le cnam

Algèbre de Boole, probabilités et arithmétique

Cours 12 Fonctions booléennes

- Vecteurs booléens

On se donne un entier $n \geq 1$. Un vecteur booléen $x = (x_1, x_2, \dots, x_j, \dots, x_n)$ est formé de n bits ; on a $x_j \in \mathbb{B}$. Son complément $\overline{x} = (\overline{x_1}, \overline{x_2}, \dots, \overline{x_j}, \dots, \overline{x_n})$ a des composantes $\overline{x_j} = 1 + x_j$ qui appartiennent aussi à $\mathbb{B} = \{0, 1\}$. Si on se donne un second vecteur booléen $y = (y_1, y_2, \dots, y_j, \dots, y_n)$, la borne inférieure $x \wedge y$ satisfait à l'égalité $(x_1, x_2, \dots, x_j, \dots, x_n) \wedge (y_1, y_2, \dots, y_j, \dots, y_n) = (x_1 \wedge y_1, x_2 \wedge y_2, \dots, x_j \wedge y_j, \dots, x_n \wedge y_n)$ et la borne supérieure $x \vee y$ suit une relation analogue : $(x_1, x_2, \dots, x_j, \dots, x_n) \vee (y_1, y_2, \dots, y_j, \dots, y_n) = (x_1 \vee y_1, x_2 \vee y_2, \dots, x_j \vee y_j, \dots, x_n \vee y_n)$. L'inégalité $x \leq y$ est satisfaite si et seulement si elle est vraie pour toutes les composantes : pour tout entier j entre 1 et n, $x_j \leq y_j$. Avec ces structures, l'ensemble des vecteurs booléens $\mathbb{B}^n$ est un treillis de Boole $(\mathbb{B}^n, \wedge, \vee, \overline{\ }, \leq, 0, 1)$. Le plus petit élément est le vecteur nul $0 = (0, 0, \dots, 0)$ et le plus grand élément le vecteur "un" dont toutes les composantes sont égales à 1 : $1 = (1, 1, \dots, 1)$. La structure $(\mathbb{B}^n, \wedge, \vee, \overline{\ }, 0, 1)$, où l'on a simplement retiré la référence à la relation d'ordre $\leq$, est appelée "algèbre de Boole".

Le cardinal de $\mathbb{B}^n$ est égal à 2^n et dans le cas $n = 2$ par exemple, on a la liste suivante des vecteurs booléens avec deux composantes : $(0, 0)$, $(0, 1)$, $(1, 1)$, $(1, 0)$.

- Fonction booléenne

Une fonction booléenne est une application de $\mathbb{B}^n$ dans $\mathbb{B}$. L'ensemble des fonctions booléennes est noté $\mathscr{F}_n$ ou $\mathbb{B}^{\mathbb{B}^n}$. On peut prendre un point de vue superficiel et considérer que ce n'est qu'un ensemble de fonctions entre deux ensembles finis. Toutefois, il faut observer que le nombre de fonctions booléennes croît très vite avec l'entier n.

Pour $n = 0$, on a $|\mathscr{F}_0| = 2$ puisque $|\mathbb{B}| = 2$.

Pour $n = 1$, on a $|\mathscr{F}_1| = 4$ et on explicite facilement ces quatre fonctions de l'ensemble des bits dans lui-même : la fonction nulle $\mathbb{B} \ni x \longmapsto 0 \in \mathbb{B}$, la fonction "identité" $\mathbb{B} \ni x \longmapsto x \in \mathbb{B}$, la fonction "un" $\mathbb{B} \ni x \longmapsto 1 \in \mathbb{B}$ et la plus intéressante, la fonction "non" $\mathbb{B} \ni x \longmapsto \overline{x} \in \mathbb{B}$.

Pour $n = 2$, on a $|\mathscr{F}_2| = 2^4 = 16$. Nous y reviendrons dans ce chapitre.

Pour $n = 3$, on a $|\mathscr{F}_3| = 2^8 = 256$ et pour $n = 4$, $|\mathscr{F}_4| = 2^{16} = 65\,536$. Il est bien entendu hors de question de faire des raisonnements fondés sur l'examen détaillé de toutes les fonctions possibles et nous approfondirons ce cas particulier au chapitre suivant.

Pour $n = 5$, on a $|\mathscr{F}_5| = 2^{32} = 4\,294\,967\,296$. Les méthodes d'analyse reposent sur l'algèbre ou sur des logiciels spécialisés, hors du sujet de ce cours.

- Porte logique “non”

La fonction logique “non” $\mathbb{B} \ni x \longmapsto \overline{x} \in \mathbb{B}$ ne demande qu'une variable. On la représente dans un langage graphique dit des “portes logiques” standardisé dans les années 1950 puis largement diffusé par des organismes américains comme l'American National Standards Institute (ansi) ou l'Institute of Electrical and Electronics Engineers (i trois e). La porte logique “non” (figure 1) symbolise la fonction $\mathbb{B} \ni x \longmapsto \overline{x} \in \mathbb{B}$. De manière implicite, le bit d'entrée x est à gauche de la porte et le bit de sortie $\overline{x}$ est à droite.

x	0	1
$\overline{x}$	1	0

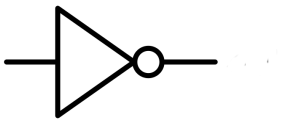


Figure 1. Fonction logique “non” représentée avec une table de vérité (à gauche), ou une porte logique (à droite).

