace le câble coaxial sur les artères (*backbones*) des réseaux de télécommunication, mais s'impose aussi pour l'infrastructure des réseaux locaux à haut débit (Gigabit Ethernet) et est parfois employée pour la boucle locale (FTTO, *Fiber To The Office*). Le câble en fibre optique reste en revanche plus cher que le câble coaxial et son installation pose des difficultés supplémentaires. Après avoir été largement employées il y a quelques années, les fibres multimode sont en cours de remplacement aujourd'hui par des fibres monomode.

6.3.1. *Wavelength Division Multiplexing*

La capacité de transmission des fibres optiques peut être sensiblement augmentée en utilisant le multiplexage de longueurs d'onde, ou *Wavelength Division Multiplexing*, WDM. A l'émission on fait correspondre à chaque flot d'informations une longueur d'onde différente. Les rayons lumineux n'interagissent pas dans la fibre, et la longueur d'onde d'un des rayons n'évolue pas. En conséquence, à la réception on peut extraire chaque flot d'informations par un simple démultiplexage optique. Un multiplexage de 8 longueurs d'onde (8 lambdas) est couramment employé, et des expérimentations sont en cours avec 160 longueurs d'onde différentes (**DWDM**, *Dense* WDM, ou **HDWDM**, *High Density* WDM). Cela permet donc d'atteindre des débits binaires allant de 20 Gbit/s à 400 Gbit/s par fibre. Voir aussi http://www.techguide.com/comm/dwave.shtml.

6.4. Les ondes en transmission à vue directe

6.4.1. Transmissions par rayons laser

Des faisceaux laser très directifs peuvent être employés comme support pour des transmissions de données entre immeubles proches. Les débits peuvent être très importants (comme pour la fibre optique) et l'absence de support à installer présente l'avantage d'un coût nettement moins élevé. En revanche, les conditions météorologiques peuvent affecter dans des cas extrêmes la qualité des communications.

6.4.2. Transmissions par faisceaux hertziens

Pour des distances plus importantes, mais toujours à vue directe (dizaines de km, en fonction de la hauteur des antennes), des faisceaux dirigés d'ondes radio peuvent être employés et présentent le même avantage de coût d'installation réduit. Les transmissions sont à transposition de fréquence, la plage de fréquences employé pour la porteuse étant de 2 à 40 GHz. Il faut noter que l'atténuation du signal émis augmente fortement avec la fréquence de la porteuse. Les émetteurs utilisés pour les télécommunications sont de faible puissance (1 W).

La dispersion des faisceaux étant relativement importante, des techniques de cryptage doivent être employées afin de maintenir la confidentialité des communications.

6.5. Transmissions par satellite

Les transmissions par satellite emploient les satellites *géostationnaires*, qui se trouvent sur une orbite à 36000 km altitude au-dessus de l'équateur. Les bandes de fréquences attribuées aux réseaux de communications par satellite sont 3,7-4,2 GHz, 5,925-6,425 GHz, 12-14 GHz et 20-30GHz. Dans les deux premières bandes, l'écart de position entre deux satellites doit être supérieur à 4° (ou 8° pour les satellites de télédiffusion, de puissance plus élevée) afin d'assurer une bonne sélectivité (éviter les interférences). Les satellites de télécommunication possèdent des émetteurs de faible puissance (< 10 W) par rapport aux satellites de télédiffusion (> 500 W). Dans la troisième bande, les écarts angulaires peuvent être de seulement 1° mais l'atténuation dans l'atmosphère des signaux est beaucoup plus forte (surtout dans les particules d'eau). De façon générale, les conditions atmosphériques au sol ou en altitude peuvent affecter temporairement la qualité des communications. La quatrième bande commence seulement à être utilisée.

Les débits accessibles aux utilisateurs peuvent aller jusqu'à plusieurs Mbit/s.

Les délais de transmission sont relativement importants (250-300 milisecondes) et doivent être pris en compte dans la conception des protocoles de communication (notamment pour la correction des erreurs par retransmission).

La dispersion des faisceaux au sol étant importante, des techniques de cryptage sont indispensables pour maintenir la confidentialité des communications.

7. Types de transmission

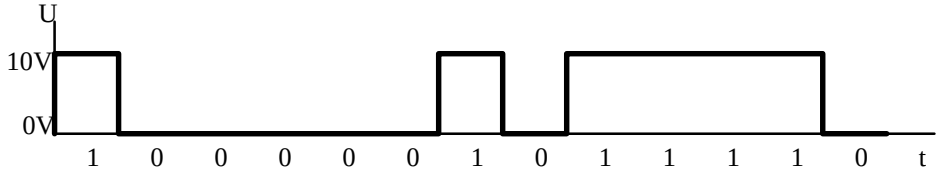
7.1. Bande de base ou transposition de fréquence

Dans ce qui suit nous appellerons *fréquence de bit* (notée *b*) la fréquence à laquelle les bits successifs sont transmis.

Les informations à transmettre sont représentées par une suite de symboles binaires. Un codeur transforme cette suite en une autre, binaire ou non, en employant un codage spécifique au canal. La suite codée à nouveau peut soit correspondre directement au signal qui circule sur le canal de communication — transmission en *bande de base* — soit être employée pour modifier (moduler) un signal (*porteuse*) de fréquence supérieure à la fréquence de bit — transmission par *transposition de fréquence* ou *modulation*.

7.1.1. Transmission en bande de base. Codage du signal

Considérons d'abord un exemple où les informations sont transmises par un signal binaire correspondant au codage source (sans codage canal) :



Problèmes qui se posent (exemple considéré) :

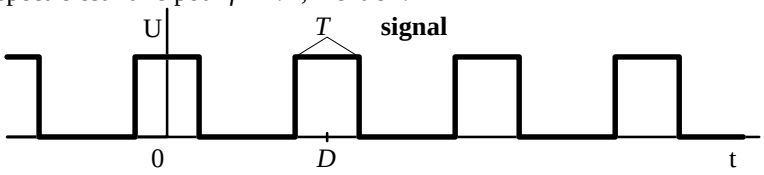
1° Une composante continue de 5 Volts est présente dans le signal, donc la moitié de l'énergie transmise est inutile.
2° La présence d'un nombre important de 0 successifs (ou de 1 successifs) dans le signal peut nuire à la synchronisation entre l'émetteur et le récepteur.
3° La présence possible à la fois de séquences de type 10101010... (alternantes) et de séquences de type 100000..001.. (constantes) fait que le spectre du signal — et donc la bande passante exigée au canal — soit très large.

