Cours de Réseaux — Partie 1

IUT de Villetaneuse — R&T 1ère année

Auteure : Laure Petrucci
Responsable du cours : Giulio Manzonetto

7 septembre 2012

Table des matières

Chapitre 1

Concepts de base

1.1 Généralités

Le terme *informatique* provient d'*information* et d'*automatique*, l'informatique étant le *traitement automatique de l'information*.

Un *réseau* est une organisation de voies de communication entre différentes entités. Cette définition est générale et peut s'appliquer par exemple aux réseaux routiers, ferroviaires, de télécommunications, ... Les entités qui communiquent au sein d'un *réseau informatique* sont des ressources informatiques dont on distingue deux types :

- les ressources *matérielles* : ordinateur, imprimante, scanner, ... qui sont des *composants de traitement*, les modems, cartes réseaux, commutateurs, routeurs, câbles, ... qui sont des *composants de transmission*.
- les ressources *logicielles* : applications informatiques, jeux, bases de données, ...

Un réseau informatique est constitué des moyens à la fois matériels et logiciels mis en œuvre pour assurer les communications entre des ressources informatiques.

Un réseau informatique permet aux entités reliées de partager des informations, les résultats de traitements, les ressources, par exemple pour que plusieurs utilisateurs travaillant sur des ordinateurs différents puissent utiliser la même imprimante.

1.2 Représentation de l'information

1.2.1 Quelle information représenter ?

Supposons qu'une machine doive envoyer l'image de la Figure 1.1 à une autre, après avoir convenu de la taille de cette image et de l'ordre d'envoi des éléments la constituant. La description se fera, par exemple, carré par carré, ligne par ligne, en commençant en haut à gauche, pour finir en bas à droite. Il est en effet impossible d'envoyer l'image telle quelle sans la coder. La séquence de couleurs à envoyer est donc (en notant blanc B et noir N) :

NNNNN NBBBN NBNBN NBBBN NNNNN

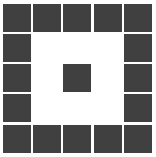


FIGURE 1.1 – Une image à transmettre

Une manière de coder la couleur de chaque carré consiste à associer une valeur à chaque couleur possible, par exemple 1 à B (le pixel sur l'écran est allumé) et 0 à N (le pixel est éteint). La suite de chiffres codant l'image est alors :

00000 01110 01010 01110 00000

1.2.2 Représentation des données

Les *données informatiques* sont représentées par des *suites de nombres*. Ces nombres sont écrits en *binaire* (c'est-à-dire en base 2). En *base* 2, on n'utilise que les chiffres 0 et 1.

L'utilisation de la base 2 garantit de pouvoir représenter un état stable d'un système physique, par exemple :

- circuit électrique ouvert/fermé
- carte perforée avec un trou/sans trou
- ...

Par conséquent, sur un système informatique, les données sont représentées par une suite de chiffres 0 et 1 correspondant à des états différents sur le support physique. Ces états peuvent être des tensions différentes (voir Chapitre 3).

1.2.3 Unités utilisées

Définition 1 (bit) *Un* symbole binaire *(donc en base* 2*) est appelé un* bit *(***bi***nary digi***t***).*

1 **bit** permet de coder 2 états : 0 et 1 ;

2 **bits** permettent de coder 4 états : 00, 01, 10 et 11 ;

3 **bits** permettent de coder 8 états : 000, 001, 010, 011, 100, 101, 110 et 111 ;

...

n **bits** permettent de coder 2^n états.

Définition 2 (octet) *Une* suite de 8 bits *est appelée un* octet.

Attention, en anglais le bit est appelé *bit*, alors que l'octet est appelé *byte* !

Les unités multiples des bits et des octets sont décrites dans les Tableaux 1.1 et 1.2 :

Unité	Symbole	Valeur (bits)
kilo-bit	Kb	$10^3 = 1\,000$
méga-bit	Mb	$10^6 = 1\,000\,000$
giga-bit	Gb	$10^9 = 1\,000\,000\,000$
téra-bit	Tb	$10^{12} = 1\,000\,000\,000\,000$

TABLE 1.1 – Unités multiples des bits

Unité	Symbole	Valeur (octets)
kibi-octet	Kio	$2^{10} = 1\,024$
mébi-octet	Mio	2^{20}
gibi-octet	Gio	2^{30}
tébi-octet	Tio	2^{40}

TABLE 1.2 – Unités multiples des octets

Traditionnellement, lorsque les préfixes “kilo”, “méga”, “giga” et “tera” sont appliqués aux octets, ils ne représentent pas une puissance de 10, mais une puissance de 2. Cet usage reste largement en vigueur chez les professionnels comme le grand public. Cependant cette tradition viole les normes en vigueur qui imposent d'utiliser les préfixes “kibi”, “mébi”, “gibi”, “tébi” pour les puissances de 2.

1.3 Mesures de performance

1.3.1 Débit

Définition 3 (débit) *Le* débit *d'un réseau mesure la* quantité d'information *que le réseau peut transmettre* par unité de temps *:*

$$\textit{débit} = \frac{\textit{quantité d'information}}{\textit{temps}}$$

L'unité est par conséquent le *bit par seconde*, noté b/s ou $b.s^{-1}$. Les réseaux actuels ayant un débit assez élevé, on utilise plus souvent des méga-bits par secondes, notés Mb/s ou $Mb.s^{-1}$.

Définition 4 (débits nominal et utile)
- *Le* débit nominal *d'un réseau est la quantité théorique maximale d'information pouvant être transmise par unité de temps.*
- *Le* débit utile *est la quantité d'information effectivement transmise par unité de temps.*

Définition 5 (taux d'utilisation) *Le* taux d'utilisation *du réseau est donc le rapport du débit utile au débit nominal :*

$$\textit{taux d'utilisation} = \frac{\textit{débit utile}}{\textit{débit nominal}}$$

Le taux d'utilisation est inférieur à 100%. Ceci est dû entre autres aux pertes sur la voie de communication et à l'intervalle de temps laissé entre l'envoi de deux messages.

1.3.2 Délais

Définition 6 (délai) *Le* délai total *d'acheminement d'un message se compose de deux parties :*
- *le* délai de transmission *est le temps mis pour transmettre la quantité d'information du message, c'est-à-dire :*

$$\textit{délai}_{\textit{transmission}} = \frac{\textit{quantité information}}{\textit{débit}}$$

- *le* délai de propagation *est le temps mis pour que le signal se propage sur le matériel. Les équipements traversés peuvent introduire des* retards*.*

$$\textit{délai}_{\textit{propagation}} = \frac{\textit{distance parcourue}}{\textit{vitesse}} + \textit{retards}$$

On a donc :

$$\text{délai}_{\text{total}} = \text{délai}_{\text{transmission}} + \text{délai}_{\text{propagation}}$$

1.4 Classification des réseaux

Les réseaux sont caractérisés non seulement par leur débit, mais également par le *rayon de couverture géographique* qu'ils permettent d'atteindre. Les différentes caractéristiques sont présentés dans le Tableau 1.3.

Le PAN est utilisé chez un particulier, le LAN dans un bâtiment (ou plusieurs bâtiments proches) d'une entreprise, le MAN interconnecte différents sites à l'échelle d'une agglomération, et le WAN s'étend sur un pays.

Sigle	Nom	Distance	Débit
PAN	Personal Area Network	quelques mètres	$1Mb/s$
LAN	Local Area Network	jusqu'à $2km$	de $10Mb/s$ à $1Gb/s$
MAN	Metropolitan Area Network	jusqu'à $100km$	environ $100Mb/s$
WAN	Wide Area Network	milliers de km	quelques Mb/s

TABLE 1.3 – Caractérisation des réseaux

1.5 Topologies de réseaux

1.5.1 Composants

Les composants des réseaux se répartissent selon deux types :
- les *composants de traitement* sont les entités produisant et/ou consommant les informations qui circulent sur le réseau (par exemple les ordinateurs) ;
- les *composants de routage* assurent la transition et la circulation des informations échangées entre les composants de traitement (par exemple, les câbles, commutateurs).