- Premiers exemples de fonctions booléennes de deux variables

La fonction logique “ou” correspond à la fonction booléenne $\vee : \mathbb{B}^2 \ni (x,y) \longmapsto x \vee y \in \mathbb{B}$. On peut la représenter avec une table de vérité, un tableau à double entrée, encore appelé “diagramme de Karnaugh” ou un symbole spécialisé, la porte logique “ou” (figure 2).

(x,y)	00	01	11	10
$x \vee y$	0	1	1	1

ou	$x=0$	$x=1$
$y=0$	0	1
$y=1$	1	1

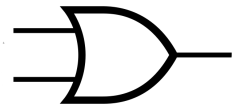


Figure 2. Fonction logique “ou” représentée avec une table de vérité (à gauche), un diagramme de Karnaugh (au centre) ou une porte logique (à droite).

Avec l'algèbre de Boole, la conjonction logique “et” devient la fonction booléenne $\wedge$ et cette fonction $\mathbb{B}^2 \ni (x,y) \longmapsto x \wedge y = xy \in \mathbb{B}$ se représente avec une table de vérité, un diagramme de Karnaugh ou une porte logique (figure 3).

(x,y)	00	01	11	10
$x \wedge y$	0	0	1	0

et	$x=0$	$x=1$
$y=0$	0	0
$y=1$	0	1

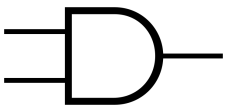


Figure 3. Fonction logique “et” représentée avec une table de vérité (à gauche), un diagramme de Karnaugh (au centre) ou une porte logique (à droite).

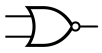
Enfin, l'implication logique “$\Rightarrow$” correspond à la fonction booléenne $\mathbb{B}^2 \ni (x,y) \longmapsto \overline{x} \vee y \in \mathbb{B}$. On peut présenter sa table de vérité et son diagramme de Karnaugh mais elle n'a pas de symbole bien répertorié chez les ingénieurs (figure 4).

(x,y)	00	01	11	10
$x \Rightarrow y$	1	1	1	0

$\Rightarrow$	$x=0$	$x=1$
$y=0$	1	1
$y=1$	1	0

Figure 4. Fonction d'implication logique “$\Rightarrow$” représentée avec une table de vérité (à gauche) ou un diagramme de Karnaugh (à droite).

Nous pouvons synthétiser ces différents vocabulaires dans le lexique qui suit.

Logique	vrai	faux	non	ou	et
algèbre de Boole	1	0	$\overline{\ }$	$\vee$	$\wedge$
portes logiques					

- D'autres fonctions booléennes de deux variables

On peut nier la fonction "ou", ce qui nous permet de définir la porte "non ou", appelée "nor" en anglais. Elle est présentée figure 5.

(x,y)	00	01	11	10
$\overline{x \vee y}$	1	0	0	0

non ou	$x=0$	$x=1$
$y=0$	1	0
$y=1$	0	0

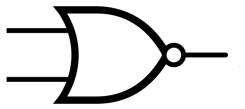


Figure 5. Fonction logique "non ou" représentée avec une table de vérité (à gauche), un diagramme de Karnaugh (au centre) ou une porte logique (à droite).

On peut également nier la fonction "et". On définit alors la porte "non et" qui se traduit par "nand" en anglais. Nous nous référons à la figure 6.

(x,y)	00	01	11	10
$\overline{x\,y}$	1	1	0	1

non et	$x=0$	$x=1$
$y=0$	1	1
$y=1$	1	0

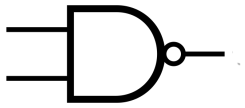


Figure 6. Fonction logique "non et" représentée avec une table de vérité (à gauche), un diagramme de Karnaugh (au centre) ou une porte logique (à droite).

Enfin, la fonction d'addition "+", avec $x+y=\overline{x}y \vee x\overline{y}$, donne lieu au symbole "xor" avec une dénomination anglo-saxonne. Nous renvoyons le lecteur à la figure 7.

(x,y)	00	01	11	10
$\overline{x}y \vee x\overline{y}$	0	1	0	1

+	$x=0$	$x=1$
$y=0$	0	1
$y=1$	1	0

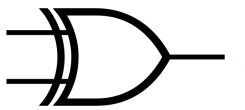


Figure 7. Fonction logique "+" représentée avec une table de vérité (à gauche), un diagramme de Karnaugh (au centre) ou une porte logique (à droite).

