ANNEXE 1 LOGIQUE ET ALGEBRE DE BOOLE

La logique constitue en quelque sorte la grammaire des mathématiques, on y traite de la forme des expressions utilisées, et de la validité des raisonnements formulés. Outre le fait que son étude a pour effet d'inciter à une rédaction rigoureuse des spécifications, des calculs et de l'enchaînement des raisonnements, elle possède des applications en électronique, automatique et informatique, par le biais de l'algèbre de Boole.

Alphabet du langage mathématique, pour éviter les confusions, on a besoin d'un langage précis, à commencer par les symboles qu'on utilise et qui constituent un alphabet.
Constantes : les chiffres 0 1 2 3 4 5 6 7 8 9, qui eux même s'assemblent en "mots" conformément au principe de la numération de position, pour noter des valeurs numériques; des symboles désignant des valeurs utiles telles que π (3,14159...) , e (2,718... base des logarithmes néperiens); mais aussi des symboles attachés à des ensemble remarquables, Ø pour l'ensemble vide, N, Z, Q, R, C pour les nombres respectivement entiers positifs, entiers relatifs, rationnels (ou encore fractionnaire), réels, et complexes; $\aleph_0$ (aleph) pour l'infini dénombrable (cardinal de N); $+\infty$ et $-\infty$ pour borner l'ensemble R.
Variables : l'écriture d'expressions mathématiques a besoin d'un réservoir inépuisable de signes pouvant être substitués par des constantes, aussi prend-t-on les minuscules a, b, c ... a', b', c',... a", b", c",... indicées a1, a2, a3,... et de même avec les majuscules A, B, C,..., les lettres grecques α, β, γ, ... ,Γ, Δ, Φ ... Q S D F K
Symboles de fonctions : ce sont les "opérations" à une variable comme l'extraction de racine carrée $\sqrt{}$, de valeur absolue | | , de complémentation $\complement$, mais aussi des abréviations qu'il faut prendre comme des symboles élémentaires sin, cos, exp, Ln, ...; des fonctions à deux variables arithmétiques +, - ,... ; ensembliste $\cap$, $\cup$, Δ,...
Symboles de relations : =, <, >, $\neq$, $\leq$, $\perp$, //, $\in$, $\notin$, $\supset$, ... ont le rôle d'indiquer une proposition élémentaire.
Connecteurs logiques : constants 0 (le faux, ou contradiction), 1 (le vrai ou tautologie); à une place, la négation notée $\neg$ ou par un surlignement; à deux places, conjonction "et" $\wedge$, disjonction "ou" $\vee$, implication $\Rightarrow$, équivalence $\Leftrightarrow$...
Quantificateurs $\exists$ (il existe au moins un), $\forall$ (quelquesoit) et aussi ! (!x P(x) signifiant qu'un élément au plus vérifie la proposition P)
Symboles séparateurs tels que les parenthèses, crochets, accolades, la virgule etc ...
Termes ou expressions. Les constantes et variables sont les termes les plus simples. La juxtaposition d'un symbole fonctionnel et de termes séparés, constitue un nouveau terme, c'est-à-dire une expression pouvant recevoir une valeur dans la mesure où les variables en ont une et où la nature et le nombre de termes composant ce nouveau terme est en accord avec l'acceptation de chaque symbole de fonctions. Ainsi par exemple sin(π), a+b, $\sqrt{x}$, x^y (ou x^y), cos[$(2x - 1)^3$] ,e^{x^2} et $\frac{5(y - 8)}{2(x^2 + \sqrt{3})}$ sont des termes.
Il faut noter à ce propos que le symbole de multiplication est bien souvent omis, et que le symbole de division sous forme de barre de fraction permet de se dispenser de parenthèses. Rappelons de plus, parmi toutes les règles de priorité des opérateurs, que l'exponentiation est prioritaire sur la multiplication, laquelle l'emporte sur l'addition, par exemple $5 + 4*3^2$ a pour résultat 41 et non pas 5 plus le carré de 12, ni le carré de 17.
Les termes ou expressions sont donc les expressions bien formées suivant des règles précises qu'il serait trop long d'énumérer ici, mais apprises par l'usage. On peut donner π + - sin /5 ou $\cap$ A $\cup$ B comme exemples de "mots" mathématiques ne correspondant pas à des termes, il ne peuvent être évalués.
Propositions mathématiques ou énoncés. Les propositions "atomiques" sont le vrai, le faux, et tous les "mots" écrits (la plupart du temps) avec un symbole de relation binaire et deux termes appartenant aux types sur lesquels portent la relation. Ainsi $3 < 5$, $x \neq y$, $(a + b)^2 = a^2 + b^2 - 3ab$, $x \in$ N, $14 + 5 > 53$, D || D' sont des énoncés atomiques, certains sont vrais, d'autres faux, et les autres enfin dépendent de la valeur des variables y figurant. Une proposition atomique correspond à une phrase simple du type sujet - verbe - complément, les équations f(x) = 0 sont des cas de proposition où le verbe peut se prononcer par "est égal à". Les propositions structurées sont construites à partir des propositions atomiques en les assemblant grâce aux connecteurs logiques (non, et, ou, $\Rightarrow$ pour se limiter) et aux quantificateurs.
Exemples: $x = y \Rightarrow x^2 = y^2$, $ab = 0 \Rightarrow (a = 0)$ ou $(b = 0)$, $\exists x \ \forall y \ (x < y)$ sont des propositions.
Proposition close : toutes les variables y figurant sont "muettes", ou "liées" par un quantificateur ce qui signifie que le sens de la proposition ne change pas si on les remplace par d'autre variables, ainsi :
$\forall x \ \forall y \ (x = y \Rightarrow y = x)$ est close, $\forall x \ (x = y)$ ne l'est pas puisque dépendant de y.
Valeurs de vérité. Le principe de la logique binaire est d'attribuer à chaque proposition close une valeur de vérité qui ne peut être que le vrai (noté 1) ou bien le faux (noté 0), c'est le principe du "tiers exclu" qui n'est plus vrai dans d'autres logiques (à trois valeurs, probabiliste, ...). Ce "connecteur" est défini par sa "table de vérité" à savoir $\neg$P vaut 0 quand P vaut 1, et 1 quant P vaut 0, est donc contraire à P, toute proposition prenant la valeur opposée de celle de P. Puis en notant $\wedge$ la conjonction "et", $\vee$ la disjonction "ou", $\Rightarrow$ l'implication, $\Leftrightarrow$ l'équivalence logique, on définit de même l'exclusion réciproque "ou bien" notée $\oplus$ ici, et le symbole de Sheffer noté par une barre |.