1.5.2 Architectures des réseaux

L'*architecture d'un réseau* comprend 3 parties :

L'architecture physique définit la topologie physique d'*interconnection* des composants du réseau.

L'architecture logique définit la topologie de *circulation de l'information*. Elle peut être différente de l'architecture physique.

L'architecture logicielle définit les *logiciels assurant l'acheminement* des données.

Le Tableau 1.4 montre les architectures physiques et logiques les plus classiques.

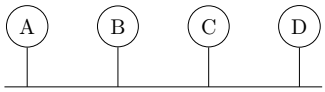
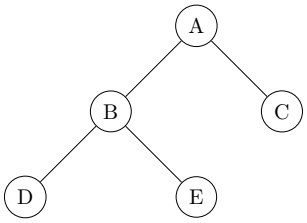
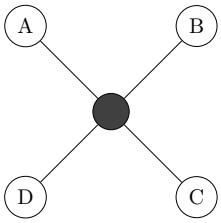
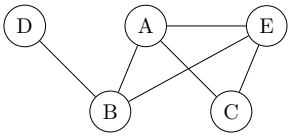
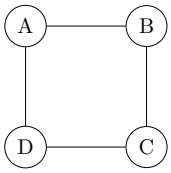
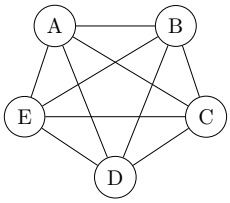
Bus		**Arbre**	
Étoile		**Graphe**	
Anneau		**Complète**	

TABLE 1.4 – Topologies classiques

Chapitre 2

Modèle de référence

2.1 Normalisation

2.1.1 Qu'est-ce et pourquoi ?

L'établissement de *normes* permet d'avoir une structure homogène pour faire communiquer différents équipements. La *conformité* à une norme garantit la satisfaction de règles précises.

Ainsi, des matériels différents, fabriqués par diverses entreprises, peuvent communiquer car la norme offre un cadre compatible entre ces entités hétérogènes.

La norme permet également d'assurer un niveau minimum de qualité.

2.1.2 Organismes de normalisation

La normalisation est effectuée par des organismes compétents au sein desquels les différents acteurs du domaine sont représentés. Trois organismes internationaux sont concernés par la normalisation dans le domaine des réseaux :

UIT (Union Internationale des Télécommunications) est l'institution spécialisée de l'ONU (Organisation des Nations Unies) dans le domaine des télécommunications. Elle comprend deux branches : l'UIT-T chargée de la normalisation dans le domaine des télécommunications et l'UIT-R qui s'occupe du domaine des radiocommunications.

IEC (International Electrotechnic Commission), fondée en 1906, est chargée de coordonner et d'unifier les normes dans le domaine de l'électricité.

ISO (International Standards Organisation) est une organisation privée chargée de la normalisation dans tous les domaines sauf l'électricité et l'électronique.

Ces organismes regroupent des représentants d'organismes nationaux. En France, les normes sont gérées par l'AFNOR (Association Française de NORmalisation).

2.2 Modèle de référence OSI

Le modèle de référence défini par l'ISO est l'OSI (*Open System Interconnection*). Il permet à des systèmes hétérogènes de s'interconnecter et d'échanger des informations. Il est par conséquent indépendant de la structure et de la technologie des matériels employés. Ce modèle offre un cadre permettant d'assurer une compatibilité maximum entre les entités communicantes tout en minimisant les contraintes permettant de les réaliser.

La complexité de conception, de réalisation et de maintenance des logiciels et de l'architecture des réseaux, est maîtrisée grâce à une organisation en *couches*, chaque couche étant bâtie sur la précédente.

2.2.1 Principes de la structuration en couches

Le modèle OSI est composé de *sept couches*.

Chaque couche peut *interagir uniquement avec les deux couches adjacentes*.

Une couche N est constituée d'un ensemble d'entités formant un sous-système de niveau N. Elle ne peut *dialoguer qu'avec une couche de même niveau N* sur une autre machine. Les communications se font donc entre *entités homologues*. La communication entre deux entités homologues de niveau N obéit à un ensemble de *règles et formats, syntaxiques et sémantiques*, prédéfinis pour les entités de niveau N. Ces règles et formats définissent le *protocole* de niveau N.

Une couche de niveau N fournit des *services* pour la couche de niveau $N + 1$. La couche de niveau $N + 1$ communique à la couche N les caractéristiques du service attendu. Les services fournis par une couche N sont identifiés par des *SAP (Service Access Point)* ou *ports*.

2.2.2 Couches du modèle OSI

Les *sept couches* sont organisées comme indiqué dans le Tableau 2.1.

Niveau	Couche
7	application
6	présentation
5	session
4	transport
3	réseau
2	liaison de données
1	physique

TABLE 2.1 – Les sept couches du modèle OSI

Chaque couche a un rôle spécifique :

7. La *couche application* offre aux utilisateurs des services normalisés pour la conception de leurs applications.

6. La *couche présentation* réalise la compression, le cryptage et vérifie la syntaxe des données échangées.

5. La *couche session* contrôle le dialogue entre les machines qui communiquent. Elle gère en particulier la synchronisation du dialogue et la reprise après interruption.

4. La *couche transport* assure le transport de bout en bout, c'est-à-dire entre les deux stations qui communiquent. Elle garantit que le message est acheminé entre les deux stations avec *la qualité de service* demandée. Le terme qualité de service désigne un ensemble de propriétés que le demandeur du service exige du prestataire, telles que la garantie d'un débit minimum, le respect d'une borne maximum de temps de livraison de messages, ...

3. La *couche réseau* assure l'acheminement des blocs d'information à travers le sous-réseau. Elle choisit le meilleur chemin entre les deux commutateurs d'entrée-sortie du sous-réseau. Les blocs d'information de niveau 3 sont appelés *paquets*.

2. La *couche liaison de données* est responsable de l'acheminement sans erreur des blocs d'information entre les deux machines qui se trouvent aux extrémités d'une liaison de données. Les blocs d'information de niveau 2 sont appelés *trames*.

1. La *couche physique* définit les moyens mécaniques (connecteurs), électriques et fonctionnels nécessaires à l'activation, au maintien et à la désactivation des connexions physiques destinées à la transmission des données binaires au niveau de la couche liaison de données. Elle fournit

donc tous les éléments matériels et logiciels nécessaires au transport correct des données binaires, comme :
- les interfaces de connexion des équipements informatiques au support de transmission, appelées *jonctions* ;
- les supports de transmission ;
- les cartes réseaux ;
- les modems ;
- les multiplexeurs, qui concentrent plusieurs communications sur une ligne de transmission unique.

2.3 Interactions entre couches

2.3.1 Protocoles et Services

Les notions de protocole et de service sont fondamentales.

Un protocole est un ensemble de *règles et formats, syntaxiques et sémantiques* prédéfinis pour les entités d'un même niveau N de deux machines différentes.

Un service est fourni par une couche de niveau N à la couche de niveau $N+1$ d'une même machine.

La Figure 2.1 décrit la communication entre les 7 niveaux de couches de deux entités communicantes A et B.