- Fonctions booléennes de deux variables

Nous pouvons représenter l'ensemble des fonctions boolénnes de deux variables avec une table de vérité. Nous choisissons, pour une raison qui ne sera explicitée qu'au chapitre suivant, l'ordre suivant pour l'ensemble des couples de bits : $(0,0)$, $(0,1)$, $(1,1)$, $(1,0)$. On obtient alors une liste des seize fonctions booléennes de deux variables présentées sous la forme d'une table de vérité. Nous constatons sur la Table 1 ci-dessous qu'il est possible d'exprimer toutes les fonctions booléennes de deux variables à l'aide uniquement des fonctions logiques "non", "ou" et "et".

(x,y)	0	$\overline{xy}$	$\bar{x}y$	xy	$x\bar{y}$	$\bar{x}$	$xy\vee\overline{xy}$	$\bar{y}$	y	$x+y$	x	$x\vee y$	$x\vee\bar{y}$	$\bar{x}\vee\bar{y}$	$\bar{x}\vee y$	1
00	0	1	0	0	0	1	1	1	0	0	0	0	1	1	1	1
01	0	0	1	0	1	1	0	0	1	1	0	1	0	1	1	1
11	0	0	0	1	0	0	1	0	1	0	1	1	1	0	1	1
10	0	0	0	0	1	0	0	1	1	1	1	1	1	1	0	1

Table 1. Liste des seize fonctions booléennes de deux variables.

Nous savons aussi qu'il est possible de réduire le nombre de fonctions qui servent à exprimer toutes les autres à deux, grâce aux lois de De Morgan puisque $xy=\overline{\bar{x}\vee\bar{y}}$ et $x\vee y=\overline{\bar{x}\bar{y}}$.
Mais on peut faire mieux et une seule fonction peut suffire pour générer toutes les autres. On parle alors de "porte universelle". C'est le cas des fonctions "non ou" et "non et". Nous pouvons montrer que l'opérateur de négation et l'une des deux portes logiques "ou" ou "et" peuvent être exprimés à l'aide uniquement d'une de ces portes. En effet, on a d'une part $\text{nor}\,(x,x)=\overline{x\vee x}=\bar{x}$ ainsi que $x\vee y=\overline{\text{nor}\,(x,y)}=\text{nor}\,\big(\text{nor}\,(x,y),\text{nor}\,(x,y)\big)$ et d'autre part $\text{nand}\,(x,x)=\overline{xx}=\bar{x}$ et $xy=\overline{\text{nand}\,(x,y)}=\text{nand}\,\big(\text{nand}\,(x,y),\text{nand}\,(x,y)\big)$

- Relation d'ordre dans l'ensemble des fonctions booléennes

On peut définir une relation d'ordre dans $\mathscr{F}_n$: $f\leq g$ si et seulement si $\forall x\in\mathbb{B}^n,\ f(x)\leq g(x)$.
Cette relation d'ordre permet de définir le treillis de Boole $(\mathscr{F}_n,\wedge,\vee,\,\bar{}\,,\leq,0,1)$, isomorphe au treillis $\mathbb{B}^{2^n}$. Il est illustré figure 8 pour $n=2$.