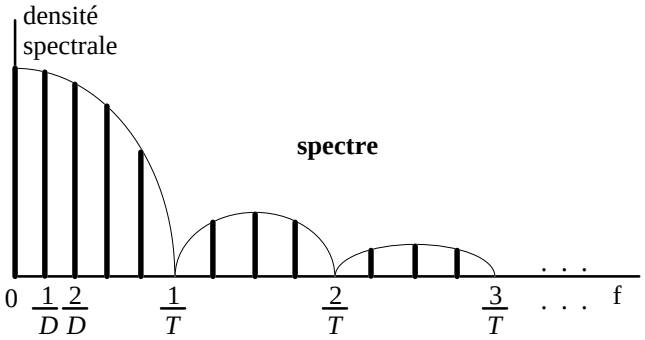
Le codage canal essaye de résoudre ou d'atténuer ces problèmes par :

1° une meilleure adaptation aux caractéristiques physiques du canal de transmission.
2° une meilleure synchronisation entre l'émetteur et le récepteur.
3° la réduction de la bande passante exigée.

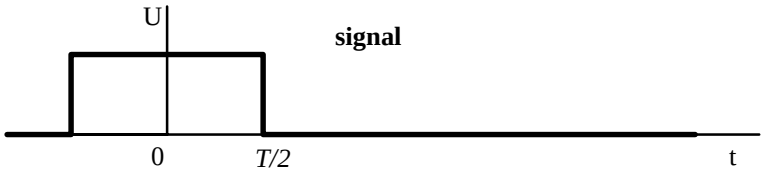
7.1.1.1. Spectre de fréquences d'un signal binaire

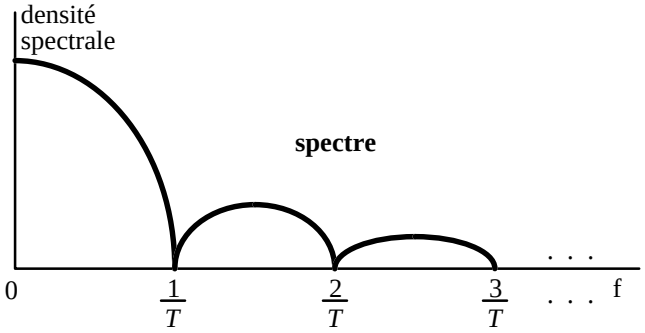
Train périodique d'impulsions (D = *k* T, *k* entier) : le spectre est un spectre de raies espacées de 1/*D*, l'amplitude des composantes du spectre est nulle pour *f* = *n*/*T*, *n* entier :





Impulsion unique : le spectre correspond à l'enveloppe du spectre précédent.





Suite aléatoire d'impulsions : représente le mieux la réalité pour une transmission de données ; le spectre de la suite aléatoire correspond, à une constante près, au spectre d'une impulsion unique. Il suffit de transmettre une bande de fréquences comprenant le premier zéro de ce spectre pour que la quantité d'énergie transférée soit suffisante.

7.1.1.2. Codages NRZ et NRZI

Ce codage est similaire au codage binaire, sauf pour les niveaux de tension symétriques employés afin de réduire la composante continue du signal transmis :

NRZ (*No Return to Zero*) : chaque information binaire "1" est codée par un niveau +*a*, les informations "0" par un niveau –*a* (voir la figure suivante).

NRZI (*No Return to Zero, Invert on one*) : chaque bit à "1" produit une inversion (+*a* →–*a*, –*a* → +*a*), les bits "0" laissent le signal inchangé.

Malheureusement, les problèmes 2° et 3° sont toujours présents. Le spectre de puissance du signal NRZ est :

$$\gamma(f) = b \cdot \left[\frac{a \cdot \sin(\pi \cdot f/b)}{\pi \cdot f} \right]^2 .$$

7.1.1.3. Codage Manchester

Ce codage prend en compte les problèmes 1° et 2° mentionnés. D'abord, des niveaux de tension symétriques sont employés afin de réduire la composante continue du signal transmis. Ensuite, une transition est présente au milieu du temps-bit pour chaque bit transmis (codage biphase) afin d'améliorer la synchronisation (transition-horloge) ; si le bit est "1" la transition est ascendante, si le bit est "0" la transition est descendante (voir la figure suivante). Le spectre de puissance d'un signal Manchester est :

$$\gamma(f) = b \cdot \left[\frac{2a}{\pi \cdot f}\right]^2 \cdot \sin^4\left(\frac{\pi \cdot f}{2b}\right).$$

7.1.1.4. Codage Manchester différentiel

Les transitions au milieu du temps-bit codent maintenant la différence entre deux bits successifs : la transition est ascendante lorsque les deux bits sont identiques et descendante dans le cas contraire (voir la figure suivante). Le spectre de puissance est le même que pour le codage Manchester.

L'absence de transitions-horloge peut être employée pour la signalisation au cours de la transmission (par exemple marqueurs de début ou de fin sur les réseaux à circulation de jeton — *Token Ring*, recommandation IEEE 802.5).