P	Q	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$	$P \Leftarrow Q$	$P \oplus Q$	$P \mid Q$
1	1	1	1	1	1	1	0	0
1	0	0	1	0	0	1	1	1
0	1	0	1	1	0	0	1	1
0	0	0	0	1	1	1	0	1

On retiendra que la conjonction (P et Q) n'est vraie que si les deux sont vraies simultanément, en revanche, la disjonction n'est fausse que si les deux sont fausses. L'implication $P \Rightarrow Q$ n'est fausse que dans le cas où le vrai entrainerait une conclusion fausse. L'équivalence de P et de Q n'est vraie que si P et Q ont les mêmes valeurs, alors que l'exclusion réciproque $P \oplus Q$ n'est vraie que si l'une est vraie tandis que l'autre ne l'est pas.

Propriétés algébriques des connecteurs. Les tautologies, sont des propositions closes (mêmes si quelquefois les quantificateurs sont omis parceque sous-entendus) vraies. Les premières tautologies du calcul propositionnel sont tout d'abord (facile à vérifier) : **La négation est involutive $\neg(\neg P) \Leftrightarrow P$**

Exemple : "vous n'etes pas sans savoir que ..." équivaut (en négligeant les nuances) à "vous savez...".

Lois de Morgan, cherchons à dresser les tables de vérité pour la négation d'une conjonction ou la négation d'une disjonction, en examinant les quatre cas de distributions de valeurs pour deux propositions P et Q, il suffit de vérifier que les sixième et septième colonnes sont identiques (puis les deux dernières) pour affirmer le résultat.

P	Q	$\neg P$	$\neg Q$	$P \wedge Q$	$\neg(P \wedge Q)$	$\neg P \vee \neg Q$	$P \vee Q$	$\neg(P \vee Q)$	$\neg P \wedge \neg Q$
1	1	0	0	1	0	0	1	0	0
1	0	0	1	0	1	1	1	0	0
0	1	1	0	0	1	1	1	0	0
0	0	1	1	0	1	1	0	1	1

La négation d'une conjonction est la disjonction des négations $\neg$(P ou Q) $\Leftrightarrow \neg P$ ou $\neg Q$

La négation d'une disjonction est la conjonction des négations $\neg$(P ou Q) $\Leftrightarrow \neg P$ et $\neg Q$

Exemple : Il est possible d'envoyer un colis par la poste s'il ne dépasse pas 3 kg en poids, ni 60 cm dans sa plus grande dimension, alors un colis est refusé s'il fait plus de 3 kg ou plus de 60 cm, mais s'agissant du "ou" disjonctif et donc non exclusif, cela signifie trois éventualités élémentaires (plus de 3 kg et moins de 60 cm) ou bien (moins de 3 kg et plus de 60 cm) ou bien (plus de 3 kg et plus de 60 cm).

Structure de treillis distributif

Il est évident que les connecteurs $\wedge$ et $\vee$ sont commutatifs : $P \wedge Q \Leftrightarrow Q \wedge P$ $P \vee Q \Leftrightarrow Q \vee P$

Idempotence, c'est la propriété $P \wedge P \Leftrightarrow P$ $P \vee P \Leftrightarrow P$

Associativité de $\wedge$ et $\vee$: $P \wedge (Q \wedge R) \Leftrightarrow (P \wedge Q) \wedge R$ et $P \vee (Q \vee R) \Leftrightarrow (P \vee Q) \vee R$

Distributivités mutuelles : $P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$ et $P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$

On voit que la conjonction se comporte à l'égard de la disjonction comme la multiplication à l'égard de l'addition, mais ici l'inverse est aussi vrai,. Ces tautologies se démontrent par une table détaillant les 8 cas.

Eléments neutre et absorbant : 0 et 1 désignant respectivement le faux et le vrai, et P étant une proposition quelconque, on vérifie grâce aux définitions des connecteurs que :

$0 \wedge P \Leftrightarrow 0$ $0 \vee P \Leftrightarrow P$ 0 est absorbant pour $\wedge$, et neutre pour $\vee$.

$1 \wedge P \Leftrightarrow P$ $1 \vee P \Leftrightarrow 1$ 1 est neutre pour $\wedge$, et absorbant pour $\vee$.

Enfin les deux propriétés $P \wedge \neg P \Leftrightarrow 0$ et $P \vee \neg P \Leftrightarrow 1$ font du calcul prositionnel un "treillis de Boole".

Propriétés de l'implication. La définition donnée par la table de vérité plus haut peut paraître surprenante, mais elle s'explique par le fait que P entraîne Q signifie que chaque fois que P est vrai, Q doit l'être aussi, en d'autres termes il ne peut y avoir P sans Q, écrivons formellement cette dernière assertion, et par une suite d'équivalence, cherchons à la transformer : (P et non Q) est faux $\Leftrightarrow \neg (P \wedge \neg Q) \Leftrightarrow \neg P \vee Q$, en cherchant alors la table de vérité de $\neg P \vee Q$, on voit que cette dernière n'est fausse que si P est vrai et Q est faux, ce qui est la définition donnée plus haut de l'implication. La négation de $P \Rightarrow Q$ est donc $P \wedge \neg Q$.

Contraposée d'une implication. Pour deux propositions P et Q formons la table de $\neg Q$, $\neg P$, $\neg Q \Rightarrow \neg P$, puis comparons cette dernière distribution de valeurs à celle de $P \Rightarrow Q$

P	Q	$\neg Q$	$\neg P$	$P \Rightarrow Q$	$\neg Q \Rightarrow \neg P$	$P \wedge \neg Q$	0	$(P \wedge \neg Q) \Rightarrow 0$
1	1	0	0	1	1	0	0	1
1	0	1	0	0	0	1	0	0
0	1	0	1	1	1	0	0	1
0	0	1	1	1	1	0	0	1

Nous obtenons alors un premier résultat :

Démontrer une implication $P \Rightarrow Q$ est équivalent à démontrer sa contraposée $\neg Q \Rightarrow \neg P$

Exemple, pour deux réels x, y, en élevant au cube les deux membre de la première égalité, on obtient :

$\sqrt[3]{x} = \sqrt[3]{y} \Rightarrow x = y$ est vraie, donc sa contaposée aussi : $x \neq y \Rightarrow \sqrt[3]{x} \neq \sqrt[3]{y}$ cette dernière implication s'énonce par le fait que la racine cubique est une fonction injective.