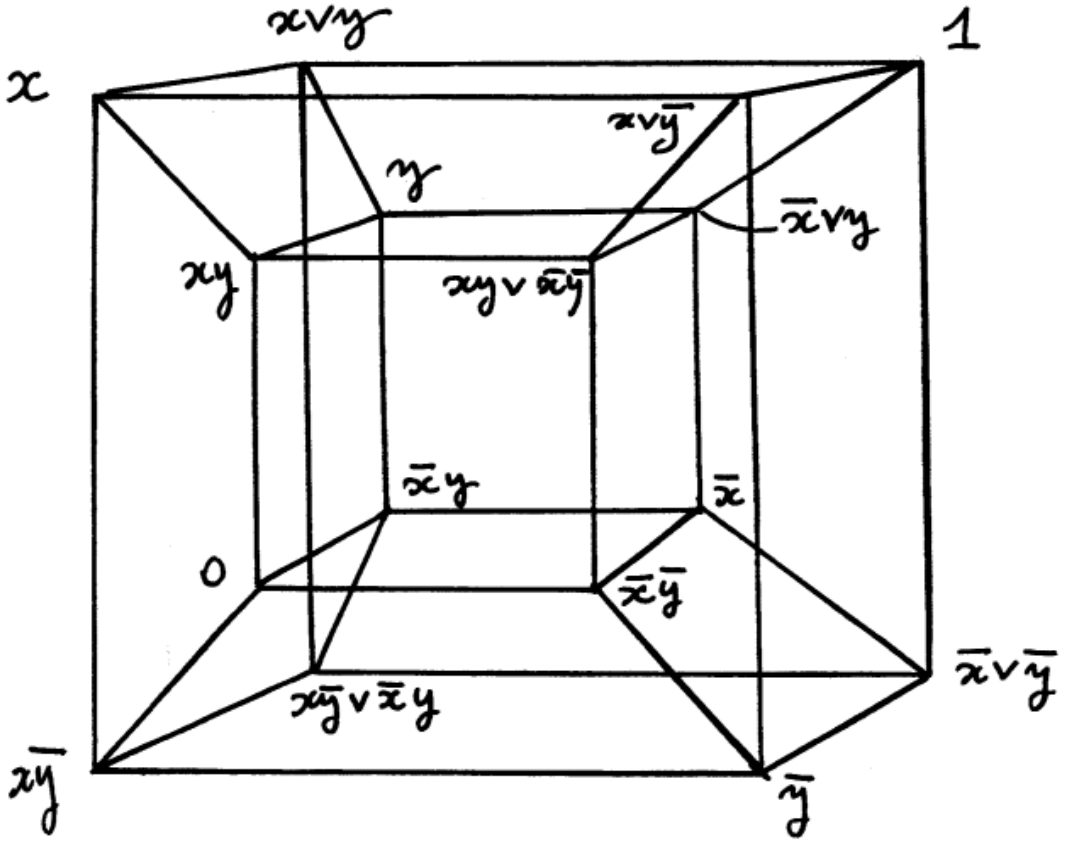


Figure 8. Treillis des fonctions booléennes de deux variables. Il est isomorphe au treillis $\mathbb{B}^4$ et son diagramme de Hasse est un 4-cube.

Le produit $fg = f \wedge g$ est la fonction $\mathbb{B}^n \ni x \longmapsto (f \wedge g)(x) = f(x) \wedge g(x) \in \mathbb{B}$, la borne supérieure $f \vee g$ se caractérise par l'ensemble des bornes supérieures des images :
pour tout $x \in \mathbb{B}^n$, $(f \vee g)(x) = f(x) \vee g(x) \in \mathbb{B}$ et le complément $\overline{f}$ associe à pour tout $x \in \mathbb{B}^n$ le bit complémentaire de $f(x)$: $\overline{f}(x) = \overline{f(x)}$. On remarque qu'il ne faut pas confondre $\overline{f}(x)$ et $f(\overline{x})$, même s'il sont parfois égaux ! Si on prend par exemple $f(x,y) = x \vee y$, on a $\overline{f}(x,y) = \overline{x}\,\overline{y}$ alors que $f(\overline{x},\overline{y}) = \overline{x} \vee \overline{y}$ et ces deux fonctions sont bien différentes.

- Chaîne de contacts

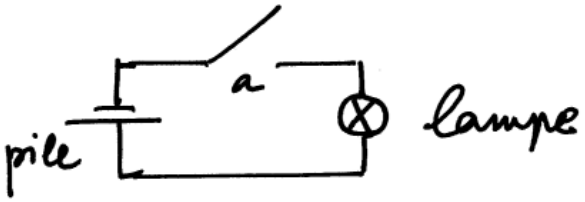


Figure 9. Chaîne de contacts fondamentale avec un simple interrupteur.

Il s'agit d'un mode de représentation qui fait un lien explicite avec les circuits électriques (figure 9). L'interrupteur a peut avoir deux positions : il peut être fermé, le courant ne passe pas et la lampe est éteinte. Dans ce cas, la variable booléenne a vaut 0. Il peut être au contraire ouvert ; alors le courant passe et la lampe est allumée. Dans ce cas, on a $a = 1$. On dit que la chaîne de contacts de la figure 9 réalise la fonction booléenne $f = a$. Il est utile aussi de modéliser un dispositif de type "anti-interrupteur". Dans ce cas, c'est la fonction complémentaire de la précédente qui est réalisée par la chaîne de contacts et $f = \overline{a}$.

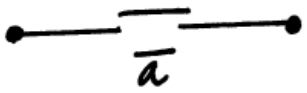


Figure 10. Chaîne de contacts de type anti-interrupteur : $f = \overline{a}$.

Le montage en série (figure 11) permet de réaliser la fonction logique "et" alors qu'un montage en parallèle (figure 12) permet d'implémenter la fonction logique "ou".

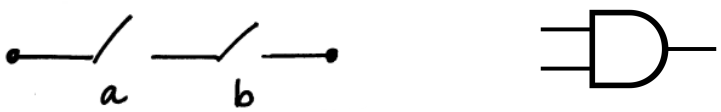


Figure 11. Un montage en série (à gauche) permet de réaliser la porte logique "et" : $f = ab$.

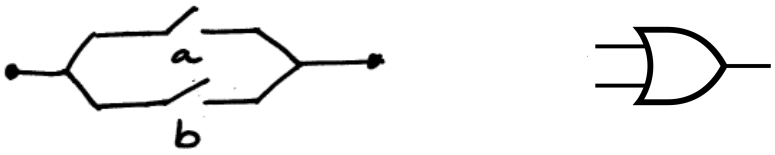


Figure 12. Un montage en parallèle (à gauche) permet de réaliser la porte logique "ou" : on a $f = a \vee b$.

- Synthèse de la fonction "+" (porte logique "xor")

À partir des portes logiques "non", "ou", "et", on peut faire la synthèse de la fonction booléenne $(x,y) \longmapsto x+y$ d'au moins deux façons : avec la formulation $x+y = x\overline{y} \vee \overline{x}y$ (voir la figure 13) ou avec la représentation $x+y = (x \vee y)\,(\overline{x} \vee \overline{y})$ proposée figure 14.