7.1.1.5. Codage de Miller

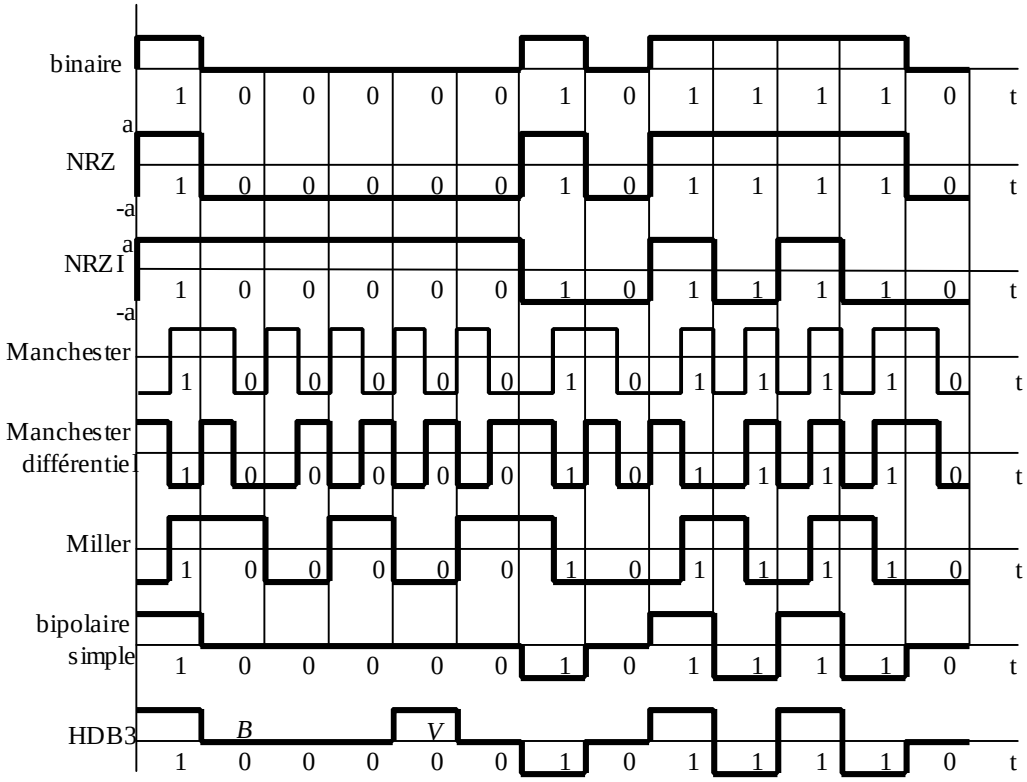
Des niveaux de tension symétriques sont employés afin de réduire la composante continue du signal transmis. Le codage de Miller consiste à mettre une transition au milieu du temps-bit si le bit est "1", la transition suivante ayant lieu après le temps-bit suivant si celui-ci correspond à un "0" suivi par un deuxième "0" ou au milieu du temps-bit suivant si celui-ci correspond à "1" (voir la figure suivante). Ce codage peut être obtenu à partir du codage Manchester en supprimant une transition sur deux ; le spectre de fréquence est beaucoup plus étroit que pour le codage Manchester.

7.1.1.6. Codages bipolaires

Ces codages utilisent trois états (niveaux de tension électrique) pour le signal. En codage bipolaire simple les bits "1" sont mis alternativement à l'état positif ou négatif et les bits "0" à l'état zéro (voir la figure suivante). Le spectre de puissance est :

$$\gamma(f) = b \cdot \left[\frac{a}{\pi \cdot f}\right]^2 \cdot \sin^4\left(\frac{\pi \cdot f}{b}\right),$$

donc deux fois plus étroit que pour le codage Manchester. Lors d'une longue série de "0", ce codage pose des problèmes de synchronisation (absence de transitions). Les codes bipolaires à haute densité d'ordre *n* (ou HDB*n*) corrigent cela en codant le (*n*+1)-ème bit "0" par un état (positif ou négatif) identique au dernier bit "1". Pour HDB3 (voir la figure suivante), par exemple, une séquence de 4 bits "0" est codée par la séquence "B00V" ; V est le bit dit de viol, dont l'état est identique à l'état du dernier bit "1" si B est zéro, ou au bit B sinon ; B est un bit dit de bourrage, dont l'état — positif, négatif ou zéro — est choisi de façon à ramener la composante continue à une valeur nulle. Le spectre utilise la même bande de fréquences que le code bipolaire simple.



Le choix entre les différents codages est effectué en fonction des caractéristiques du canal, du type de transmission et du débit binaire exigé. Le seul codage non redondant est NRZ/NRZI, mais il pose d'importants problèmes de synchronisation. Le codage Manchester, le codage de Miller et le codage bipolaire sont plus sensibles au bruit (pour Manchester, le spectre est deux fois plus large ; pour Miller, l'annulation de la composante continue n'est pas totale ; pour les codes bipolaires trois niveaux de tension sont employés). Les plus utilisés pour des transmissions synchrones sont les codages Manchester différentiel, Miller et HDB*n*.

La transmission en bande de base présente l'avantage de la simplicité et donc du coût réduit des équipements. Elle exige en revanche des supports n'introduisant pas de décalage en fréquence (transmission sur câble).

7.1.2. Transmission par transposition de fréquence

La transposition de fréquence est un ensemble de procédés par lesquels la bande de fréquence d'un signal est décalée dans le domaine fréquence. La transmission par transposition de fréquence assure en général une meilleure protection contre le bruit et permet le multiplexage en fréquence (voir plus loin). La transposition de fréquence devient indispensable quand le signal à transmettre n'est pas dans un domaine de fréquence correspondant au support.

Considérons le signal modulant $s(t)$ (en général un signal en bande de base) à bande limitée $[-B_s, +B_s]$, caractérisé par la densité spectrale de puissance $\gamma_s(f)$, et une onde porteuse sinusoïdale

$$p(t) = A_p \cos(\omega_p \cdot t + \varphi_p)$$

d'amplitude A_p, fréquence $f_p = \frac{\omega_p}{2\pi}$ et phase φ_p. Le signal modulé sera

$$x(t) = a(t) \cdot \cos \phi(x).$$

La modulation est d'*amplitude* si le signal modulant entre dans $a(t)$, *angulaire* s'il entre dans $\phi(t)$ et combinée s'il est présent dans les deux. Pour une modulation angulaire, la phase se décompose

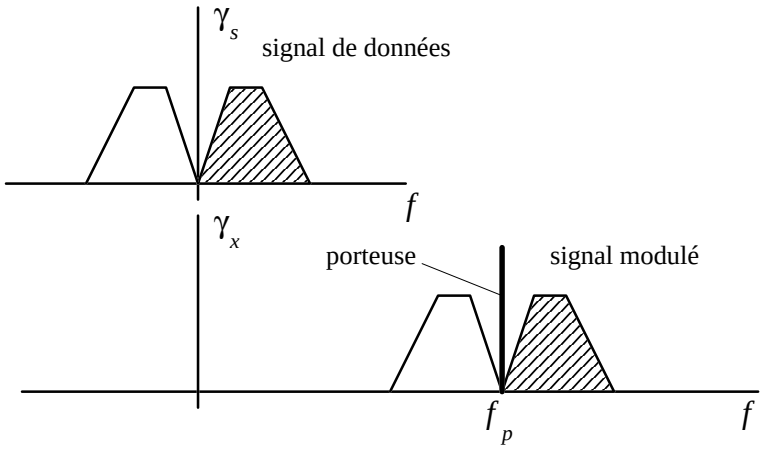
$\phi(t) = \omega_p \cdot t + \varphi_p + \varphi(t)$; pour une modulation de *phase* $\varphi(t) = m \cdot s(t)$ alors que pour une modulation de *fréquence* $\varphi(t) = m \cdot \int_0^t s(\tau) d\tau$, m étant une constante.