En relisant le dernier tableau on constate que l'implication $P \Rightarrow Q$ est encore équivalente à la dernière colonne, or $(P \wedge \neg Q) \Rightarrow 0$ peut se lire de la manière suivante : "P et non Q entraîne une contradiction".
C'est le raisonnement par l'absurde, pour prouver une implication, on suppose l'hypothèse vraie et la conclusion fausse, alors on en tire une contradiction.
Exemple, le triangle de côtés mesurés par 3, 5, 7 n'est pas rectangle. En effet, supposons que le contraire de la conclusion anoncée soit vrai, c'est-à-dire que le triangle soit rectangle, en ce cas le théorème de Pythagore permet d'affirmer que le carré 49 de l'hypothénuse (le plus grand des trois) est égal à la somme des deux autres, soit 9 + 25, on déduit donc de $P \wedge \neg Q$, le fait que 49 = 34 ce qui constitue une proposition fausse (une contradiction).
Autre exemple, pour démontrer dans R que $(ab = 0) \Rightarrow (a = 0) \vee (b = 0)$, on suppose que ab = 0 et le contraire de la conclusion, mais d'après les lois de Morgan, ce contraire va s'écrire $(a \neq 0) \wedge (b \neq 0)$. N'étant pas nuls ces deux réels ont donc des inverses, soit a^{-1} celui de a, alors l'hypothèse ab = 0 donne $(a^{-1}a)b = 0$ soit b = 0 ce qui est contradictoire avec $b \neq 0$.
Opérateur de Sheffer P | Q peut être défini comme $\neg(P \wedge Q)$, cette proposition est vraie sauf si P et Q sont vraies en même temps. On en déduit que $P|P \Leftrightarrow \neg P$ et on peut démontrer que tous les connecteurs logiques peuvent être réécrits grâce à lui (voir en exercice). Cette propriété présente un intérêt en électronique, dans la mesure où toutes les "portes logiques" peuvent être montées uniquement avec la "porte NAND" réalisant précisément le connecteur de Sheffer. L'opérateur de Peirce défini par $\neg(P \vee Q)$ peut jouer le même rôle.

Quantificateurs Soit P(x) une proposition mathématique où x est une variable libre, par exemple x = 5, ou bien $x/2 \in N$, $x^2 - 5x + 6 = 0$, $x = x + 1$, ... dans tout domaine fixé à l'avance, par exemple R, ces propositions ont un "ensemble de vérité" défini comme le sous-ensemble où la variable libre prenant ses valeurs, rend P(x) vrai. Pour la première, il s'agit de l'ensemble {5}, pour la seconde, l'ensemble de tous les entiers positifs pairs, pour la troisième on vérifiera que c'est {2 , 3}, et enfin pour la dernière c'est l'ensemble vide ø.
Etant donné un ensemble de référence E, E_P l'ensemble de vérité d'une proposition à une variable libre P, on définit grâce aux quantificateurs universels et existenciels, deux nouvelles propositions :
$\forall x$ P(x) par le fait que $E_P = E$ et $\exists x$ P(x) par le fait que $E_P \neq \emptyset$
D'après le principe du tiers exclu, une proposition P(x), pour une valeur déterminée de x, est soit vraie, soit fausse, il est donc évident que les sous-ensembles E_P et $E_{\neg P}$ sont complémentaires, en ce cas la négation de la première va s'écrire : $\neg(\forall x \ P(x)) \Leftrightarrow \neg (E_P = E) \Leftrightarrow E_P \neq E \Leftrightarrow E_{\neg P} \neq \emptyset \Leftrightarrow \exists x \ \neg P(x)$
en clair, cela signifie que nier un fait universel, c'est affirmer l'existence au moins une fois du contraire.
Négation des propositions quantifiées $\neg(\forall x \ P(x)) \Leftrightarrow \exists x \ \neg P(x)$ et $\neg(\exists x \ P(x)) \Leftrightarrow \forall x \ \neg P(x)$
Ces règles de négation vont pouvoir s'enchaîner, ainsi à cause des lois de Morgan la négation de :
$\forall x \ \forall y \ \exists r \ \exists t \ \forall r' \ \forall t' \ x = r\cos(t) \wedge y = r\sin(t) \wedge [(x = r'\cos(t) \wedge y = r'\sin(t) \wedge r \neq 0) \Rightarrow (r = r') \wedge (t = t')]$
est-elle $\exists x \ \exists y \ \forall r \ \forall t \ \exists r' \ \exists t' \ x \neq r\cos(t) \vee y \neq r\sin(t) \vee [x = r'\cos(t) \wedge y = r'\sin(t) \wedge r \neq 0 \ \wedge ((r \neq r') \vee (t \neq t'))]$
Ces propriétés sont souvent utilisées, notamment pour prouver que quelquechose est faux, en donnant un "contre-exemple", à savoir, un exemple du contraire. Prenons la proposition $\forall x \in R \ x^2 - 1 > 0$, elle est fausse, tout simplement parceque son contraire $\exists x \in R \ x^2 - 1 \leq 1$ est vrai, ne serait-ce qu'à cause de x = 0. On dira que 0 est un contre-exemple pour la proposition $\forall x \in R \ x^2 - 1 > 0$.
Supposons maintenant que l'on veuille vérifier si la divisibilité dans N est un ordre total ou non, en notant par une simple barre | cette relation, donc par la proposition 3 | 12, le fait que 3 divise 12. Que la relation soit totale s'exprimerait par $\forall x \in N \ \forall y \in N \ (x \mid y) \vee (y \mid x)$ son contraire s'exprimera donc par :
$\exists x \in N \ \exists y \in N \ \neg(x \mid y) \wedge \neg(y \mid x)$, ce qui s'énonce : il existe au moins deux entiers tels que le premier ne divise pas le second, ni l'inverse. Cette dernière proposition est vraie pour 3 et 7 par exemple, (3, 7) est un contre-exemple montrant que la divisibilité dans N n'est pas une relation totale.

Application à la théorie élémentaire des ensembles
Dans un ensemble quelconque E, toute partie A est l'ensemble de vérité d'au moins une proposition, ne serait-ce que de la proposition $x \in A$. Les opérations ensemblistes (complémentaire d'une partie, intersection, union, et différence symétrique) peuvent être définies à partir des connecteurs logiques : $\complement E_P = E_{\neg P}$
$E_P \cap E_Q = E_{P \wedge Q}$ $E_P \cup E_Q = E_{P \vee Q}$ $E_P \Delta E_Q = E_{P \oplus Q}$
Plus traditionnellement, on définira par exemple l'intersection : $A \cap B = \{ x \in E \mid x \in A \wedge x \in B\}$
l'intérêt de ces définitions est que toutes les propriétés établies sur les connecteurs logiques se transportent immédiatement sur les opérations ensemblistes, ainsi peut-on dire que l'intersection et l'union sont des opérations idempotentes, commutatives, associatives et mutuellement distributives.
Cas particuliers: E est l'ensemble de vérité de la proposition vraie 1, et ø est l'ensemble de vérité de la proposition fausse 0. A ce propos on peut démontrer que l'ensemble vide ø est inclu dans tout ensemble E, cette proposition s'écrit : $\forall x \ (x \in \emptyset) \Rightarrow (x \in E)$, son contraire sera donc : $\exists x \ (x \in \emptyset) \wedge \neg (x \in E)$, or cette dernière est fausse car on ne peut trouver aucun élément vérifiant $(x \in \emptyset)$.