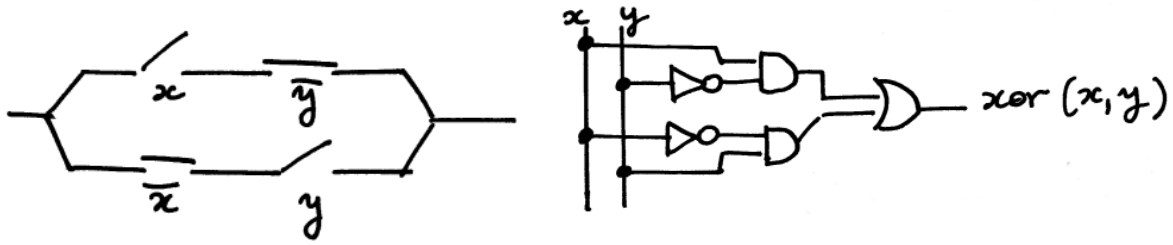


Figure 13. Une synthèse possible et la porte "+" (ou "xor") $x+y = x\overline{y} \vee \overline{x}y$ avec une chaîne de contacts (à gauche) et un réseau de portes logiques (à droite).

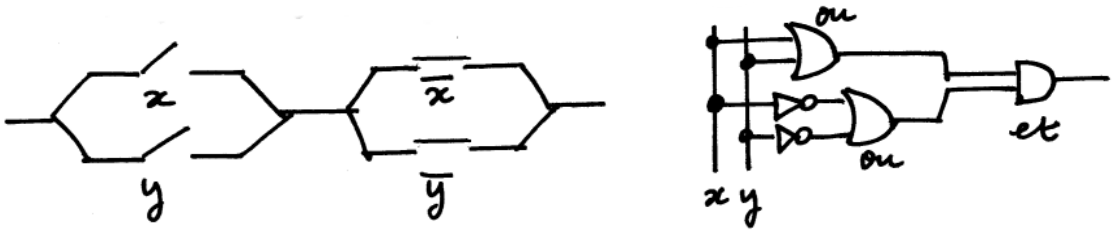


Figure 14. Une autre synthèse possible et la porte "+" (ou "xor") $x+y = (x\vee y)\,(\overline{x}\vee\overline{y})$ avec une chaîne de contacts (à gauche) et un réseau de portes logiques (à droite).

- Circuit de Shannon (1938)

Dans son article "A symbolic analysis of relay and switching circuits", Claude Shannon se propose de calculer la fonction logique f décrite par une chaîne de contacts présentée (aux notations près !) à la figure 15.

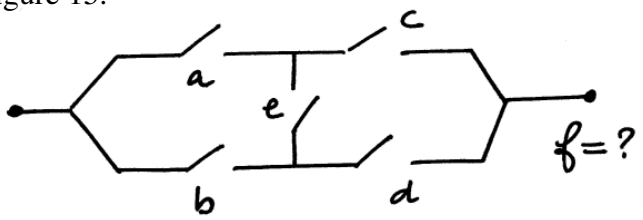


Figure 15. Circuit de Shannon. L'analyse consiste à étudier les deux cas possibles selon que l'interrupteur "e" est ouvert ou fermé.

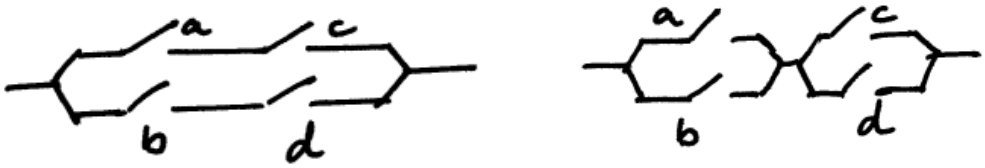


Figure 16. Circuit de Shannon. Chaîne de contacts pour la fonction $f_0 = f(a,b,c,d,0)$ (à gauche) et pour la fonction $f_1 = f(a,b,c,d,1)$ (à droite).

Si l'interrupteur est fermé ($e=0$), le courant ne passe pas dans le fil représenté par un trait vertical et dans ce cas, $f = f_0 = ac \vee bd$. Si l'interrupteur est ouvert ($e=1$), le courant passe dans le fil vertical et $f = f_1 = (a\vee b)\,(c\vee d)$. La fonction globale correspondant à la figure 15 est donc donnée par la relation $f = \overline{e}\,(ac\vee bd) \vee e\,(a\vee b)\,(c\vee d)$.

De façon très générale, le théorème de Shannon permet d'écrire une fonction booléenne avec l'aide des fonctions partielles f_0 et f_1. Nous l'énonçons pour trois variables et la généralisation est immédiate. Pour toute fonction booléenne f, on a $f(a,b,c) = \overline{a}\,f(0,b,c) \vee a\,f(1,b,c)$.