7.1.2.1. Modulation d'amplitude

L'amplitude instantanée du signal modulé présente une dépendance affine du signal de données modulant :

$$x(t) = A_p \cdot [K + m \cdot s(t)] \cdot \cos(\omega_p \cdot t + \varphi_p), \text{ avec } m \cdot s(t) \geq K ,$$

m étant le *taux de modulation*. Le spectre de puissance contient une raie correspondant à la porteuse à laquelle s'ajoute le spectre décalé du signal modulant :



Cette modulation est appelée à double bande car son occupation spectrale est double par rapport à celle du signal modulant. Afin de réduire la largeur de bande occupée sur le canal de transmission, différentes techniques peuvent être employées pour éliminer la moitié inférieure, redondante, du spectre (par exemple un filtre passe-bande) et obtenir une modulation à bande latérale unique (BLU).

La modulation d'amplitude est utilisée pour la radiodiffusion, le signal multiplex téléphonique des groupes primaires (signaux modulants analogiques) et la transmission de signaux numériques (codage Manchester préalable) sur des faisceaux hertziens locaux à 2 GHz.

7.1.2.2. Modulation de fréquence

La fréquence du signal modulé dépend du signal de données modulant :

$$x(t) = A_p \cdot \cos\left[\omega_p \cdot t + \varphi_p + 2\pi \cdot \Delta f \cdot \int_0^t s(\tau) d\tau\right],$$

Δf étant l'excursion en fréquence. Le calcul du spectre du signal modulé est complexe ; pour un signal modulant sinusoïdal de fréquence f_s, le spectre contient une infinité de raies aux fréquences $f_p \pm k \cdot f_s$. Les amplitudes des bandes latérales convergent rapidement vers 0, mais la largeur de bande exigée est toutefois plus importante que pour une modulation en amplitude.

Avantages de la modulation de fréquence : limitation du bruit de souffle (environ 1000 fois plus réduit qu'en modulation d'amplitude), distorsion inférieure à 1%. Désavantages : spectre de fréquences occupé beaucoup plus large qu'en modulation d'amplitude (environ 1 fois), d'où la nécessité d'utiliser des porteuses à fréquence élevée, ce qui implique une transmission quasi-optique des ondes (donc nombreux relais de transmission nécessaires), équipements coûteux.

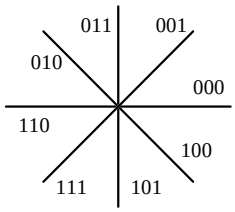
La modulation de fréquence est employée principalement pour des signaux modulants analogiques (radiodiffusion, télédiffusion, multiplex téléphoniques). En raison de la largeur de bande exigée, la modulation de fréquence est limitée à des débits faibles pour la transmission de signaux modulants numériques.

7.1.2.3. Modulation de phase

La phase du signal modulé dépend du signal de données modulant :

$$x(t) = A_p \cdot \cos\left[\omega_p \cdot t + \varphi_p + 2\pi \cdot m \cdot s(t)\right].$$

Pour un signal modulant numérique on utilise 2, 4 ou 8 états de phase uniformément répartis dans l'intervalle $[0, 2\pi]$. Afin de minimiser la probabilité d'erreur par élément binaire, la relation entre les états de phase et les suites binaires doit correspondre à un code de Gray (un seul bit change entre deux codes voisins), par exemple, pour 8 états de phase :



Lorsque le rapport entre la fréquence de la porteuse f_p et celle du signal modulant est grand, le spectre est concentré autour de f_p.

La modulation de phase est couramment employée (seule ou en conjonction avec la modulation d'amplitude, voir le paragraphe suivant) pour la transmission des signaux numériques sur lignes téléphoniques, pour les faisceaux hertziens à 2 GHz, pour les transmissions par satellite (modulation de phase à 4 états, qui présente un bon compromis puissance – efficacité spectrale).

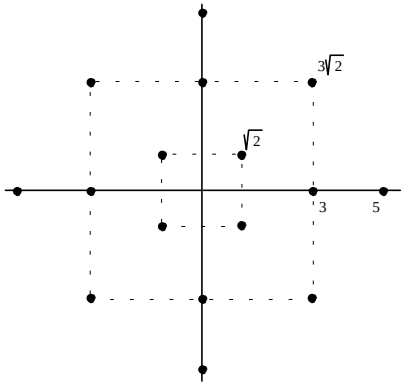
7.1.2.4. Modulations combinées d'amplitude et de phase

Les états du signal numérique peuvent être associés à plusieurs caractéristiques du signal modulé ; l'utilisation conjointe de l'amplitude et de la phase a été retenue. Sur un intervalle de modulation T (intervalle durant lequel les caractéristiques du signal modulé ne changent pas), le signal modulé sera

$$x(t) = A_i \cdot \cos\left(\omega_p \cdot t + \varphi_i\right) \text{ sur l'intervalle } [iT, (i+1)T]$$

où A_i et φ_i dépendent d'une *séquence d'états* du signal numérique (une suite de bits à transmettre).

Pour l'avis V29 du CCITT qu'utilisent certains modems sur le réseau téléphonique commuté, la rapidité de modulation est de 2400 bauds et le débit binaire de 9600 bits/s ; ce qui signifie que chaque intervalle de modulation code 4 bits consécutifs (16 combinaisons possibles). Ceci est possible grâce à l'emploi d'une modulation combinée d'amplitude et de phase, pour laquelle le diagramme suivant a été retenu :



La modulation combinée d'amplitude et de phase présente une bonne efficacité spectrale pour un rapport signal/bruit donné.