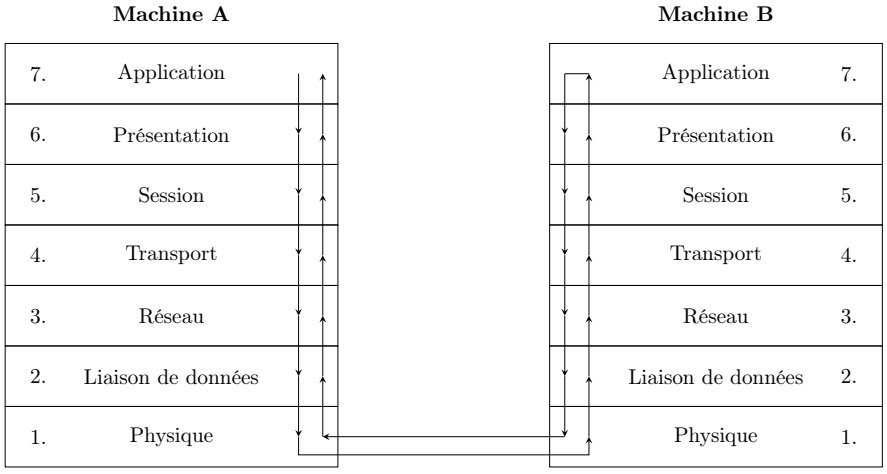


FIGURE 2.1 – Communication entre couches

2.3.2 Encapsulation, PDU et SDU

Les messages échangés par un protocole de niveau N sont appelés des PDU_N *(Protocol Data Unit de niveau N).*

Les messages échangés entre la couche N et la couche inférieure $N-1$ sont appelés des SDU_{N-1} *(Service Data Unit de niveau $N-1$).*

De plus, un protocole de niveau N ajoute au SDU_N qu'il a reçu des *informations de contrôle* visant à contrôler la bonne exécution du protocole. Ces informations de contrôle sont appelées

PCI_N (Protocol Control Information de niveau N).

On a par conséquent :

$$PDU_N = SDU_N + PCI_N$$

$$SDU_N = PDU_{N+1}$$

On dit alors que le PDU_N *encapsule* le SDU_N.

Au lieu d'indexer le *PDU* ou le *SDU* par le numéro de la couche, on le fait souvent précéder de la première lettre du nom de la couche (en anglais). Par exemple, $NPDU = PDU_3$, où le N indique la couche réseau (network).

2.3.3 Primitives de service

Il existe 4 primitives de service : *requête*, *indication*, *réponse* et *confirmation*.

Une *requête* est initialement envoyée par la couche N à la couche $N-1$ d'une même entité. Ensuite, une *indication* est transmise de la couche $N-1$ à la couche N de l'autre entité communicant. La *réponse* est envoyée par la couche N à la couche $N-1$ de cette seconde entité. Enfin, une *confirmation* est transmise de la couche $N-1$ à la couche N de l'entité ayant émis la requête. Ceci est illustré dans la Figure 2.2.

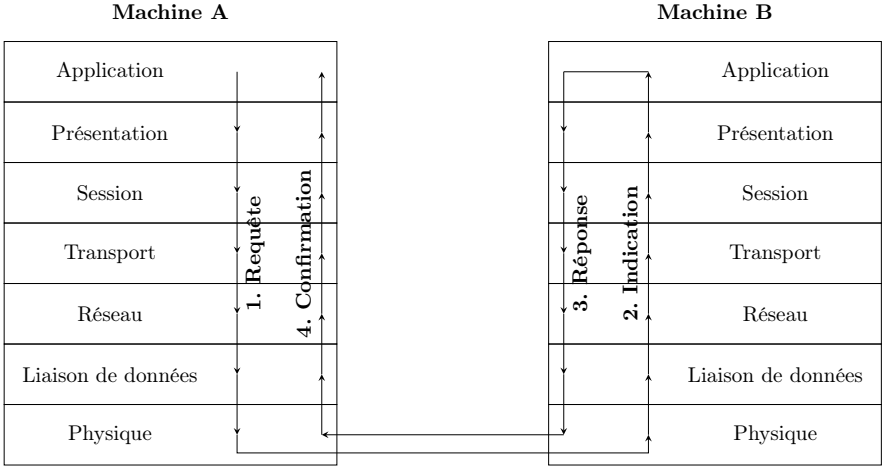


FIGURE 2.2 – Primitives de service

2.4 Types de connexion

La communication entre entités homologues de même niveau passe par l'établissement d'une *connexion*. Ce peut être une *connexion point à point* qui associe exactement deux entités, ou une *connexion multipoints* qui en associe plus.

Les *modes de communication* sont *simplex*, c'est-à-dire dans un seul sens, ou *duplex*, dans les deux sens.

Enfin, les protocoles peuvent opérer en *mode connecté* ou en *mode déconnecté*. En mode connecté, l'établissement de la connexion comporte trois phases : connexion, transfert, et déconnexion. Le contexte de la communication est préservé. Par contre, en mode déconnecté, seule la phase de transfert a lieu, et la communication s'effectue sans mémoire.

2.5 Représentation des états et messages

Lorsque l'on souhaite comprendre quelles informations transitent dans un réseau, et quand elles circulent, il convient de les représenter de manière facilement compréhensible, par exemple graphique.

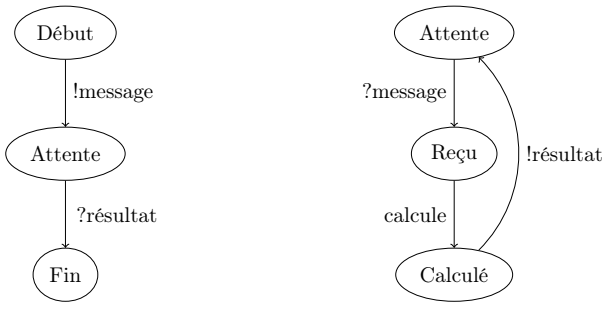
2.5.1 Automates

Un *automate* permet de représenter les différents états possibles d'un système, tel qu'une machine communicant avec une autre, ainsi que les actions lui permettant de passer d'un état à un autre.

Définition 7 (automate) *Un* automate *est un triplet* $\langle E, A, e_0 \rangle$ *où :*
- E *est un ensemble d'*états *du système. Un état* $e \in E$ *peut avoir un nom (encore appelé* étiquette*) ;*
- A *est un ensemble d'*actions *possibles (également appelées* transitions*)* $a \in A$ *telles que* $\exists e_1, e_2 \in E : a = (e_1, e_2)$*. L'action* a *peut alors être effectuée si le système se trouve dans l'état* e_1*. Une fois que l'action* a *est terminée, le système se trouve dans l'état* e_2*. Les actions dans* A *peuvent être étiquetées (c'est-à-dire avoir un nom) ou être des* envois *ou des* réceptions *de messages. L'envoi d'un message* `m` *est noté* `!m` *et la réception d'un message* `m` *est notée* `?m` *;*
- $e_0 \in E$ *est un état particulier du système dit* état initial*. Il indique par exemple dans quel état le système se trouve lorsqu'aucune des opérations qui nous intéressent n'a eu lieu.*

Un automate est représenté par un graphe dont les nœuds sont les états (cercles ou ellipses contenant le nom de l'état), et les arcs sont les actions.

Exemple : Soient une machine A et une machine B communicant entre elles. La machine A envoie un message à la machine B, pour que celle-ci effectue un calcul, puis attend le résultat. La machine B reçoit un message, effectue un calcul, renvoie le résultat, puis attend un nouveau message... Les comportements des machines A et B peuvent être représentés par des automates, comme indiqué dans la Figure 2.3.



(a) Machine A (b) Machine B

FIGURE 2.3 – Automates

2.5.2 Chronogrammes

Un *chronogramme* permet de représenter les échanges de messages au cours du temps. Chaque entité prenant part aux échanges est représentée par une ligne verticale. Le *temps* s'écoule du haut vers le bas. Un arc entre deux lignes verticales indique qu'un message a été envoyé par l'entité à la source de l'arc et est reçu par l'entité destination de l'arc. Lorsqu'un message se perd, l'arc n'atteint pas la destination et se termine par une croix (`X`).