- Simplification des expressions algébriques des fonctions booléennes

Nous présentons dans ce qui suit la transformation de l'expression algébrique précédente $f = \overline{e}\,(ac \vee bd) \vee e\,(a \vee b)\,(c \vee d)$ en une expression qui ne fait apparaître une somme de produits ("sum of products" en anglais). On développe d'abord l'expression grâce à la distributivité de la loi "et" relativement à la loi "ou" : $f = (\overline{e}\,ac) \vee (\overline{e}\,bd) \vee (eac) \vee (eac) \vee (ebc) \vee (ebd)$. On utilise ensuite la commutativite : $f = (\overline{e}\,ac) \vee (eac) \vee (\overline{e}\,bd) \vee (ebd) \vee (eac) \vee (ebc)$ puis la distributivité dans l'autre sens pour faire apparaître la relation $e \vee \overline{e} = 1$:
$f = \big((\overline{e} \vee e)\,ac\big) \vee \big((\overline{e} \vee e)\,bd\big) \vee eac \vee ebc = ac \vee bd \vee ace \vee ade.$
On peut alors synthétiser cette nouvelle expression avec une chaîne de contacts composée de plusieurs sous-chaînes de circuits en parallèle. Le circuit obtenu est plus simple topologiquement mais qui comporte plus d'interrupteurs (figure 17).

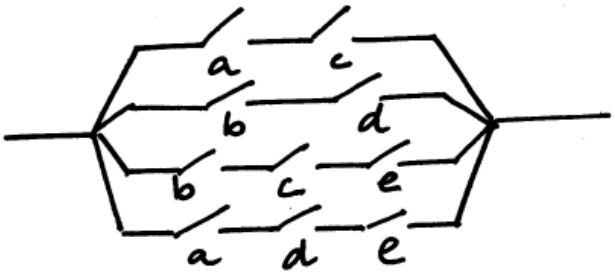


Figure 17. Synthèse du circuit de Shannon avec une chaîne de contacts qui met en évidence les divers trajets possibles du courant électrique dans le schéma initial de la figure 15.

La méthode de calcul présentée dans ce paragraphe est tout à fait générale. On peut aussi échanger les rôles des opérateurs logiques "ou" et "et" afin d'écrire une fonction booléenne avec un produit de sommes ("product of sums" en anglais) qui permet immédiatement une implémentation avec des circuits en série.

- Produits et "mintermes"

Nous avons vu au paragraphe précédent que nous avons pu écrire la fonction booléenne f comme somme de produits de deux ou trois facteurs. Par définition, un "minterme" dans $\mathscr{F}_n$ est décrit comme le produit de n facteurs différents de la forme $\widetilde{x_1}\,\widetilde{x_2} \ldots \widetilde{x_j} \ldots \widetilde{x_n}$ où $\widetilde{x_j}$ peut prendre les valeurs $\widetilde{x_j} = x_j$ ou $\widetilde{x_j} = \overline{x_j}$. On compte donc un total possible de 2^n mintermes et on peut démontrer que ce sont exactement les atomes du treillis $\mathscr{F}_n = \mathbb{B}^{\mathbb{B}^n}$.
Le cas particulier $n = 2$ est illustré figure 18. On retrouve également les quatre fonctions mintermes $\overline{x}\,\overline{y}$, $\overline{x}\,y$, $x\,y$ et $x\,\overline{y}$ au début de la Table 1. Leur table de vérité contient trois zéros et un seul "un".
Mais il est souvent intéressant d'exprimer une fonctions booléenne avec une somme de produits qui ne sont pas forcément des mintermes. C'est le cas par exemple dans l'exemple vu plus haut avec $f(a,b,c,d,e) = ac \vee bd \vee ace \vee ade$ qui contient des produits de deux ou trois variables alors que c'est une fonction de cinq variables.
Un produit dans $\mathscr{F}_n$ est une fonction booléenne de la forme $p = \xi_1\,\xi_2 \ldots \xi_j \ldots \xi_n$ où l'on a trois cas de figure possibles : $\xi_j = x_j$, $\xi_j = \overline{x_j}$ ou $\xi_j = 1$. Dans ce dernier cas, la variable x_j n'apparaît pas dans le produit. On compte donc 3^n produits dans l'algèbre de Boole des

fonctions booléennes de n variables. Dans le cas $n = 2$ par exemple, ce sont les fonctions 1, x, $\bar{x}$, y, $\bar{y}$ et les quatre mintermes $\bar{x}\bar{y}$, $\bar{x}y$, xy et $x\bar{y}$.

Nous rappelons enfin quelques relations toujours très utiles pour la simplifications des calculs algébriques : $xx = x$, $x\bar{x} = 0$, $x \vee \bar{x} = 1$, $x \vee xy = x$.