7.1.2.5. Principaux avis du CCITT (actuel UIT-T) concernant les modems

V21 : modem pour utilisation sur le réseau téléphonique commuté (RTC) à 300 bits/s (ou bps).
V22 : modem 1200 bps duplex intégral bifilaire pour RTC.
V22 bis : 2400 bps duplex intégral bifilaire sur RTC.
V23 : modem RTC, 600 ou 1200 bps sémi-duplex ou 1200/75 bps duplex (employé pour Télétel).
V25, V25 bis : composition automatique du numéro et/ou réponse automatique à un appel sur RTC.
V26 : modem 2400 bps duplex pour circuits quadrifilaires point à point.

V26 bis : modem 2400 bps (1200 en repli) duplex pour RTC.
V27 : modem 4800 bps pour lignes spécialisées.
V27 bis : modem 4800 bps (2400 en repli) pour transmissions synchrones ; égaliseur adaptatif automatique ; duplex symétrique sur ligne spécialisée quadrifilaire ou sémi-duplex dissymétrique — voie de retour à 75 bps — sur ligne spécialisée bifilaire.
V27 ter : identique au précédent, mais pour RTC.
V29 : modem 9600 bps pour lignes spécialisées.
V32 : modem 9600 bps (4800 en repli) duplex sur ligne bifilaire RTC.
V32 bis : modem 14400 bps duplex pour ligne bifilaire ; vitesses de repli 12000, 9600, 7200 et 4800 bps ; intègre une fonction d'auto-adaptation permanente du débit aux conditions de la ligne, par négociation entre deux modems V32 bis.
V42 : protocole de correction d'erreurs pour le transfert de données asynchrones.
V42 bis : algorithme normalisé de compression de données.
V34 (Vfast) : modem 28800 bps, les autres caractéristiques étant celles du V32 bis.
V54 : normalisation des boucles de test.

7.1.3. Modulation par impulsion et codage (MIC)

La transmission numérique de signaux analogiques fait appel à une technique appelée modulation par impulsion et codage (MIC) qui consiste en trois étapes successives : échantillonnage, quantification et codage.

Le signal analogique $x(t)$ est échantillonné à une fréquence $f_e = 1/T$, ce qui correspond à la transformation de $x(t)$ en $\tilde{x}(t) = x(t) \cdot \delta_T(t)$. $\delta_T(t)$ est la fonction Delta périodique ("peigne") de période T, $\delta_T(t) = \sum_k \delta(t - k \cdot T)$, δ étant la distribution de Dirac. En série de Fourrier complexe, $\delta_T(t)$ peut s'écrire

$$\delta_T(t) = \frac{1}{T} \sum_k e^{2j\pi kt/T} .$$

Le spectre de $\tilde{x}(t)$ est donné par

$$\tilde{X}(f) = \int_{-\infty}^{+\infty} \tilde{x}(t) \cdot e^{-2j\pi ft} dt = \int_{-\infty}^{+\infty} x(t) \cdot \delta_T(t) \cdot e^{-2j\pi ft} dt ,$$

soit en remplaçant $\delta_T(t)$

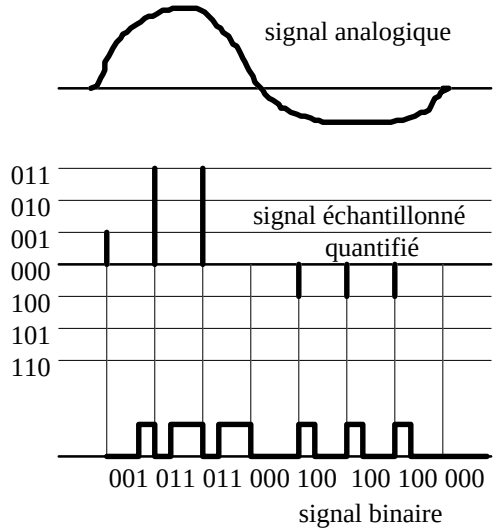
$$\tilde{X}(f) = \frac{1}{T} \sum_k \int_{-\infty}^{+\infty} x(t) \cdot e^{-2j\pi\left(f - \frac{k}{T}\right)t} dt .$$

Si $X(f)$ est la transformée de Fourrier de $x(t)$, alors $\tilde{X}(f)$ peut s'écrire

$$\tilde{X}(f) = \frac{1}{T} \sum_k X\left(f - \frac{k}{T}\right),$$

c'est à dire $\tilde{X}(f)$ est une somme infinie de la densité spectrale $X(f)$ décalée à toutes les fréquences multiples de $1/T$. Si le spectre de $x(t)$ est borné ($(\forall f > f_{\max}, X(f) = 0)$ et les "morceaux" $X(f)$ qui constituent le spectre $\tilde{X}(f)$ ne se recouvrent pas, c'est à dire $f_e = \frac{1}{T} \geq 2 f_{\max}$ (critère de Shannon ou *théorème d'échantillonnage*), alors $x(t)$ peut être facilement reconstitué à partir de $\tilde{x}(t)$ par le filtrage d'une bande de $\tilde{X}(f)$ et la multiplication par T.

Chaque échantillon du signal est ensuite quantifié selon une échelle à 2^n niveaux (correspondant à *n* bits) — appelés niveaux de quantification. Les erreurs de quantification déterminent un bruit de quantification. Si les échantillons, codés chacun sur *n* bits, sont ensuite transmis tels quels, le débit binaire exigé est $D = 2 f_{\max} \cdot n$ bits/s.