Exemple : Supposons que 2 machines de type A (A_1 et A_2) et une machine de type B communiquent, telles que celles décrites dans la Figure 2.3. Supposons également que le message envoyé par A_2 se perde. Les échanges de messages peuvent être décrits par le chronogramme de la Figure 2.4.

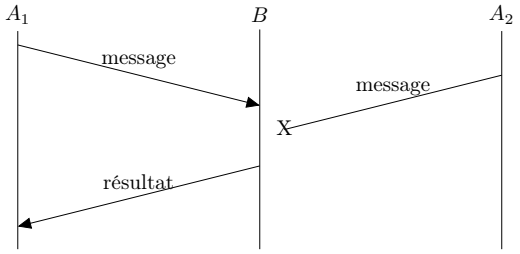


FIGURE 2.4 – Chronogramme

Chapitre 3

La couche physique

3.1 Transmission du signal

La *couche physique* assure le transfert de l'information. Celle-ci est transmise par la couche supérieure comme une suite de bits.

Dans cette couche sont définis : les signaux, les voies de transmission, les ETCDs (Équipement Terminal de Circuit de Données), les jonctions et les connecteurs de raccordement des voies de transmission aux ETCDs.

Les informations à transmettre étant des 0 et 1, on peut imaginer transmettre sur la voie physique un signal "carré" avec une tension différente pour les 0 et les 1 : c'est le principe du codage NRZ (voir Section 3.2.2). Or le support physique altère facilement les signaux carrés. Tant que le débit et la distance à parcourir sont faibles, on peut utiliser un signal carré, mais on est amené à mettre en œuvre des techniques plus sophistiquées pour améliorer les performances (voir Sections 3.2 et 3.3).

3.1.1 Obtention d'un signal carré

Théorème 1 (Fourier) *Tout* signal périodique *de fréquence f peut être décomposé en la somme de signaux sinusoïdaux dont la fréquence est un multiple de f, appelés* composantes spectrales *et éventuellement d'une fonction constante appelée* composante continue.

Une *composante spectrale* est déterminée par son *amplitude* A, son *rang* n et sa *phase* ϕ. Elle s'exprime sous la forme :

$$S(t) = \sin(2\pi n f t + \phi)$$

Les sinusoïdes de fréquences f, $2f$, ... sont appelées *harmoniques*. La fréquence f est dite *fréquence fondamentale*.

Plus il y a de composantes spectrales, plus la fonction obtenue s'éloigne d'une sinusoïde. C'est ainsi que l'on obtient une *signal carré*.

3.1.2 Bande passante

Le support physique ne peut pas faire circuler des signaux de n'importe quelle fréquence sans perte d'énergie et par conséquent d'amplitude.

Définition 8 (bande passante) *Chaque support admet une* fréquence de coupure basse*, qui indique la fréquence minimale pouvant circuler, et une* fréquence de coupure haute*, fréquence maximale pouvant circuler. La* bande passante *est la différence entre la fréquence de coupure haute et la fréquence de coupure basse.*

On tolère un affaiblissement de puissance de 50%*, c'est-à-dire que :*

$$\frac{P_{sortie}}{P_{entrée}} \geq \frac{1}{2}$$

La bande passante diminue avec la longueur du support. Elle dépend également des conditions physiques (matériau, ...).

Soit B la bande passante. Les fréquences des composantes spectrales étant des multiples entiers de la fréquence fondamentale f, le *nombre de composantes pouvant être transmises de façon fiable* est $N_s = \frac{B}{f}$.

Au cours de la transmission, le signal est affaibli. De plus ses composantes peuvent acquérir des déphasages distincts. Ceci peut conduire à des erreurs lors de la réception : ce qui est reçu est différent de ce qui a été envoyé. Pour éviter de tels problèmes, il est nécessaire d'utiliser des codes plus sophistiqués, qui satisfont certaines propriétés.

3.1.3 Théorème de Shannon

Pour transformer un signal analogique en signal numérique, on utilise une *technique d'échantillonnage*, qui consiste à mesurer l'amplitude du signal à intervalles de temps réguliers. On effectue ensuite une *quantification* du signal, c'est-à-dire que l'on définit des subdivisions régulières de l'amplitude du signal, et enfin une *numérisation* du signal, c'est-à-dire que l'on transforme le signal en nombres en fonction des hauteurs de quantification.

Théorème 2 (Shannon) *Il est nécessaire, pour reconstituer un signal propagé dans un milieu de bande passante B, que la fréquence d'échantillonnage soit au moins le double de B : $f_e \geq 2B$.*

3.1.4 Signaux et modulation

Transmission en bande de base

Ce type de transmission dit *en bande de base* est principalement réservé aux réseaux locaux. Le *codeur en bande de base* transforme les bits de données en un signal électrique numérique. Ce signal se présente comme une suite de *niveaux de tension* dont les amplitudes sont choisies parmi un nombre fini de possibilités.

Propriété 1 *Les signaux carrés imposent de véhiculer au moins* 6 *ou* 7 *composantes spectrales pour une bonne reconstitution du signal. Les fréquences sont alors élevées (d'où une forte consommation de bande passante) et la distance à parcourir doit être faible.*

On peut augmenter un peu les distances grâce à l'utilisation de *répéteurs*. Ceux-ci reforment le signal et le rendent à nouveau carré.

La *valence* est le nombre de niveaux possibles.

Exemple : La Figure 3.1 montre un signal numérique de valence 4, car il y a 4 niveaux possibles de tension : V_0, V_1, V_2 et V_3. Chaque niveau de tension a une durée identique Δ.

Théorème 3 (Nyquist) *Si un signal à V niveaux significatifs est transmis avec une bande passante B, le débit binaire maximal que l'on puisse obtenir est : $D_{max} = 2Blog_2V$.*

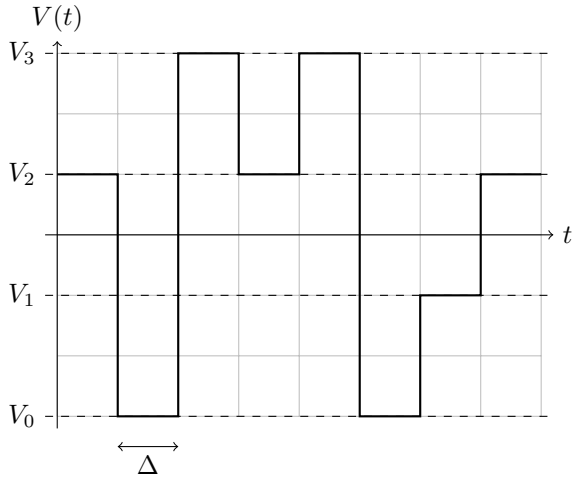


FIGURE 3.1 – Un signal numérique

Transmission à large bande

Lorsque la longueur de la liaison dépasse quelques centaines de mètres, les informations ne peuvent plus être transmises sur le support de liaison sans transformation du signal numérique en signal analogique. Dans une *transmission à large bande*, les données modulent une *onde porteuse* sinusoïdale dont la fréquence est bien plus élevée que le rythme de transmission des données. Le *codage* est alors appelé *modulation* et le *décodage* est nommé *démodulation*, d'où le nom de l'appareil *modem*.

Propriété 2 *La transmission à large bande permet de véhiculer une onde non carrée, ayant peu de composantes spectrales. La bande des fréquences utilisées est alors plus faible, les distorsions du signal moins importantes. La distance entre émetteur et récepteur peut donc être plus élevée.*

3.1.5 Rapidité de modulation

La *rapidité de modulation* est le nombre de changements d'états par seconde. L'unité de mesure de la rapidité de modulation R est le *baud* (du nom de Baudot, inventeur du télex) :

$$R = \frac{1}{\Delta} bauds$$

où Δ est l'intervalle de temps entre deux changements de niveau, appelé *intervalle significatif*.