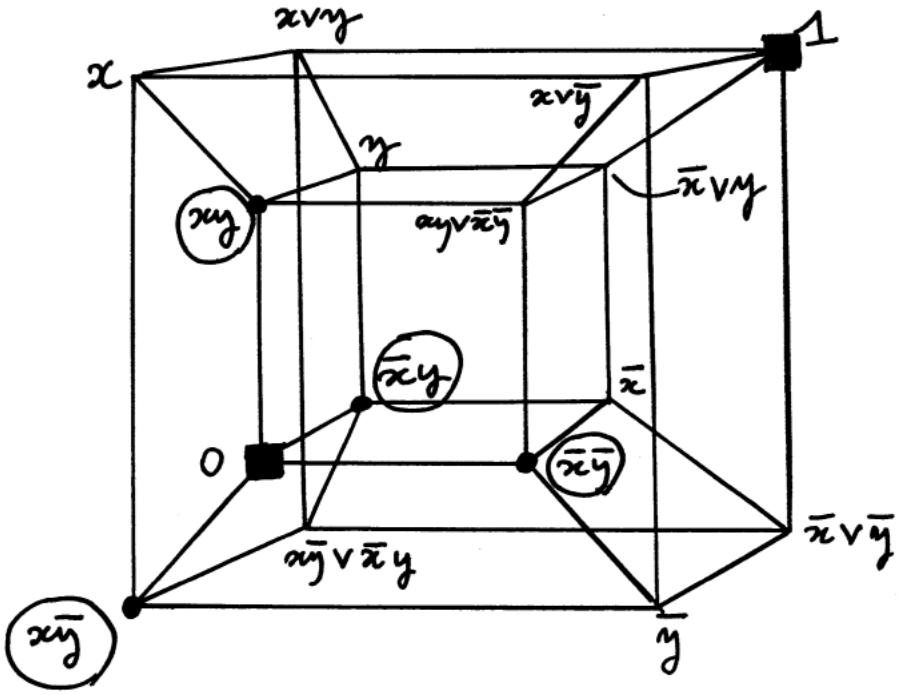


Figure 18. Treillis $\mathscr{F}_2 \sim \mathbb{B}^4$ des fonctions booléennes de deux variables. Les mintermes $\bar{x}\bar{y}$, $\bar{x}y$, xy et $x\bar{y}$ sont les atomes du treillis de Boole correspondant.

- Portes logiques matérielles

Nous terminons ce chapitre en rappelant rapidement quelques grandes étapes de la mise en œuvre matérielle des portes logiques, donc des machines automatiques destinées au calcul, les ordinateurs.

On compte essentiellement quatre prototypes parmi les tous premiers ordinateurs
"Zuse 3" (Allemagne, 1941, Konrad Zuse, 1910-1995), binaire avec virgule flottante,
"Colossus" (Grande Bretagne, 1943, Thomas Flowers, 1905-1998), binaire programmable, technologie des tubes à vide,
"Mark 1" (USA, IBM, 1944, Howard Aiken, 1900-1973), électromécanique, possibilités limitées de programmation,
"Electronic Numerical Integrator And Computer" (Eniac, USA, US ARmy, 1945, John Mauchly, 1907-1980), technologie des tubes à vide, registres décimaux, doit être recablé pour effectuer de nouveaux programmes.

La découverte de l'"effet transistor" aux "Bell Labs" en 1947 par John Bardeen (1908-1991), William Shockley (1910-1989) et Walter Brattain (1902-1987) et les propriétés des semi-conducteurs à base de silicium vont transformer de façon fondamentale l'informatique. Il n'est bien entendu pas possible ici de parler de façon rigoureuse du transistor, sujet qui mérite tout un

cours à lui tout seul. En deux mots, il est essentiel de retenir qu'un transistor, avec ses trois fils, permet d'implémenter une porte logique comme "non", "ou", "et", *etc*.

Le circuit intégré (1958) a permis à Jack Kilby (Texas Instruments, 1923-2005) de résoudre le problème dit de la "tyrannie des nombres" à cause du grand nombre de transistors nécessaires à la réalisation de fonctions logiques complexes. Une importante réduction du volume permet le regroupement su un même support de plusieurs fonctions logiques.

Le microprocesseur enfin, inventé en 1969 chez Intel par Marcian Hoff (né en 1937) et Federico Faggin (né en 1941), puis chez Motorola en 1970 par Chuck Peddle (1937-2019), a permis de regrouper tous les composants de plus en plus miniaturisés dans un unique boitier. Nous vivons toujours aujourd'hui à l'heure du microprocesseur. Les principales étapes sont résumées dans le tableau ci-dessous. En première approximation, il faut deux à cinq transistors pour réaliser une porte logique.

dénomination	année	nombre de transistors
Medium Scale Integration	1968	10 à 500
Large Scale Integration	1971	500 à 20000
Very Large Scale Integration	1980	$2\ 10^4$ à 10^6
Ultra Large Scale Integration	1984	plus d'un million
Intel Pentium	1993	3 millions
Pentium 4	2000	42 millions
Intel Core	2015	plus d'un milliard