Algèbres de Boole, définition Une algèbre de Boole est une structure algébrique (B, +, .) formée par un ensemble muni de deux lois binaires notée + et ., ayant les propriétés suivantes :
Idempotence $\forall a \quad a + a = a$ et $aa = a$
Commutativité $\forall a \; \forall b \quad a + b = b + a$ et $ab = ba$
Associativité $\forall a \; \forall b \; \forall c \quad a + (b + c) = (a + b) + c$ et $a(bc) = (ab)c$
Distributivités $\forall a \; \forall b \; \forall c \quad a (b + c) = (ab) + (ac)$ et $a + (bc) = (a + b)(a + c)$
Complémentation $\exists\, 0 \; \exists\, 1 \; \forall x \quad \exists! \bar{x} \quad x + 0 = x$ et $1x = x$ et $x + \bar{x} = 1$ et $x\bar{x} = 0$
Le calcul propositionnel avec les deux lois binaires de la disjonction et conjonction, la contradiction 0, la tautologie 1, et la loi "unaire" de négation, répond à toutes ces propriétés. Afin de conserver les habitudes de calculs acquises on note souvent multiplicativement la conjonction, et de façon additive la disjonction, aussi la proposition $P \wedge (Q \vee R)$ sera-t-elle notée $P(Q + R)$. Il est plus commode également, comme cela se fait dans le calcul des probabilités, de noter la négation avec une barre. Pour un ensemble quelconque E, l'ensemble de ses parties P(E) muni de l'union et l'intersection, est un exemple d'algèbre de Boole.
Autres propriétés. A partir de la définition d'une algèbre de Boole, on démontre facilement :
Propriétés d'absorbtion $\forall x \quad x + 1 = 1$ car $x + 1 = x + (x + \bar{x}) = (x + x) + \bar{x} = x + \bar{x} = 1$
$\forall x \quad x0 = 0$ car $x0 = x (x\bar{x}) = (xx)\bar{x} = x\bar{x} = 0$ et $\forall x \; \forall y \quad x + xy = x$ car $x + xy = x1 + xy = x(1 + y) = x1 = x$
Propriété de consensus $\forall x \; \forall y \quad x + \bar{x}y = x + y$ car $x + \bar{x}y = (x + xy) + \bar{x}y = x + (x + \bar{x})y = x + y$
On en déduit d'autres propriétés telles que la règle de redondance : dans une somme booléenne, si deux termes contiennent x, pour le premier et $\neg x$ pour le second, alors on peut ajouter ou supprimer tout terme ne contenant ni l'un ni l'autre. Exemple : $xyz + \bar{x}yz + yz + \bar{y}z = xyz + \bar{x}yz$
Involution et lois de Morgan $\bar{\bar{x}} = x$ et $\overline{x + y} = \bar{x} \, . \, \bar{y}$ et $\overline{xy} = \bar{x} + \bar{y}$
Enfin si on définit $x \oplus y = x\bar{y} + \bar{x}y$ alors $(B, \oplus, .)$ est un anneau (voir en exercice).
Exemple , les conditions d'accorder l'entrée au demi-tarif dans un musée sont les suivantes :
Toute personne, non étudiant, de plus de 60 ans ne faisant pas partie de l'association des amis du musée, tout étudiant non membre de l'association, toute personne de moins de 60 ans et membre de l'association, toute personne de plus de 60 ans, ne faisant pas partie de l'association mais au chômage, tout étudiant de moins de 60 ans et non chômeur, toute personne membre de l'association au chômage, toute personne de plus de 60 ans, membre de l'association, non au chômage.
Quelles sont, simplifiées les conditions d'attribution de cette prime ? Exprimons par h la proposition "non étudiant", par v, "plus de 60 ans", par a "membre de l'association", et par r, "non chômeur". La condition s'écrit :
$x = hva + \bar{h} \, . \, a + \bar{v}a + v\bar{a} \, . \, r + \bar{h} \, . \, \bar{v}r + a\bar{r} + var$
$x = (hv + \bar{h}) \, a + \bar{v}a + v\bar{a} \, . \, \bar{r} + \bar{h} \, . \, \bar{v}r + a(\bar{r} + vr)$ en factorisant,
$x = (v + \bar{h} + v\bar{r}) \, a + \bar{h} \, . \, \bar{v}r + a(\bar{r} + v + \bar{v})$ factorisation et consensus,
$x = (v + \bar{h})a + \bar{h} \, . \, \bar{v}r + a(\bar{r} + 1)$
$x = (v + \bar{h})a + a$ par redondance,
$x = a + v + \bar{h}$ par consensus.
En d'autres termes, il s'agit des amis du musée, des plus de 60 ans, et des étudiants.

Simplification des expressions booléennes. L'exemple précédent a montré à quel point une expression booléenne pouvait être simplifiée. Le but de ce problème, essentiel à la réalisation des circuits électroniques, est d'arriver à minimiser le nombre d'opérations présentes dans l'expression. Généralement, on souhaite arriver à une expression écrite sous forme normale disjonctive, c'est-à-dire, avec les notations + et . , à une somme de monômes , lesquels mônomes sont des produits des variables figurant dans l'expression, ou de leur négation. Un théorème de logique permet d'affirmer que cette forme normale disjonctive existe toujours et qu'elle est unique.

Méthode de Karnaugh: On place sur un tableau, toutes les occurrences du vrai (notée par 1) pour l'expression f à simplifier, soit d'après sa table de vérité, soit d'après sa forme normale disjonctive. Les lignes ou colonnes remplies de "1" sont celles qui apportent des simplifications, de la manière suivante; prenons pour exemple une expression f de trois variables x, y et z.
$f(X, Y, Z) = \bar{X}\,\bar{Y}\,Z + \bar{X}YZ + X\bar{Y}Z + X\bar{Y}\,\bar{Z} + \bar{X}\,Y\,Z + X\,Y\,Z$, on montre que c'est $Y + X\bar{Y}\,\bar{Z} + \bar{X}\,\bar{Y}Z$

Z \ XY	00	01	11	10
0		1	1	1
1	1	1	1	

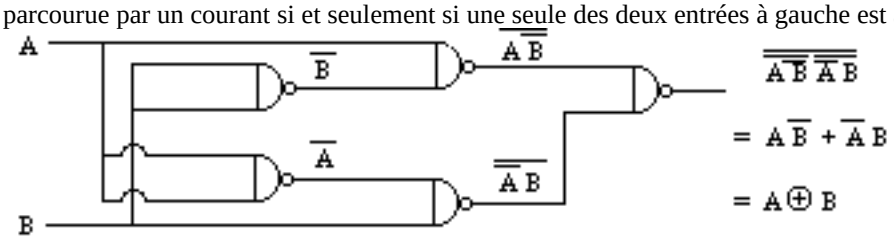
Le carré entouré correspond à tous les cas possibles pour X et Z lorsque Y = 1, on peut donc simplifier l'expression f en disant que c'est la disjonction de Y et de deux autres termes.
Soit à présent l'expression à quatre variables :

$h(R , S , T , U) = \overline{R}\,S\,\overline{T}\,\overline{U} + R\,S\,\overline{T}\,\overline{U} + R\,\overline{S}\,\overline{T}\,\overline{U} + \overline{R}\,\overline{S}\,\overline{T}\,U + \overline{R}\,S\,\overline{T}\,U + R\,\overline{S}\,\overline{T}\,U$
$+ \overline{R}\,\overline{S}\,T\,U + \overline{R}\,S\,T\,U + R\,S\,T\,U + R\,\overline{S}\,T\,U + \overline{R}\,\overline{S}\,T\,\overline{U} + R\,\overline{S}\,T\,\overline{U}$