Exemple : Supposons que, dans l'exemple de la Figure 3.1, V_0 code 00, V_1 code 01, V_2 code 10 et V_3 code 11. Le signal représenté est 10 00 11 10 11 00 01 10. Le débit est de 2 bits toutes les Δ secondes, donc $D = \frac{2}{\Delta} bits/s$. La rapidité de modulation est $R = \frac{1}{\Delta} bauds$ soit la moitié du débit binaire.

Si un état transporte n bits, il faut un signal de valence $V = 2^n$. On a alors un débit $D = nR$. Comme $n = \log_2 V$:

$$D = R \log_2 V$$

3.2 Codage en bande de base

3.2.1 Phases du codage

Le codage comporte 2 phases : le *codage logique* et le *codage physique*.

Le *codage logique* permet de pallier les défauts du codage physique. Il transforme la suite de bits à envoyer en une nouvelle suite au moins aussi longue. Les bits reçus sont groupés en paquets de n_e bits, puis codés selon une table de transformation en paquets de n_s bits, tels que $n_s \geq n_e$.

Soient D_e le débit binaire à l'entrée du codeur, et D_s celui à la sortie. On a :

$$D_s = \frac{n_s}{n_e} D_e$$

Le *codage physique* produit les signaux sur la ligne. Il doit être bien adapté au support de transmission.

Propriété 3 *Un* bon code *doit satisfaire au mieux les propriétés suivantes :*

1. Faible rapidité de modulation : *D'après le théorème de Nyquist (Théorème 3)* $D_{max} = 2B \log_2 V$*, et la relation indiquant le débit* $D = R \log_2 V$*, on obtient* $R \leq 2B$*. Cette relation est appelée* critère de Nyquist*. Elle indique que la largeur de bande du support limite la rapidité de modulation et donc le débit maximal sur la ligne.*
2. Limitation de la désynchronisation : *Pour que l'horloge du décodeur reste synchrone avec celle du codeur, l'intervalle de temps entre deux changements de niveau de signal doit être le plus court possible.*
3. Absence de composante continue : *Les codeurs et décodeurs physiques comportent des transformateurs et des amplificateurs qui suppriment la composante continue car elle peut endommager les équipements électroniques présents sur le support.*
4. Absence de polarisation du support : *Si l'on utilise des paires torsadées, il faut pouvoir croiser les fils sans avoir de conséquence sur l'information codée.*

Les Propriétés 3(1) à 3(4) ne peuvent pas être satisfaites simultanément. C'est pourquoi un bon code doit faire des compromis.

3.2.2 Codage NRZ (No Return to Zero)

La *valence* du *code NRZ* est 2. Un bit 1 est traduit par un niveau v et un bit 0 par le niveau $-v$. Le code NRZ vérifie la Propriété 3(1) avec $R = D$, mais aucune des autres propriétés. C'est un codeur physique et non un codeur logique.

Exemple : Le codage NRZ de la séquence de bits `1110 0001 1011` est représenté dans la Figure 3.2.

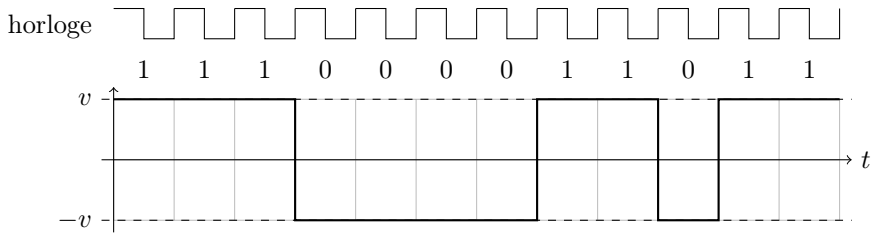


FIGURE 3.2 – Un codage NRZ

3.2.3 Codage NRZI (No Return to Zero Inverted)

La *valence* du *code NRZI* est 2. Lorsque l'on commence à transmettre un bit 1, on change de niveau. Par contre, lorsque l'on transmet un bit 0 le niveau précédent est conservé. Le code NRZI vérifie les Propriétés 3(1), 3(4), et 3(2) pour les suites de 1. Il est utilisé dans le codage 4b/5b (voir Section 3.2.7).

Exemple : Le codage NRZI de la séquence de bits `1110 0001 1011` est représenté dans la Figure 3.3, en supposant que le niveau précédent était $+v$.

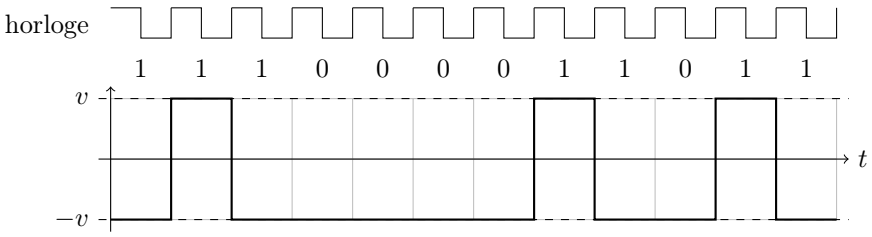


FIGURE 3.3 – Un codage NRZI

3.2.4 Codage Manchester

Les codes Manchester sont des *codes biphase*, c'est-à-dire qu'ils associent un codage physique avec un codage logique. Le codage logique transforme 1 bit en 2 bits. Il est dit *1b/2b*.

La *valence* du *code Manchester* est 2. Un bit 1 est transformé en deux bits 10, et un bit 0 est transformé en 01. Après le codage logique, le codage physique *NRZ* est employé. La combinaison de ces deux codages implique une transition au milieu du bit d'entrée, comme on peut le constater sur l'exemple de la Figure 3.4.

La Propriété 3(2) est satisfaite, car l'intervalle de temps le plus long entre deux changements de niveau est $T_b = \frac{1}{D}$. La Propriété 3(1) est quasiment satisfaite, car le plus court intervalle de temps entre deux changements de niveau est $\Delta = \frac{T_b}{2}$, donc $R = \frac{2}{T_b}$. Il n'y a pas de composante continue, d'où la Propriété 3(3). Par contre, la Propriété 3(4) n'est pas satisfaite car les bits sont polarisés.

Exemple : Le codage Manchester de la séquence de bits `1110 0001 1011` est représenté dans la Figure 3.4.

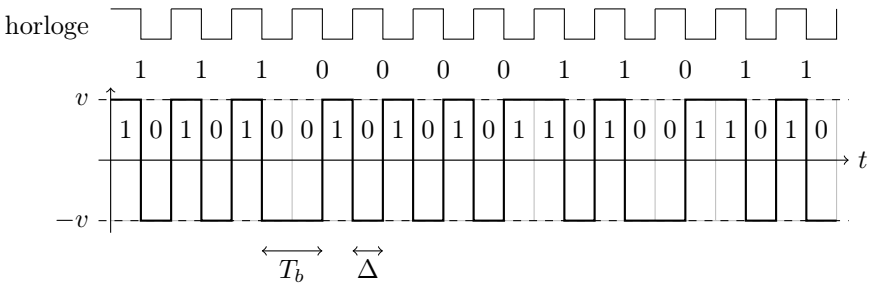


FIGURE 3.4 – Un codage Manchester

3.2.5 Codage Manchester différentiel

Le *codage Manchester différentiel* a, comme le codage Manchester, une valence 2 et emploie le codage NRZ comme codage physique. Le *codage logique* 1b/2b dépend du dernier bit β généré.

Lorsque l'on code un bit 1, on génère 2 bits $\beta\overline{\beta}$. Lorsque l'on code un bit 0, on génère 2 bits $\overline{\beta}\beta$. La combinaison de ce codage et du NRZ implique une transition au milieu du bit d'entrée, plus une transition en début de bit lorsque l'on transmet un 0, comme on peut le constater sur l'exemple de la Figure 3.5. Ce code satisfait les quatre propriétés requises pour un bon codage physique.

