

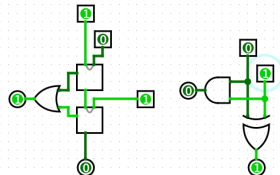
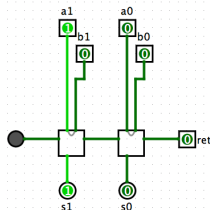
Principes de fonctionnement des machines binaires

2022–2023

Matthieu Picantin



- ◆ numération et arithmétique
- ◆ numération et arithmétique en machine
- ◆ codes, codages, compression,
- ◆ **contrôle d'erreur (détection, correction)**
- ◆ crypto (confidentialité, authenticité, intégrité)
- ◆ logique et calcul propositionnel
- ◆ circuits numériques



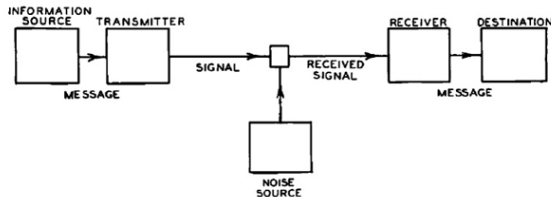


Fig. 1—Schematic diagram of a general communication system.

Détection (+retransmission)

- ♦ **juste assez de redondance** pour que le récepteur puisse **détecter** d'éventuelles erreurs & **demande une retransmission**
- ♦ adapté sur canal fiable

Correction

- ♦ **suffisamment de redondance** pour que le récepteur puisse **corriger** d'éventuelles erreurs
- ♦ adapté sur canal bruité

Redondance et mots de code

- ♦ mot de code (n bits) = données (m bits) + contrôle (r bits)
- ♦ 2^m mots de code (légaux) parmi 2^n mots possibles

Distance de Hamming

- ♦ nombre de bits différents entre 2 mots
- ♦ somme des 1 du XOR des 2 mots
- ♦ nombre minimum d'erreurs simples pour passer d'un mot à l'autre

Distance de Hamming d'un code

distance minimale entre deux mots du code:

$$d_H(\mathcal{C}) = \min \{d_H(u, v) : u \neq v \in \mathcal{C}\}$$

Qualité d'un code

- ♦ un code \mathcal{C} vérifiant $d_H(\mathcal{C}) \geq k + 1$ permet de détecter k erreurs
- ♦ un code \mathcal{C} vérifiant $d_H(\mathcal{C}) \geq 2k + 1$ permet de corriger k erreurs

Un code 1-détecteur

L'ajout d'un **bit de parité** produit un code de distance de Hamming 2

Un code 2-correcteur

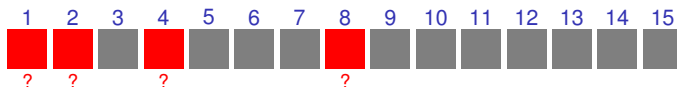
Le code $\{000000000, 000001111, 111110000, 111111111\}$ est de distance de Hamming 5

En théorie: de l'espace pour les boules (condition nécessaire)

Pouvoir corriger toute erreur simple pour m bits de données
demande r bits de contrôle avec $m + r < 2^r$

En pratique: le code de Hamming (condition suffisante)

- ♦ bits numérotés de 1 à $n = m + r$ de gauche à droite
- ♦ r bits de contrôle aux positions puissances de 2 (1, 2, 4, 8, 16, ...)
- ♦ m bits de données aux autres positions (3, 5–7, 9–15, 17, 18...)
- ♦ bits de contrôle = calcul de parité sur les bits de données
aux positions dont la décomposition en somme de puissances de 2
fait intervenir la position du bit de contrôle concerné
- ♦ détection du bit erroné (et correction) par somme
des positions des bits de contrôle non-conformes à la parité



Mieux encore: le code de Hamming étendu

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

000	001	002	003	004	005	006	007	008	009	010	011	012	013	014	015
016	017	018	019	020	021	022	023	024	025	026	027	028	029	030	031
032	033	034	035	036	037	038	039	040	041	042	043	044	045	046	047
048	049	050	051	052	053	054	055	056	057	058	059	060	061	062	063
064	065	066	067	068	069	070	071	072	073	074	075	076	077	078	079
080	081	082	083	084	085	086	087	088	089	090	091	092	093	094	095
096	097	098	099	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Mieux encore: le code de Hamming étendu

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

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

Mieux encore: le code de Hamming étendu

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

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

Mieux encore: le code de Hamming étendu

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

- ◆ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ◆ un bit de parité supplémentaire (parité globale) numéroté 0

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

Mieux encore: le code de Hamming étendu

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

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

Mieux encore: le code de Hamming étendu

- ◆ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ◆ un bit de parité supplémentaire (parité globale) numéroté 0

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

Mieux encore: le code de Hamming étendu

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

- ◆ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ◆ un bit de parité supplémentaire (parité globale) numéroté 0

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

Mieux encore: le code de Hamming étendu

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

- ◆ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ◆ un bit de parité supplémentaire (parité globale) numéroté 0

[illegible]

Mieux encore: le code de Hamming étendu

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

[illegible]

Mieux encore: le code de Hamming étendu

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

- ♦ des bits numérotés de 1 à $n = m + r$ comme pour le code original
- ♦ un bit de parité supplémentaire (parité globale) numéroté 0

Codes CRC (Cyclic Redundancy Check)

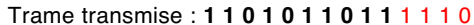
- ♦ correspondance entre rang des bits et degré des monômes
 - ▶ le mot **10011** code le polynôme $x^4 + x + 1$
- ♦ arithmétique polynomiale (soustraction modulo 2 et division euclidienne)

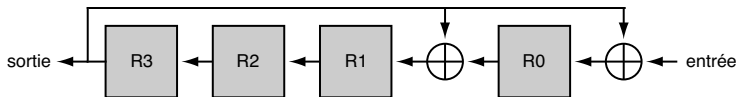
Utilisation d'un polynôme générateur $G(x)$

- ♦ $G(x)$ de degré r et message $M(x)$
- ♦ ajout de r bits à 0 après le bit de poids faible de $M(x)$
- ♦ division de $x^r M(x)$ par $G(x)$: reste $R(x)$
- ♦ envoi de $T(x) = x^r M(x) - R(x)$
- ♦ $T(x)$ est divisible par $G(x)$ (à vérifier par le récepteur !)

$$\text{CRC16 : } x^{16} + x^{15} + x^2 + 1$$

$$\text{CRC32 : } x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$





	0	0	0	0	1101101011 0000
0	0	0	0	1	101101011 0000
00	0	0	1	1	01101011 0000
000	0	1	1	0	1101011 0000
0000	1	1	0	1	101011 0000
00001	1	0	0	0	01011 0000
000011	0	0	1	1	1011 0000
0000110	0	1	1	1	011 0000
00001100	1	1	1	0	11 0000
000011001	1	1	1	0	1 0000
0000110011	1	1	1	0	0000
00001100111	1	1	1	1	000
000011001111	1	1	0	1	00
0000110011111	1	0	0	1	0
00001100111111	0	0	0	1	

⊕	0	1
0	0	1
1	1	0