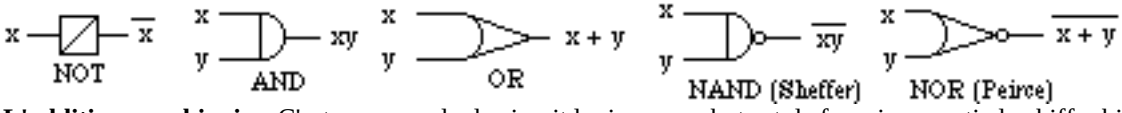
TU \ RS	00	01	11	10
00		1	1	1
01	1	1		1
11	1	1	1	1
10	1			1

En considérant, dans l'ordre, la dernière colonne (entourée), le carré entouré, la troisième ligne, les quatre cases doublement entourée, et enfin les deux cases restant dans la première ligne, on obtient :
$h(R , S , T , U) = R\,\overline{S} + \overline{R}\,U + T\,U + \overline{S}\,T + S\,\overline{T}\,\overline{U}$
Cette méthode a l'avantage de visualiser immédiatement de bonnes simplifications pour un petit nombre de variables, à partir de cinq, cela devient plus difficile.

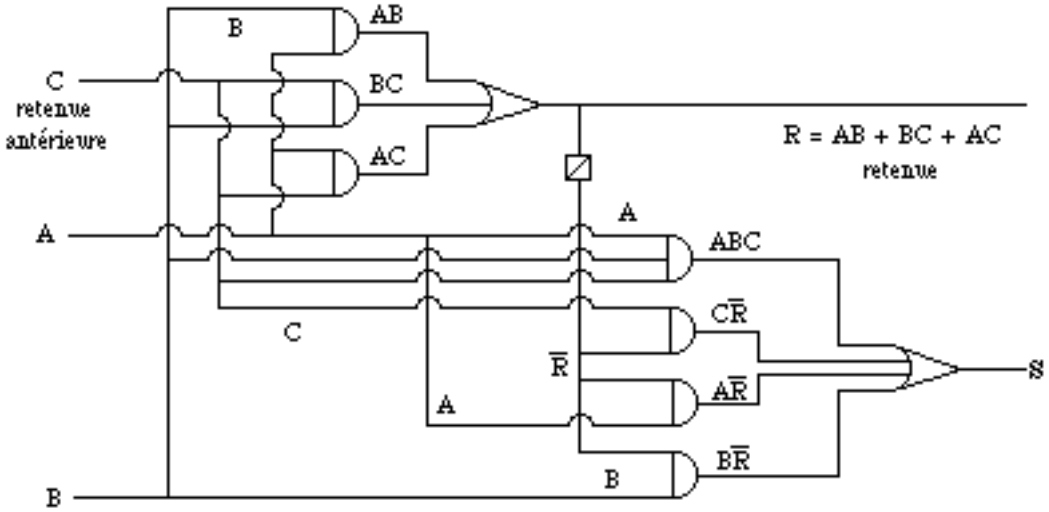
Les circuits logiques
Le problème de la simplification des expressions booléennes est essentiel en vue de la réalisation matérielle minimale des circuits. Nous convenons dans la suite qu'un fil représente une variable booléenne qui est vraie lorsqu'il est traversé par un courant électrique, et fausse dans le cas contraire. Par exemple le schéma suivant est un circuit permettant d'obtenir l'exclusion réciproque de deux variables booléennes. La sortie à droite est parcourue par un courant si et seulement si une seule des deux entrées à gauche est alimentée.



Ce circuit est réalisé grâce à l'utilisation d'un seul type de "porte logique", celle délivrant en sortie la négation de la conjonction de ses deux entrées. Les symboles utilisés pour les différentes portes logiques sont :

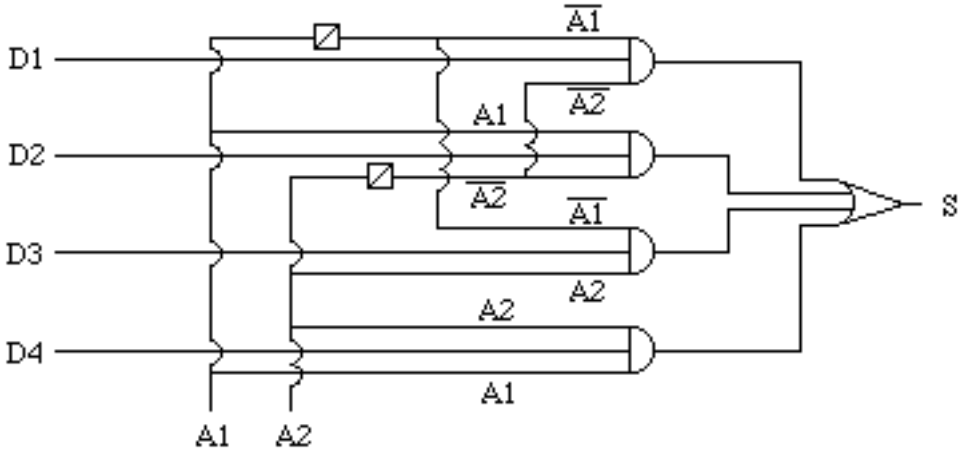


L'additionneur binaire. C'est un exemple de circuit logique, son but est de fournir en sortie le chiffre binaire (0 ou 1) S résultat et la retenue R de l'addition de deux chiffres binaires A et B avec une éventuelle retenue C.



Lorsqu'on additionne trois chiffres binaires A, B, C, le chiffre obtenu S est 1 si et seulement si parmi ces trois chiffres il y a trois 1 ou bien un 1 et deux 0. Pour qu'il y ait une retenue 1, il suffit qu'il y ait au moins deux 1 parmi les trois valeurs, soit $R = A\,B + A\,C + B\,C$
On a donc $S = A\,B\,C + \overline{A}\,\overline{B}\,C + \overline{A}\,B\,\overline{C} + A\,\overline{B}\,\overline{C} = A\,B\,C + \overline{R}\,(A + B + C)$
Cette dernière expression peut être vérifiée algébriquement, ou bien en faisant un tableau des huit possibilités d'entrées A, B, C et en effectuant l'addition binaire dans chaque cas.
Pour des régistres de n cases binaires, on utilise une cascade de n additionneurs dont le premier a une entrée retenue nulle, et dont le dernier indique une éventuelle erreur de dépassement si la dernière retenue est 1.

Deuxième exemple, le multiplexeur. C'est un circuit permettant de sélectionner une donnée D qui sera obtenue en sortie S, grâce à un "code d'adresse" A. Il faut n adresses binaires pour coder 2^n données, par exemple, pour les quatre possibilités de signaux reçus en A_1 et A_2, on pourra "aiguiller" en sortie l'une des quatre données D_1, D_2, D_3, D_4, c'est un multiplexeur à quatre voies. A cause des portes "AND", on voit très clairement que la sortie S prend la valeur de D_1 seulement dans le cas où $A_1 = A_2 = 0$, D_2 si $A_1 = 1$ et $A_2 = 0$ etc...