Exemple : Le codage Manchester différentiel de la séquence de bits `1110 0001 1011` est représenté dans la Figure 3.5, en supposant que le dernier bit généré était $\beta = 1$.

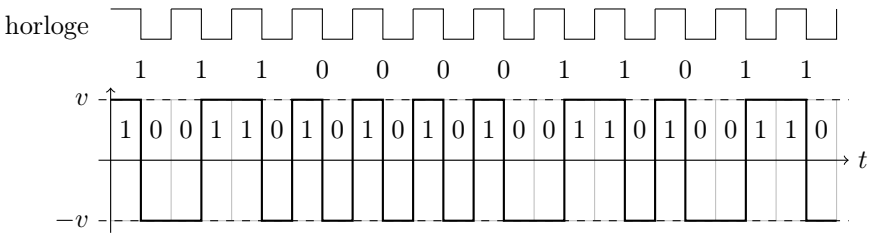


FIGURE 3.5 – Un codage Manchester différentiel

3.2.6 Codage bipôlaire

Le *code bipôlaire* est le code à plusieurs niveaux le plus répandu. Ces codes utilisent une valence > 2, souvent 3 (avec des niveaux v, 0 et $-v$) ou une puissance de 2. Le codage bipôlaire est principalement utilisé pour coder la parole. Lors de la transmission d'un bit 0 aucune impulsion n'est envoyée (le niveau reste 0). Lors de la transmission d'un bit 1, une impulsion est émise durant la moitié d'un bit, avec une polarisation inverse de celle précédemment utilisée, comme illustré sur la Figure 3.6.

La rapidité de modulation est $\frac{2}{T_b}$. En fait, les états importants sont 0 et *non*-0. Ce code ne vérifie pas la Propriété 3(2). Il peut conduire à de longues suites de 0. Il a par conséquent été amélioré en *code bipôlaire à haute densité*. Un code bipôlaire d'ordre n limite à nT_b le plus grand intervalle de temps entre deux impulsions.

Exemple : Le codage bipôlaire de la séquence de bits `1110 0001 1011` est représenté dans la Figure 3.6, en supposant que la dernière impulsion envoyée était à $+v$.

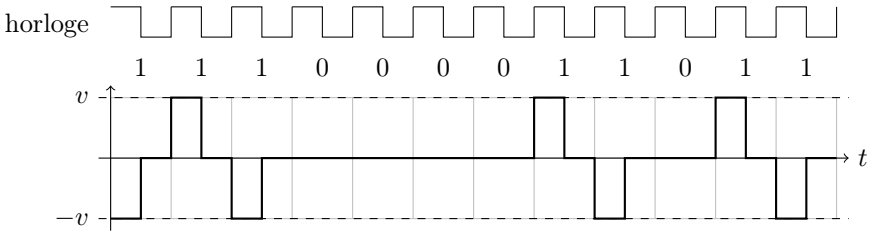


FIGURE 3.6 – Un codage bipôlaire

3.2.7 Codage 4b/5b

Le *code 4b/5b* est un *code par blocs*. Ceux-ci sont utilisés dans les réseaux à débit élevé.

Le *codage 4b/5b* est composé d'un codage logique transformant 4 bits en 5 bits, puis du codage physique NRZI. Le codage logique utilise une *table de codage* (voir Table 3.1) conçue de manière à éviter les longues suites de 0 (en fait il n'y a pas plus de 3 bits 0 consécutifs).

La rapidité de modulation est $\frac{5}{4T_b}$. La Propriété 3(1) est par conséquent satisfaite. Pour vérifier la Propriété 3(2), on considère le plus grand intervalle de temps entre deux niveaux : $4\frac{4}{5}T_b$. La

plus longue suite de 0 en comportant 3, elle est encadrée par deux 1. On a donc la forme 10001, de longueur $\Delta_{max} = 4\Delta$.

Entrée	Sortie	Entrée	Sortie	Entrée	Sortie	Entrée	Sortie
0000	11110	0100	01010	1000	10010	1100	11010
0001	01001	0101	01011	1001	10011	1101	11011
0010	10100	0110	01110	1010	10110	1110	11100
0011	10101	0111	01111	1011	10111	1111	11101

TABLE 3.1 – Table de codage 4b/5b

Exemple : Le codage 4b/5b de la séquence de bits `1110 0001 1011` est représenté dans la Figure 3.7, en supposant que le niveau précédent était $+v$.

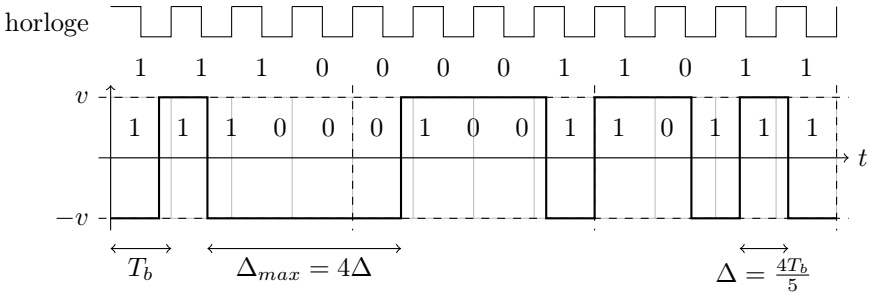


FIGURE 3.7 – Un codage 4b/5b

3.3 Modulation/démodulation (codage large bande)

Le codage en bande de base ne peut pas être utilisé pour la transmission de données à des vitesses très élevées ou sur de très grandes distances. Le signal est déformé car les hautes fréquences sont atténuées et les basses fréquences altérées par les répéteurs. Le bande passante étant réduite, il faut transformer le signal numérique en signal analogique par *modulation d'une onde porteuse*. Les opérations de *modulation en émission* et de *démodulation en réception* sont effectuées par un *modem* (modulateur-démodulateur).

Comme en bande de base, un codage logique peut avoir lieu avant la modulation.

L'*onde porteuse* est une sinusoïde de fréquence F_0 et d'amplitude A_0. Il y a 3 principaux types de modulation liés aux 3 *paramètres de la sinusoïde* :

- modulation d'*amplitude* ;
- modulation de *fréquence* ;
- modulation de *phase*.

3.3.1 Modulation d'amplitude

La *modulation d'amplitude* (en abrégé AM — Amplitude Modulation, ou ASK — Amplitude-Shift Keying) est la plus simple. Elle fixe (au moins) 2 *niveaux logiques* à l'amplitude de la porteuse : A_0 et A_1. Elle n'est en général pas utilisée seule, mais en conjonction avec une des deux autres méthodes de modulation. La Figure 3.8 montre un exemple de modulation d'amplitude.

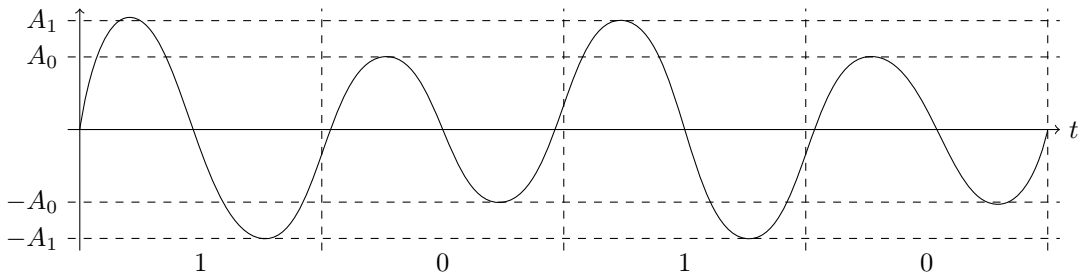


FIGURE 3.8 – Exemple de modulation d'amplitude

3.3.2 Modulation de fréquence

En *modulation de fréquence* (FM — Frequency Modulation, ou FSK — Frequency-Shift Keying), la porteuse de fréquence F_0 est modulée par deux valeurs opposées de fréquence f_1 et $-f_1$, permettant ainsi la représentation de deux niveaux logiques.