Exercices

- Égalité de fonctions booléennes

À l'aide des règles de calcul dans le treillis $\mathbb{B}^n$, démontrer les relations suivantes

a) $\overline{a \vee b \vee bc} = \overline{a}\,\overline{b}$

b) $a \vee b\overline{c} = (a \vee b)(a \vee \overline{c})$

c) $(a \vee b)(b \vee c)(c \vee a) = ab \vee bc \vee ca$

d) $(a \vee b \vee \overline{c})(a \vee \overline{b} \vee c)(\overline{a} \vee b \vee c) = \overline{a}\,\overline{b}\,\overline{c} \vee ab \vee bc \vee ca.$

- Égalité de fonctions booléennes

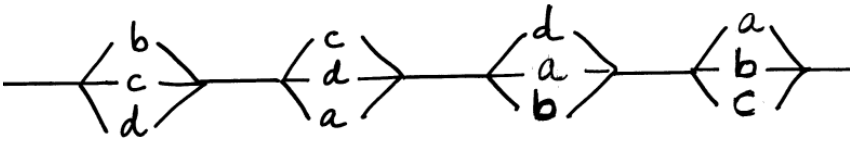
On pose $f(a,b,c,d) = (a \vee b)(c \vee \overline{d}) \vee (\overline{a} \vee b)(c \vee d)$ et
$g(a,b,c,d) = (a \vee b \vee c \vee d)(\overline{a} \vee b \vee c \vee \overline{d})$. Montrer que ces deux fonctions sont égales

a) en utilisant une table de vérité

b) à l'aide d'un calcul algébrique. $[f = g = b \vee c \vee a\overline{d} \vee \overline{a}d]$

- Une chaîne de contacts

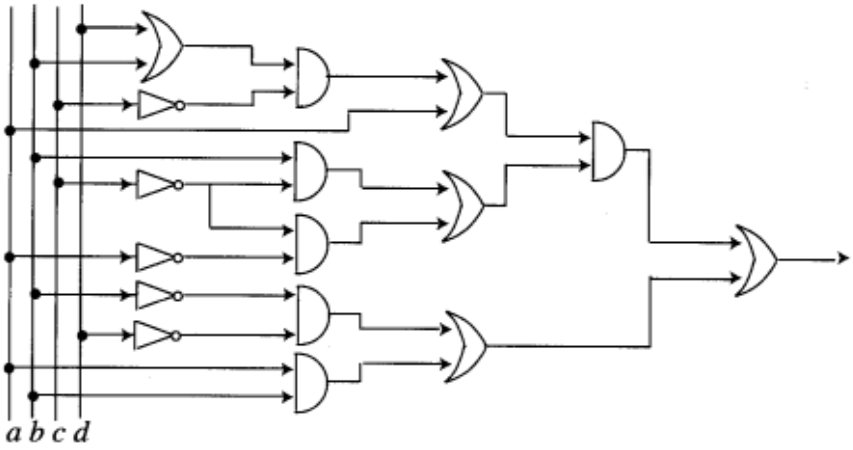
On se donne une chaîne de contacts définie par le graphe ci-dessous.



a) Écrire sous forme algébrique la fonction de transmission $f(a,b,c,d)$ définie par cette chaîne de contacts.
b) Montrer que l'on a $f=1$ si et seulement si au moins deux quelconques des interrupteurs laissent passer le courant. [$f = ab \vee ac \vee ad \vee bc \vee bd \vee cd$]
c) Proposer une chaîne de contacts équivalente ne contenant que huit interrupteurs.

- Circuit de portes logiques [d'après Gil *et al.*]

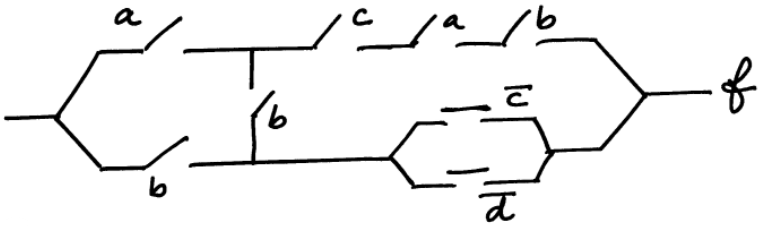
On se donne un circuit de portes logiques à l'aide du graphe ci-dessous.



a) Identifier sous forme d'une expression algébrique la fonction de transmission qui est une fonction boolénne f des quatres variables a, b, c, d.
b) Simplifier l'expression de cette fonction et montrer que $f(a,b,c,d) = \overline{a}\,\overline{c} \vee \overline{b}\,\overline{d} \vee ab$.
c) En déduire une synthèse à l'aide d'une chaîne de contacts comprenant six interrupteurs.
d) Reprendre la question b) en utilisant une table de vérité.

- Une autre chaîne de contacts

On se donne une chaîne de contacts définie par le graphe ci-dessous



Elle définit une fonction booléenne de quatre variables $f = f(a,b,c,d)$.
a) Donner une expression algébrique de la fonction f. [$f = abc \vee b\overline{cd}$]
b) Simplifier cette expression algébrique et démontrer que la fonction f peut être synthétisée avec les quatre portes suivantes : une triple porte "et", une double porte "et", une porte "ou" et une porte "non et".