1° A un référendum étaient posés deux questions êtes-vous pour "l'élargissement du marché commun" et pour "la réforme du sénat" ? Quels sont tous les cas élémentaires traduisant un vote "non" ?

2° Donner l'interprétation de "Défense de fumer et défense de cracher." et "Défense de (fumer et cracher)."

3° Démontrer l'associativité des connecteurs $\wedge$ et $\vee$, puis les distribitivités mutuelles à l'aide des huit distributions possibles de "vrai" ou "faux" pour trois propositions.

4° Démontrer à l'aide de tables de vérité, que la conjonction $\wedge$ est distributive sur l'exclusion réciproque $\oplus$. Démontrer de même la distributivité de la disjonction sur l'équivalence.

5° Montrer que l'exclusion réciproque $\oplus$ est commutative, associative, involutive ($P \oplus P \Leftrightarrow 0$) et 0 est neutre.

6° Montrer la compatibilité de $\wedge$ avec $\Rightarrow$. C'est-à-dire : si $P \Rightarrow Q$ est vrai, alors: $P \wedge R \Rightarrow Q \wedge R$. Montrer que 1 est neutre à gauche de $\Rightarrow$ et absorbant à droite soit : $(1 \Rightarrow P) \Leftrightarrow P$ et $(P \Rightarrow 1) \Leftrightarrow 1$.

7° Démontrer par l'absurde l'impossibilité du voyage dans le passé, Monsieur X remonte au jour de sa naissance armé de mauvaises intentions envers lui-même et ... Continuez.

8° Montrer que $\neg P \Leftrightarrow P \mid P$ puis que : $(P \vee Q) \Leftrightarrow (P|P)|(Q|Q)$, $(P \wedge Q) \Leftrightarrow (P|Q)|(P|Q)$ et $(P \Rightarrow Q) \Leftrightarrow P|(Q|Q)$
Enfin : $(P \oplus Q) \Leftrightarrow [P \mid (Q|Q)] \mid [Q \mid (P|P)]$ et $(P \Leftrightarrow Q) \Leftrightarrow [(P \mid (Q|Q)) \mid (Q \mid (P|P))] \mid [(P \mid (Q|Q)) \mid (Q \mid (P|P))]$

9° Donnez la négation des phrases suivantes : Tous les Belges sont grands et blonds. Si un Suédois habite au nord du cercle arctique, alors il est brun. Si un Turc n'est pas natif de la mer noire, alors il est petit mais n'est pas blond. On trouve toujours chaussure à son pied. (Cette dernière phrase comporte trois quantificateurs.)

10° Les propositions suivantes sont-elles vraies ?

$\forall x \in R \quad x^2 > 0$	$\forall x \in N \quad x \neq 0 \Rightarrow 1/x \in N$
$\exists x \in N \quad x \neq 0$	$\exists x \in R \quad -x^2 + 16 = 0$
$\forall x \in N \, \exists y \in N \quad x < y$	$\exists x \in R \, \forall y \in R \quad xy = yx = y$

11° Démontrer que $\exists x \in R \quad x^2 \in N$ et $\exists x \in R \quad -3x^7 - 5x^4 + 4x^3 - 5x + 17 \geq 0$ sont vraies.

12° Démontrer que les propositions suivantes sont fausses. $\forall x \in R \quad x^2 - 1 < 0$ et $\forall x \in R \quad \exists y \in R \quad xy = 1$

13° Ecrire les négations des propositions suivantes et étudier leur valeur de vérité.

$\forall y \in R \exists x \in R \quad xy = y$
$\forall x \in R \exists y \in R \quad xy = 1$
$\forall x \in R \quad x \geq 0 \Rightarrow 1/x \geq 0$
$\forall x \in R \quad x^2 > 0 \Leftrightarrow x \neq 0$
$\forall x \in R \forall x' \in R \forall y \in R \quad xy = x'y \Rightarrow x = x'$
$\forall x \in R \forall y \in R \quad xy = yx$
$\forall x \in R \exists y \in R \quad y = 5x - 9$
$\forall x \in R \forall x' \in R \quad x^2 = x'^2 \Rightarrow x = x'$
$\forall x \in R \quad x = 0 \Rightarrow x(x + 1)(x - 1) = 0$
$\forall \varepsilon > 0 \exists \alpha > 0 \forall x \in R \quad |x| > \alpha \Rightarrow |1/x| < \varepsilon$

14° Démontrer que pour un ensemble quelconque E, l'ensemble P(E) de ses parties est un anneau pour les opérations Δ et $\wedge$. La première loi est commutative, associative, ø est un élément neutre, tout élément A est son propre opposé, la seconde loi est associative et distributive pour la première, quelle en est le neutre ?

15° Un guerrier captif a la grâce de poser une question à laquelle sera répondu vrai ou faux. Il est en présence de deux portes gardées derrière lesquelles se trouvent, la fille du roi d'une part, un horrible monstre d'autre part. Un des gardien ne ment jamais, l'autre ment systématiquement. Soient a et b les propsitions "tu dis la vérité" et "c'est la porte conduisant à la fille du roi". Trouver l'expression booléenne fonction de a et de b, correspondant à la question qu'il faudra poser de façon à ce que la réponse soit équivalente à b.

16° Montrer que : $\overline{x\bar{y}} + \overline{\bar{x}y} = xy + \bar{x}\,\bar{y}$ et $x\bar{y} + xy + \bar{x}y + \bar{x}\,\bar{y} = 1$