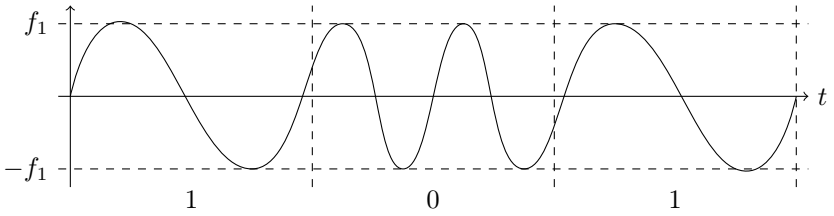


FIGURE 3.9 – Exemple de modulation de fréquence

3.3.3 Modulation de phase

En *modulation de phase* (PSK — Phase Key Shifting), la porteuse est de la forme $A_0 \sin(2\pi ft + \phi)$. Pour moduler la phase, on choisit des valeurs distinctes de ϕ. En utilisant des codes binaires d'au moins 2 bits, on peut augmenter la rapidité de transmission sans augmenter la rapidité de modulation. Par exemple, pour une modulation sur 2 bits : 00 pour une phase 0, 01 pour une phase $\frac{\pi}{2}$, 11 pour une phase $-\frac{\pi}{2}$ et 10 pour une phase π. La modulation PSK permet ainsi d'obtenir des vitesses de transmission plus élevées que la modulation FSK sur le même support, pour une bande passante similaire.

Chapitre 4

Détection et Correction d'erreurs

4.1 Généralités

Le support matériel utilisé par la couche physique n'est pas fiable à 100%. Il est par conséquent nécessaire de pouvoir *détecter des erreurs* parmi la suite de bits reçue, et éventuellement les *corriger*. Pour cela, la *couche liaison de données* de l'émetteur ajoute des bits au message à transmettre, qui permettent à la couche liaison de données de l'entité réceptrice du message de vérifier la cohérence de ce qu'elle a reçu.

La couche liaison de données construit ainsi des LPDU, encore appelées *trames*, qui comportent en particulier un *FCS* (Frame Check Sequence).

La problématique des erreurs comporte 3 aspects :
- la *détection* d'une erreur ;
- la *localisation* de l'erreur détectée ;
- la *correction* de l'erreur trouvée.

Pour répondre à ces problèmes, on utilise des *codes* qui sont appliqués au message à transmettre. Ils permettent de détecter *certaines* erreurs, mais *pas nécessairement toutes*, et *peu permettent la correction*. Ces techniques ne sont donc pas complètement fiables, d'autant que le FCS, utilisé pour vérifier et corriger le message, peut lui aussi être erroné.

4.1.1 Un code simple : la répétition

Un *approche naïve* consiste à *dupliquer* (c'est-à-dire répéter) le message à transmettre.

Supposons que le message effectivement transmis soit le double du message réel. Par exemple, pour envoyer 11100010, on transmet 1110001011100010. La détection et la localisation des erreurs sont alors simples : on cherche les différences entre la première et la seconde moitiés du message. Par contre, il est impossible de corriger une erreur détectée : le bit erroné est différent dans les deux copies, et rien ne permet de dire lequel est le bon.

Pour remédier à ce problème, on peut envoyer les message en 3 exemplaires au lieu de 2. Dans ce cas, un bit a soit la même valeur dans toutes les copies, ou la même valeur dans deux d'entre elles et l'autre valeur dans la troisième copie. Le bit correct est celui qui apparaît en deux exemplaires.

4.1.2 Inconvénients et problèmes rencontrés

Le code de répétition est simple, mais présente de nombreux *inconvénients*, illustrant ceux que l'on peut rencontrer avec un quelconque code :
- La détection et la correction d'erreurs nécessitent l'introduction de redondance dans les messages transmis. La *taille du message* à transmettre peut alors *augmenter* de manière significative.
- Certaines *erreurs peuvent ne pas être détectées*. C'est le cas lorsque l'on utilise la redondance, si le même bit est erroné dans toutes les copies du message.

– Pour une raison similaire, certaines *erreurs détectées ne peuvent pas être corrigées, voire être mal corrigées.*
– La correction nécessite plus de redondance que la détection d'erreurs.

4.2 Détection d'erreurs

4.2.1 Codes à contrôle de parité

Les codes à contrôle de parité sont de parité soit *paire*, soit *impaire*. Dans le premier cas, on va protéger une séquence de bits en ajoutant un nouveau bit de telle sorte que le nombre de bits ayant la valeur 1 (dans la séquence protégée plus le bit introduit) soit pair. Dans le second cas, ce nombre doit être impair.

VRC (Vertical Redundancy Check)

C'est la technique la plus simple. Un code ASCII étant défini sur 7 bits, on utilise le $8^{\text{ème}}$ bit de l'octet pour introduire le code vérificateur.

Exemple : Pour transmettre la chaîne de caractères `IUT`, on code chaque lettre en ASCII, puis on ajoute le code de parité.

Lettre	ASCII	VRC pair	VRC impair
I	1001001	**1**1001001	**0**1001001
U	1010101	**0**1010101	**1**1010101
T	1010100	**1**1010100	**0**1010100

Pour envoyer le message avec un code de parité pair, on transmet (avec l'ordre d'envoi des bits de gauche à droite) :

$$\underbrace{\mathbf{1}1001001}_{I}\ \underbrace{\mathbf{0}1010101}_{U}\ \underbrace{\mathbf{1}1010100}_{T}$$

Ce code permet de détecter les erreurs en nombre impair sans pouvoir corriger. Il est peu efficace.

LRC (Longitudinal Redundancy Check)

Le principe est similaire à celui du VRC, mais au lieu de protéger les caractères un par un, on protège l'ensemble des bits de même rang de tous les caractères. On obtient alors un code de protection sur 7 bits.

Exemple : Pour protéger IUT, on calcule le code :

I	1001001
U	1010101
T	1010100
LRC pair	1001000
LRC impair	0110111

Pour envoyer le message avec un code de parité pair, on transmet :

$$\underbrace{1001001}_{I}\ \underbrace{1010101}_{U}\ \underbrace{1010100}_{T}\ \underbrace{\mathbf{1001000}}_{LRC}$$

L'efficacité du code LRC dépend fortement du message transmis.

LRC+VRC

On peut également combiner les deux techniques précédentes. On protège alors chaque caractère par un code VRC et l'ensemble des bits par un code LRC. On obtient donc un LRC sur 8 bits. La parité des LRC et VRC utilisés est la même (tous les deux pairs ou tous les deux impairs).

Exemple : Pour transmettre la chaîne de caractères `IUT`, on code chaque lettre en VRC puis en LRC :

		VRC pair	VRC impair
I	1001001	**1**1001001	**0**1001001
U	1010101	**0**1010101	**1**1010101
T	1010100	**1**1010100	**0**1010100
LRC		01001000	00110111

Pour envoyer le message avec un code de parité pair, on transmet :

$$\underbrace{\mathbf{1}1001001}_{I}\ \underbrace{\mathbf{0}1010101}_{U}\ \underbrace{\mathbf{1}1010100}_{T}\ \underbrace{\mathbf{01001000}}_{LRC}$$

4.2.2 Codes polynômiaux

Les *codes polynômiaux*, encore appelés CRC (Cycling Redundancy Code), sont utilisés par la plupart des protocoles actuels.
Un *code polynômial* est basé sur l'utilisation d'un *polynôme générateur* $G(x)$ connu à l'avance par à la fois l'émetteur et le récepteur du message. Les polynômes manipulés sont binaires : tous les coefficients sont 0 ou 1. Par conséquent, un polynôme générateur de *degré* k s'écrit sous la forme :

$$G(x) = a_0 \oplus a_1.x \oplus a_2.x^2 \oplus a_3.x^3 \oplus \ldots \oplus a_k.x^k$$

où $$\forall i \in \{0, \ldots, k\}, a_i \in \{0, 1\}$$

Le polynôme $G(x)$ est associé à une valeur binaire.