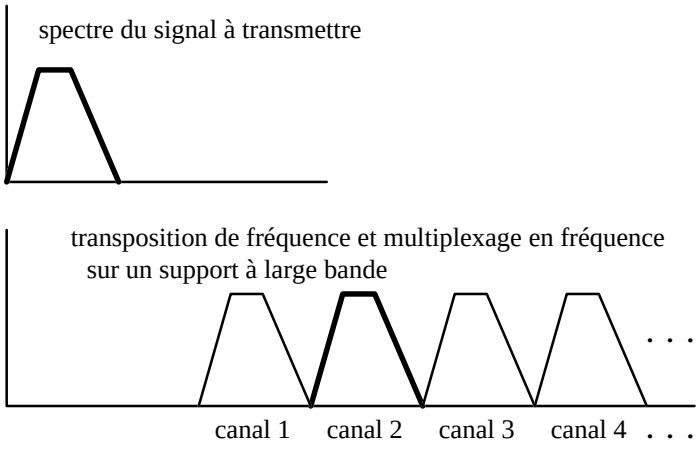
Exemple : pour le réseau téléphonique commuté, f_{max} = 4kHz, f_e= 8kHz (échantillonnage toutes les 1/125 secondes). 256 niveaux (8 bits) sont employés pour la quantification, le débit binaire nécessaire est donc de 64 kbits/s.

Afin d'obtenir une *même erreur relative* quel que soit le niveau du signal, l'échelle de quantification n'est pas linéaire. Les lois non linéaires employées sont la loi μ aux Etats-Unis et la loi *A* en Europe (loi normalisée par le CCITT). Les circuits qui implémentent ces lois sont appelés *companders*, le circuits qui réalisent l'ensemble des opérations, de l'échantillonnage/interpolation au codage/décodage sont appelés *codecs*.

Dans certains cas particuliers le débit binaire exigé peut être réduit en utilisant un codage plus économique. Ainsi, pour un signal de parole on constate que les variations de niveau entre deux échantillons successifs sont presque toujours de ± 1 ; au lieu de coder le niveau de chaque échantillon individuel, nous préférerons coder (en utilisant par exemple "1" pour +1 et "0" pour −1) la différence par rapport à l'échantillon précédent. On obtient ainsi ce qu'on appelle *modulation Delta*. Les distorsions introduites (surtout aux niveaux bas) sont relativement importantes et les variations rapides du signal sont éliminées. La modulation Delta *adaptative* consiste à utiliser une échelle non-linéaire et à coder plusieurs valeurs de la pente d'évolution du signal. Enfin, la modulation MIC différentielle adaptative avec prédiction à long terme ne transmet pas la valeur absolue de l'échantillon mesuré, mais la différence — codée sur 4 bits — entre la valeur de l'échantillon et celle prédite par un filtre adaptatif ; le débit binaire exigé est réduit à 24 ou 32 kbits/s.

7.2. Multiplexage en fréquence ou multiplexage temporel

Le multiplexage permet de transmettre *n* canaux de communication de débit *d* sur un support qui accepte le débit *D* = *nd*. Le multiplexage en fréquence consiste à appliquer une transposition de fréquence différente au signal sur chaque canal, afin d'occuper uniformément la bande disponible sur le support à large bande (voir la figure suivante) ; les signaux modulés sont alors émis simultanément. Des filtres passe-bande sont employés à l'arrivée afin de choisir un canal.



Le multiplexage temporel consiste à découper les messages sur chaque canal en tranches de longueur en général fixe et à transmettre ces tranches successivement dans des intervalles temporels au débit élevé permis par le support.

	IT 1		IT n	IT 1		IT n	
. . .	tranche i canal 1	. . .	tranche j canal n	tranche i+1 canal 1	. . .	tranche j+1 canal n	. . .

Plusieurs politiques différentes peuvent être employées afin d'associer des intervalles temporels (IT) aux différents canaux :

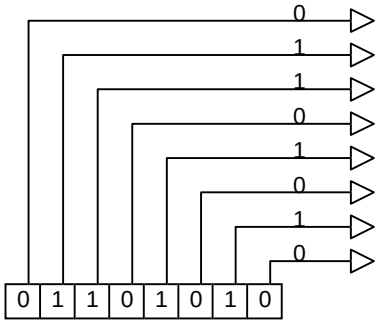
1° Les IT alloués à chaque canal sont bien définis. Cela permet de réduire la quantité d'informations de signalisation à faire circuler, mais réduit l'efficacité du multiplexage : quand plusieurs canaux sont inactifs, leurs IT restent inutilisés.

2° L'allocation des IT est dynamique : tous les IT sont alloués aux seuls canaux actifs et l'identification du canal est ajoutée à chaque tranche d'informations. Dans ce cas, le débit permis par le support peut être inférieur à la somme des débits des canaux individuels ($D < nd$).

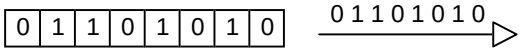
7.3. Parallèle ou série

Les octets qui composent un message sont presque exclusivement transmis les uns après les autres. En revanche, les bits qui composent un octet peuvent être transmis soit successivement — transmission série — soit simultanément — transmission parallèle.

Pour une transmission parallèle le support s'appelle **bus** et doit comporter 8 canaux élémentaires (1 bit). Le coût du support de communication est donc beaucoup plus élevé que pour une transmission série et des interférences apparaissent facilement, ce qui fait que les liaisons parallèles sont réservées aux transmissions sur des courtes distances et qui nécessitent des débits maximaux. C'est le cas par exemple pour les communications à l'intérieur d'un ordinateur, entre l'ordinateur et une unité de disque, une imprimante ou des équipements de mesure (IEEE488-HPIB).



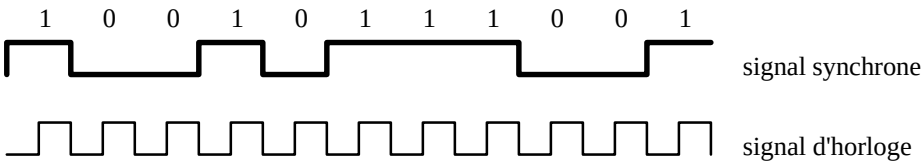
Les communications à distance plus élevée sont presque exclusivement de type série.