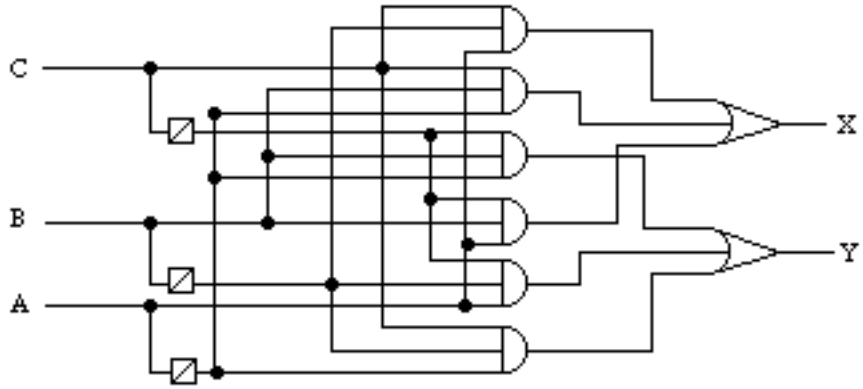
17° Simplifier les expressions suivantes grâce aux tables de Karnaugh : $\bar{A}\,\bar{B}\,C + \bar{C}$,
$A\,B\,\bar{C} + A\,\bar{B}\,\bar{C}$, $A\,B\,\bar{C}\,D + A\,B\,\bar{C}\,D + A\,B\,C\,D$,
$\bar{A}\,B\,C\,\bar{D} + \bar{A}\,B\,\bar{C}\,D + A\,B\,\bar{C}\,\bar{D} + A\,B\,C\,\bar{D} + A\,B\,C\,D$
$\bar{A}\,\bar{B}\,C\,\bar{D} + \bar{A}\,\bar{B}\,C\,D + A\,B\,\bar{C}\,\bar{D} + A\,B\,C\,\bar{D} + A\,\bar{B}\,\bar{C}\,D$
$\bar{A}\,\bar{B}\,C\,D + A\,\bar{B}\,\bar{C}\,D + \bar{A}\,B\,\bar{C}\,D + \bar{A}\,B\,C\,\bar{D} + A\,\bar{B}\,C\,\bar{D} + \bar{A}\,\bar{B}\,C\,D$
$\bar{A}\,B\,\bar{C} + A\,\bar{B}\,\bar{D} + \bar{A}\,B\,\bar{C}\,D + A\,\bar{B}\,\bar{C}\,D + A\,B\,\bar{C}\,\bar{D} + A\,\bar{B}\,C\,\bar{D} + A\,B\,C\,\bar{D} + \bar{A}\,B\,\bar{C}\,D$
$A\,B\,\bar{C}\,\bar{D} + \bar{A}\,B\,C\,\bar{D} + A\,B\,C\,D + \bar{A}\,B\,C\,D + A\,\bar{B}\,C\,D + \bar{A}\,\bar{B}\,C\,D + \bar{A}\,B\,C\,\bar{D} + \bar{A}\,\bar{B}\,C\,D$

18° Résoudre le système booléen des 4 équations $\neg A.\neg D\,(B + \neg C) = 0$, $A.B + \neg C = 1$, $\neg D.A = 0$, $C + A.D = 1$.

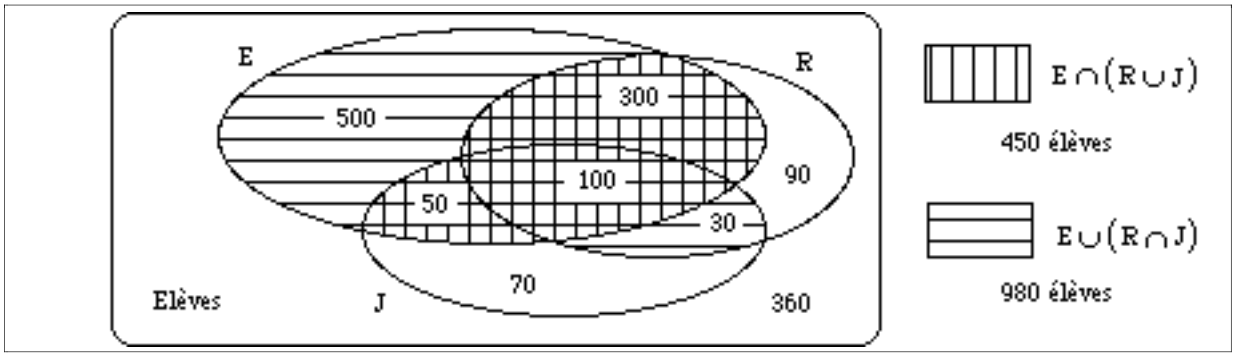
19° Construire un circuit à trois entrées A, B, C, fournissant en sortie l'exclusion réciproque $A \oplus B \oplus C$.

20° Construire le circuit réalisant un multiplexeur à huit voies de données et trois adresses binaires.

21° Calculer X et Y en fonction de A, B, C. Que fait ce circuit ?



22° Dans une école comprenant 1500 élèves, 950 sont inscrits aux cours d'espagnol (ensemble E), 520 en russe (ensemble R), et 250 en japonais (ensemble J). Mais on précise que 400 font de l'espagnol et du russe, 150 font de l'espagnol et du japonais, 130 font du russe et du japonais, et enfin 100 élèves font les trois langues.
Combien d'élèves font-ils uniquement de l'espagnol ?, du russe ?, du japonais ?, ne font pas de langue étrangère?
Combien y a-t-il d'élèves dans les ensembles $E \cap (R \cup J)$ et dans $E \cup (R \cap J)$?
La solution réside dans (schéma) une "partition" en 8 parties élémentaires deux à deux disjointes. On indique alors dans chacune des huit parties, par de simples soustractions, leurs cardinaux.


E
R
500
300
100
90
50
30
70
Elèves
J
360
E ∩ (R ∪ J)
450 élèves
E ∪ (R ∩ J)
980 élèves

ANNEXE 2 LANGAGE FORTH

Le FORTH est un langage très original qui mérite que l'on étudie au moins son principe. C'est plutôt un langage fonctionnel, mais qui utilise une pile de travail ce qui signifie qu'un programme va toujours chercher ses données au sommet de cette pile, va la transformer, et y déposer ses résultats. Ce langage a été créé en 1970 par C.Moore, le noyau en est assez restreint, le reste pouvant être redéfini en Forth (tout comme en Lisp où le langage est défini à partir de quelques fonctions primitives). Ici un programme est une suite de mots à qui on donne un nom, soit un nouveau mot.

Structure de pile : une pile est une structure de données où le dernier rentré est le premier sorti ("last in, first out" ou "lifo"), ainsi, si en rentrant au clavier les informations a, b, c avec a en premier et c en dernier, on représentera la pile par : ... a b c le "sommet" de la pile étant à droite contient ici la valeur c, et la partie gauche (les points de suspension) représentant ce qui a pu être entré antérieurement à c.

Notation polonaise suffixée (chapitre 9) c'est la notation utilisée par le Forth, l'espace étant le séparateur, ce qui veut dire que l'expression (ab + cd) / e s'écrira par : a b * c d * + e /

Commentaires, ils doivent être encadrés par des parenthèses et espaces.