Exemple : La valeur binaire associée au polynôme $G(x) = x^3 \oplus x \oplus 1$ est 1011.

Soit M le message (séquence de bits) à protéger. Un polynôme $M(x)$ lui est associé :

$$M = m_n \ldots m_2 m_1 m_0 \Rightarrow M(x) = m_n.x^n \oplus \cdots \oplus m_2.x^2 \oplus m_1.x \oplus m_0$$

Exemple : Au message $M = 1101$ est associé le polynôme $M(x) = x^3 \oplus x^2 \oplus 1$.

Le calcul du CRC s'effectue dans le corps $\mathbb{Z}/2\mathbb{Z}$, c'est-à-dire que :
- $1 \oplus 1 = 0$
- $x \oplus x = 0$
- $x = -x$

Procédure de codage

Soient :
- $G(x)$ un polynôme générateur de degré k ;
- $M(x)$ le polynôme associé au message M à transmettre.

La procédure de codage consiste à :
- calculer $P(x) = M(x).x^k$. Ceci correspond à un décalage de k bits (vers la gauche) du message M. La longueur du CRC calculé sera aussi de k bits. Cette opération de décalage revient à préparer la place nécessaire pour ces k bits de CRC.

– diviser le polynôme $P(x)$ par $G(x)$. Soient $Q(x)$ et $R(x)$ les polynômes quotient et reste ainsi obtenus :

$$P(x) = Q(x).G(x) \oplus R(x)$$

– le CRC est le reste $R(x)$ ainsi calculé. On remarque que le reste est forcément au maximum de degré $k-1$.
– le message effectivement transmis est associé au polynôme $M'(x) = P(x) \oplus R(x)$. Il est par conséquent composé du message initial M suivi de la séquence de k bits correspondant à $R(x)$.

Exemple : Soient le polynôme générateur $G(x) = x^3 \oplus x \oplus 1$ et le message à envoyer $M = 1101$. Le polynôme correspondant au message est $M(x) = x^3 \oplus x^2 \oplus 1$. Le degré de $G(x)$ est 3. Donc, $P(x) = M(x).x^3 = x^6 \oplus x^5 \oplus x^3$. Effectuons la division de $P(x)$ par $G(x)$:

$$\begin{array}{llllllll|l}
 & x^6 & \oplus x^5 & & \oplus x^3 & & & & x^3 \oplus x \oplus 1 \\
\oplus & x^6 & & \oplus x^4 & \oplus x^3 & & & & \hline
 & & x^5 & \oplus x^4 & & & & & x^3 \oplus x^2 \oplus x \oplus 1 \\
 & \oplus & x^5 & & \oplus x^3 & \oplus x^2 & & & \\
\hline
 & & & x^4 & \oplus x^3 & \oplus x^2 & & & \\
 & & \oplus & x^4 & & \oplus x^2 & \oplus x & & \\
\hline
 & & & & x^3 & & \oplus x & & \\
 & & \oplus & & x^3 & & \oplus x & \oplus 1 & \\
\hline
 & & & & & & & 1 &
\end{array}$$

Le quotient est donc $Q(x) = x^3 \oplus x^2 \oplus x \oplus 1$, et le reste $R(x) = 1$. Le message transmis a alors pour polynôme $M'(x) = x^6 \oplus x^5 \oplus x^3 \oplus 1$, d'où $M' = 1101001$.

Caractéristiques, décodage et erreurs

Propriété 4

1. *Un message M' transmis correctement a un polynôme $M'(x)$ divisible par le polynôme générateur $G(x)$.*
2. *Si $G(x)$ comporte au moins 2 termes, les erreurs simples sont détectables.*
3. *Si $G(x)$ a un facteur irréductible de 3 termes, les erreurs doubles sont détectables.*
4. *Si $G(x)$ est un multiple de $x \oplus 1$, les erreurs en nombre impair sont détectables.*

La qualité du codage dépend donc du choix du polynôme générateur $G(x)$.

Lorsque l'on reçoit un message M', deux cas peuvent se présenter :
– $M'(x)$ est divisible par $G(x)$: le CRC ne permet pas de détecter une erreur. Il y a de fortes chances que le message reçu soit correct.
– $M'(x)$ n'est pas divisible par $G(x)$: une erreur est détectée.

Le message initial M est obtenu en ignorant les k derniers bits de M' (c'est-à-dire en effectuant un décalage à droite de k bits sur M').

Le code polynômial ne permet pas de corriger les erreurs.

4.3 Correction d'erreurs

Deux approches permettent de corriger les erreurs :
– les *codes auto-correcteurs*, tels que le code de Hamming (voir Section 4.3.1) ;
– la *correction par retransmission* (voir Section 4.3.2), qui demande à l'émetteur de retransmettre le message lorsqu'une erreur est détectée.

4.3.1 Code de Hamming

Le *code de Hamming* est calculé à partir d'une mesure de dissimilarité entre deux séquences de bits de même longueur, appelée *distance de Hamming.*

Définition 9 (distance de Hamming) *La* distance de Hamming *entre deux séquences binaires m_1 et m_2 de même taille est le nombre de bits de même rang par lesquels ces deux séquences diffèrent. Elle est notée $d(m_1, m_2)$.*

Exemple : $d(1\mathbf{1}00110, 1\mathbf{0}1010) = 2$.

Définition 10 (distance d'un code) *Soit un code C comportant n séquences valides. La* distance de Hamming du code C *est la distance minimale séparant deux mots valides du code. Elle est notée $d(C)$.*

Propriété 5 *Un code de distance $d(C)$ détecte $d(C) - 1$ erreurs et corrige $k = \lfloor \frac{d(C)-1}{2} \rfloor$ erreurs.*

Exemple : Supposons que l'on souhaite transmettre des messages pouvant se coder comme l'une des séquences du code :
$C = \{000000, 001110, 011011, 100011, 101101\}$.
$d(C) = \min_{(c_1,c_2)\in C^2} d(c_1, c_2) = 3$.
Avec ce code, on peut détecter au maximum $d(C)-1 = 2$ erreurs et en corriger $k = \frac{d(C)-1}{2} = 1$.

4.3.2 Demande de retransmission

Lorsque l'on effectue de la *correction par demande de retransmission*, l'émetteur conserve une copie des données envoyées. Le récepteur applique une méthode de détection des erreurs. Quand il reçoit un message, le récepteur renvoie un paquet à l'émetteur, contenant un *acquittement positif* si aucune erreur n'a été détectée, et un *acquittement négatif* si une erreur a été trouvée. Lors de la réception d'un acquittement négatif, l'émetteur retransmet le message erroné.

Un message pouvant être perdu lors de la transmission, et donc ne jamais être acquitté, que ce soit positivement ou négativement, l'émetteur utilise un *temporisateur* qui permet de fixer un temps d'attente limite pour la réception d'un acquittement. Quand le temporisateur arrive à expiration, si aucun acquittement n'a été reçu, il est très probable que soit le message lui-même a été perdu, soit son acquittement a été perdu. Par conséquent, l'émetteur renvoie son message.

Pour assurer le bon fonctionnement d'un tel mécanisme, il est nécessaire d'identifier les messages transmis, par exemple en les numérotant.