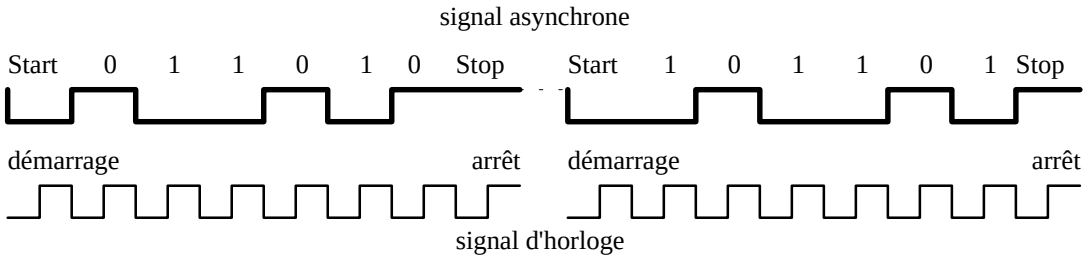
Les interfaces série les plus utilisées sont RS232C (débit maximum de 9600 bit/s pour une distance d'environ 20 m) et RS422 (distance d'environ 1km).

7.4. Synchrone ou asynchrone

Le mode de transmission est synchrone lorsque tous les instants significatifs dans la séquence de données sont séparés par des multiples d'un intervalle de temps T (la période d'un horloge). Le signal d'horloge est continu et il n'y a pas de séparation entre des caractères (ou octets) successifs. Comme le signal d'horloge à la réception est refait à partir du signal de données, les transitions de ce dernier doivent être suffisamment nombreuses pour éviter la perte de la synchronisation. Ce mode de transmission est le plus efficace et est donc employé sur toutes les liaisons à débit élevé.

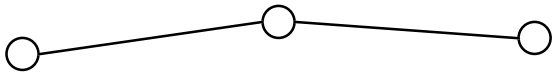


En mode de transmission asynchrone, les caractères successifs sont transmis indépendamment les uns des autres ; chaque caractère est encadré par un bit de Start et 1, 1,5 ou 2 bits de Stop, destinés à assurer le démarrage et l'arrêt correct de l'horloge du récepteur. L'horloge régit uniquement l'émission et la réception des bits d'un même caractère. Entre deux caractères successifs, la ligne peut être inactive pendant une durée quelconque. Ce mode de transmission est employé pour la communication avec des périphériques asynchrones (clavier par exemple).

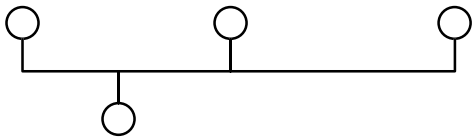


7.5. Point à point ou multipoint

Les liaisons point à point permettent la communication directe entre deux correspondants seulement. Dans un réseau composé de liaisons point à point, un message doit parcourir un certain nombre de liaisons et de nœuds intermédiaires pour arriver à destination.



Les liaisons multipoint permettent la connexion de plusieurs correspondants à une même ligne. Des communications de type *broadcast* (un émetteur et plusieurs récepteurs) peuvent être facilement assurées.



7.6. Simplex, semi-duplex ou duplex

Le transfert d'informations entre deux équipements peut s'effectuer de 3 façons distinctes :

1° En mode simplex, le transfert des informations peut s'effectuer en un seul sens. Une voie dite de retour, de débit beaucoup plus bas que la voie principale, peut exister afin de permettre l'envoi d'informations de signalisation. Deux voies unidirectionnelles sont en général utilisées : voie principale, voie de retour.

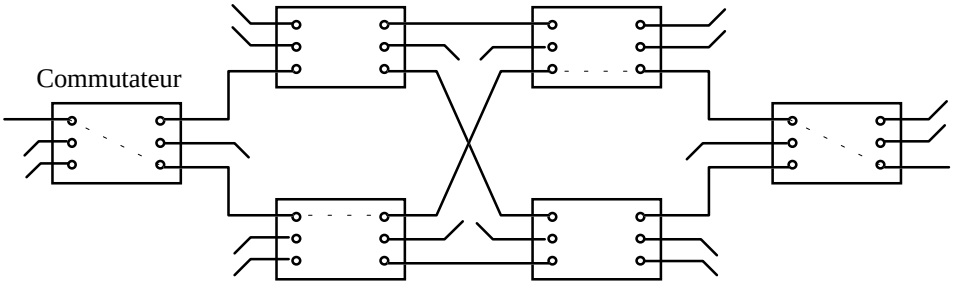
2° En mode semi-duplex (*half-duplex*), le transfert des informations peut s'effectuer dans les deux sens au même débit, mais pas simultanément. A chacune instant, le rôle d'émetteur revient à un équipement et celui de récepteur à l'autre équipement. Deux voies unidirectionnelles — alternativement voies principales et voies de retour — sont en général utilisées.

3° En mode duplex (*full-duplex*), le transfert d'informations peut s'effectuer simultanément dans les deux sens, au même débit. Chaque équipement est en même temps émetteur et récepteur. En général, quatre voies unidirectionnelles sont utilisées : deux voies principales, deux voies de retour. Certains équipements utilisent seulement deux voies unidirectionnelles, le mode duplex étant virtuel.

7.7. Commutation de circuits, de messages ou de paquets

Trois familles de techniques peuvent être employées afin de faire communiquer deux équipements connectés par un réseau complexe :

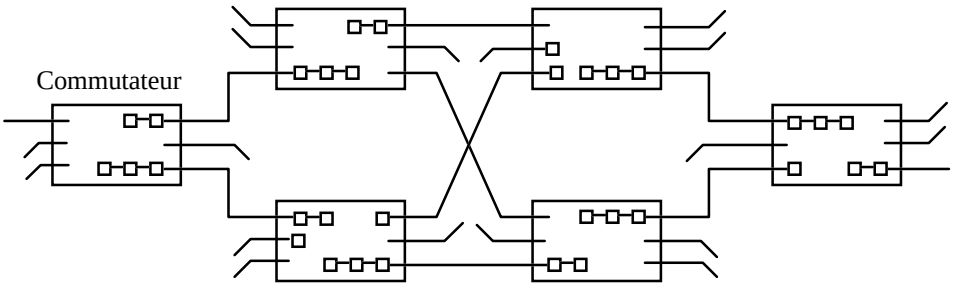
1° Afin de faire communiquer deux équipements distants, à travers un réseau complexe, la première technique utilisée a été celle de la commutation de circuits (mise en œuvre, par exemple, sur le réseau téléphonique analogique, RTC, et sur le réseau numérique à intégration de services, Numéris/RNIS). Aux débuts de la téléphonie, l'ouverture d'un circuit de communication entre deux équipements correspondait à la création d'une liaison physique temporaire. Aujourd'hui, il s'agit plutôt de l'établissement à travers un réseau complexe d'un canal **synchrone** entre les équipements terminaux. Les nœuds du réseau sont des commutateurs qui contribuent à l'établissement du canal synchrone.



Cette technique permet le transfert d'informations quelconques (même voix, images, pour lesquelles le temps de réponse est critique) mais est peu rentable en mode conversationnel, quand le rapport entre le volume d'informations à échanger et la durée de la communication est réduit. La tarification est en général proportionnelle à la durée de connexion, au débit de la ligne et à la distance entre les sites qui communiquent.

2° La commutation de messages est utile pour les cas où le temps de réponse est peu important et c'est le rendement du réseau qui est le paramètre principal. Chaque nœud du réseau a une capacité importante de stockage (en général stockage magnétique) et chaque message (de plusieurs koct, de longueur variable) est stocké dans un nœud avant d'être relayé vers le nœud suivant (qui l'approche de la destination).

3° La commutation par paquets, mise en œuvre en général sur les réseaux de transmission de données, est une évolution de la commutation de messages : les messages sont découpés en tranches de taille réduite (ex. 100 octets) et en général fixe et le stockage dans chaque nœud du réseau utilise des mémoires électroniques. Cela permet une réduction considérable du temps de réponse (qui ne peut toutefois pas être garanti à 100%) tout en assurant un bon rendement du réseau.



Les réseaux à haut débit utilisent principalement deux évolutions récentes des techniques de commutation de paquets : le relais de trames et l'ATM. Le relais de trame utilise des paquets de taille variable mais, en simplifiant le contrôle de flux et des erreurs (grâce à l'augmentation de la fiabilité des liaisons point à point, ces contrôles peuvent être effectués uniquement aux extrémités) permet une accélération de la communication. L'ATM (*Asynchronous Transfer Mode*) emploie des paquets de taille fixe et réduite (53 octets) et simplifie, comme le relais de trame, le contrôle de flux et des erreurs. La tarification est en général proportionnelle au volume de données transmis, au débit de la ligne et à la distance entre les sites qui communiquent.

7.8. Types de procédures de communication

Problèmes auxquels une transmission doit faire face :

1° il n'existe en général qu'une seule voie physique (par sens de communication) entre les équipements, pour transmettre aussi bien des données que des informations de contrôle ;
2° la communication n'est pas fiable — les données ou les informations de contrôle peuvent être erronées à la réception.

Une procédure de communication est constituée d'un ensemble de règles de reconnaissance des messages valides, de reprise sur erreur et de réinitialisation permettant de faire communiquer deux équipements. Une procédure doit se charger de :

1° transférer l'information utile entre une source et une destination qui doit être identifiée par une adresse ;
2° structurer les messages en incluant les informations à transférer et les séquences de commande et de contrôle ;
3° superviser la liaison, en enchaînant les commandes, les réponses, les accusés de réception, etc. ;
4° reprendre le transfert en cas d'erreur, soit à partir du dernier état considéré valide par les deux interlocuteurs, soit depuis le début ; des temporisations qui dépendent du temps de propagation et du débit binaire sont implémentées afin d'éviter le blocage des transferts ;
5° gérer les équipements qui interviennent.

Un processus de communication est en général décomposé en plusieurs niveaux et une procédure particulière gère chacun des niveaux.

En fonction de leur généralité, les procédures peuvent être minimales (communication très contrainte avec un type bien déterminé d'équipement), intermédiaires ou universelles (communication générale entre équipements hétérogènes).

Selon que la communication est effectuée sous le contrôle exclusif d'un et même équipement ou que les rôles des deux équipements qui communiquent peuvent être symétriques, la procédure est *hiérarchisée* ou *équilibrée*.

Bibliographie

CACM 3/39 (1996) *Communications of the ACM*, no. 3, vol. 39, mars 1996 (en bibliothèque).

Clavier, J., Niquil, M., Coffinet, G., Behr, F. (1972) *Théorie et technique de la transmission des données*, Masson et Cie, 1972 (en bibliothèque, 628/177).

Présentation détaillée de techniques de codage et de techniques de modulation. Quelques erreurs (partie codage surtout). Présentation des éléments essentiels de la théorie de l'information.

Delahousse, A. (1997) *Câblage haut débit : voix, données, images*, Hermès, Paris, 1997 (en bibliothèque, 601/7579-0), 160 p.

Présentation rapide de supports de transmission et de leurs caractéristiques, présentation plus détaillée de normes de câblage et discussion de la conception ainsi que de la mise en oeuvre du câblage. Présentation très sommaire de quatre études de cas.

Maiman, M. (1994) *Télécoms et réseaux*, Masson, Paris, 1994.

Tour d'horizon rapide. Très inégale. Un point fort : des exemples de traces (niveaux OSI 2 à 5).

Marsault, X. (1992) *Compression et cryptage en informatique*, Hermès, Paris, 1992 (en bibliothèque, 628/4679).

Présentation relativement complète, mais certains détails manquent (ex. pour DES), la présentation est parfois obscure (ex. systèmes à clés publiques) et parfois il y a des erreurs (ex. algo décompression LZW). L'annexe présente des programmes en C correspondant à différents algorithmes de compression et de cryptage (notamment DES).

Pujolle, G., Seret, D., Dromard, D., Horlait, E. (1989) *Réseaux et Télématique*, Tome 1, Eyrolles, Paris, 1989 (en bibliothèque, 628/5095-1).

Présentation relativement détaillée. Quelques erreurs. Assez inégale.

Stinson, D. (1996) *Cryptographie : théorie et pratique*, International Thomson Publishing, Paris, 1996 (en bibliothèque, 628/7568-0), 394 p.

Livre assez complet. Après la discussion de quelques techniques anciennes et la présentations de quelques notions théoriques, les principes qui sont à la base des deux techniques les plus utilisées actuellement (DES et RSA) sont présentés. Beaucoup de détails concernant l'authentification et la distribution de clef (Diffie-Hellman, Kerberos). Manque en revanche une "prise en main" des techniques ainsi que des détails concernant la mise en oeuvre des techniques dans les systèmes actuels (sécurisation des payements, authentification par tiers de confiance, etc.).