Les mots prédéfinis manipulant la pile

Il y a d'abord les fonctions arithmétique, par exemple l'addition + va modifier la pile de telle sorte que si on entre a b +, la pile se représente donc par : ... a b + et le fait d'exercer la touche "return" transformera la pile en : ... a+b

On indiquera schématiquement ... a b + ☞ ... a+b

Il existe de même les opérations *, -, / (la division entière) et "mod" qui est l'opération "modulo"

par exemple ... 23 6 mod ☞ ... 5

Remarque, l'interpréteur Forth ne fait que cette modification de pile en répondant OK, il n'affiche pas de résultat, pour cela on utilisera :

. (le point) : c'est un mot qui permet l'affichage du sommet, mais en le vidant ... a b ☞ ... a

. . permet l'affichage en les retirant, du sommet et du sous-sommet", ainsi il existe un mot "/mod" permettant d'obtenir à la fois le quotient et le reste d'une division entière, on aura donc :

... 27 4 /mod . . affichera 3 6

.S affichage (sans modification) de toute la pile (à utiliser pour contrôler tous les premiers exercices)

. " message " permet l'affichage d'un message

DROP suppression du sommet de pile ... a b c ☞ ... a b

DUP duplication du sommet ... a b c ☞ ... a b c c

SWAP permutation des deux éléments au sommet ... a b c ☞ ... a c b

ROT rotation sur trois éléments ... a b c ☞ ... b c a

OVER recopie du second par dessus le sommet ... a b c ☞ ... a b c b

n PICK permet une recopie du n-ième (à partir de 0) au sommet ainsi "over" est équivalent à "1 pick" et par exemple "3 pick" effectuera ... a b c d ☞ ... a b c d a

n ROLL effectue une rotation sur la sous pile des n éléments du sommet, ainsi ROT équivaut à "3 roll" et "4 roll" effectuera ... a b c d e f ☞ ... a c d e f b

CR simple effet de bord permettant le passage à la ligne

Programme = définitions de nouveaux mots
On définit un programme par "deux points" suivi du nom choisi, puis de la suite des opérations à effectuer et achevé par "point virgule", un bon programme doit prendre ses données sur la pile, en les retirant, et y déposer ses résultats. Par exemple pour définir le carré d'un nombre, le programme sera : : carre dup * ;
L'utilisation s'en fera au même titre que les autres mots, par exemple :

6 carre . répondera 36 ou vu sur la pile : carre 6 ... ☞ 36 ...

Le programme : moyenne + 2 / ; pourra s'utiliser par la suite des quatre informations 7 5 moyenne . qui affichera 6.

: cube dup dup * * ;

Le mot "over" peut lui-même être défini à partir des autres par :

: over swap dup rot swap ;

Prédicats
On dispose des relations =, <, <=, >=, 0=, 0>, etc, la réponse à un prédicat est 0 (faux) ou -1 c'est à dire 65535 (vrai). Les constantes TRUE et FALSE existent cependant.
... 4 5 = . affiche 0

... 4 5 1 - = ☞ ... 1

... 3 4 < ☞ ... 1

Sélection (le "if" vide le sommet c'est à dire la valeur booléenne du test qui précède)
condition IF action-si-vrai ELSE action-si-faux THEN
exempe : abs (définit la valeur absolue) dup 0< if -1 * then ;
Itérations : 3 formes existent, la syntaxe de la première est :
valmax valmin DO traitement LOOP réalise un traitement de 1 en 1
Exemple : suite (affiche les entiers de 0 à 10) 0 10 0 do dup . 1 + loop drop ;
mais aussi : suite 1 10 1 do dup dup . 1 + loop drop ;
ou encore : suite 10 0 do I . loop ; sachant que I est le nom du compteur en Forth

BEGIN action test UNTIL est une autre structure

Exemple de la suite de Syracuse (chapitre 2)

Séquences de mots	Vue de la pile	Commentaires
: syracuse	... u	
begin dup	... u u	la valeur courante u est recopiée
2 mod	... u r	r est le reste de la division de u par 2
0= if	... u	la valeur de vérité de "r = 0" est vidée par "if"
2 / else	... u/2	dans les deux cas,
3 * 1 + then	... 3u+1	le suivant de u se trouve au sommet
dup cr . 1 =	... u f	u est le terme suivant, f est le booléen "u=1"
1 = until ;	... u	

BEGIN action test WHILE action REPEAT permet un branchement arrière inconditionnel

Récursivité elle se fait grâce au mot RECURSE signifiant que le mot en cours de définition doit s'exercer sur le sommet de la pile (péalablement emplit des bons paramètres)

Exemple : factorielle dup 1 > if dup 1 - recurse * else drop 1 then ;

Affectation valeur CONSTANT nom (le Forth n'est pas vraiment fait pour cela)

Editeur du Turbo-Forth sur PC
On accède à l'éditeur par la touche F2 et la compilation d'un fichier par F3
On repasse de l'éditeur au Forth grâce à ^KD puis au système par le mot "bye".

ANNEXE 3 AIDE-MEMOIRE MSDOS

Codes ASCII

...		
13	D	return
...		
27	1B	escape
...		
32	20	espace
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
...	...	
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?
64	40	@
65	41	A
...	...	
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_
96	60	‘
97	61	a
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	

Alphabet grec

α	A	alpha
β	B	beta
γ	Γ	gamma
δ	Δ	delta
ε	E	epsilon
η	H	eta
ζ	Z	zéta
θ	Θ	théta
ι	I	iota
κ	K	kappa
λ	Λ	lambda
μ	M	mu
ν	N	nu
ξ	Ξ	ksi
o	O	omicron
π	Π	pi
ρ	P	rô
σ	Σ	sigma
τ	T	tau
υ	Y	upsilon
ϕ	Φ	phi
χ	X	khi
ψ	Ψ	psi
ω	Ω	oméga

Une lettre hébreue à connaître:

א aleph

B: permet de rendre actif le second lecteur

A: pour revenir au premier

C: pour le disque dur (ou D: ou E:)

DIR (directory) donne la liste des fichiers
(on peut jouer des masques *.*)

TYPE affiche un fichier ascii
donné par son nom.son ext

DEL effaçe un fichier
(ou encore ERASE)

RENAME nom1.ext1 nom2.ext2
change le nom d'un fichier

COPY nom.ext B: (prn) vers B ou imprimante

DISKCOPY A: B: copie complète

FORMAT B: formattage du disque en B

CHKDSK (checking) donne capacités, et
place restante.

CD nom (change directory) permet dans une hiérarchie d'accéder à un sous-répertoire.

CD .. remonte au répertoire-père

MD nom (make directory) crée un nouveau répertoire.

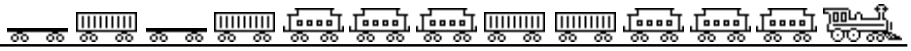
RD nom (remove directory) supprime un répertoire déjà vidé.

Dans un répertoire arborescent un fichier peut être appellé par \père\fils\....\nom.ext

Création d'un fichier de commandes
COPY CON: nom.BAT
liste de commandes
^Z (^ désigne la touche ctrl)
AUTOEXEC.BAT est le nom réservé à un fichier fixant le démarrage.

A propos du clavier :

SHIFT permet d'obtenir l'autre sens des touches
ALT n°ascii pour les caractères absents du clavier
ALT CTRL permet d'obtenir les symboles @ ^ # [ ] \
INS insertion (touche bascule)
DEL suppression
^A ^E début et fin de ligne ^L début d'écran
← (back-space ou ^D) en haut du clavier, efface le dernier caractère tapé.
^P écho sur l'imprimante
^C arrêt d'une exécution ^S arrêt d'un affichage ^Q reprise
^ALT DEL redémarrage
PRTSCR (print-screen) copie d'écran sur imprimante.
← → ↓ ↑ déplacement du curseur dans la page.
PGDN et PGUP déplacent d'une page en bas ou en haut



ANNEXE 4 AIDE MEMOIRE UNIX

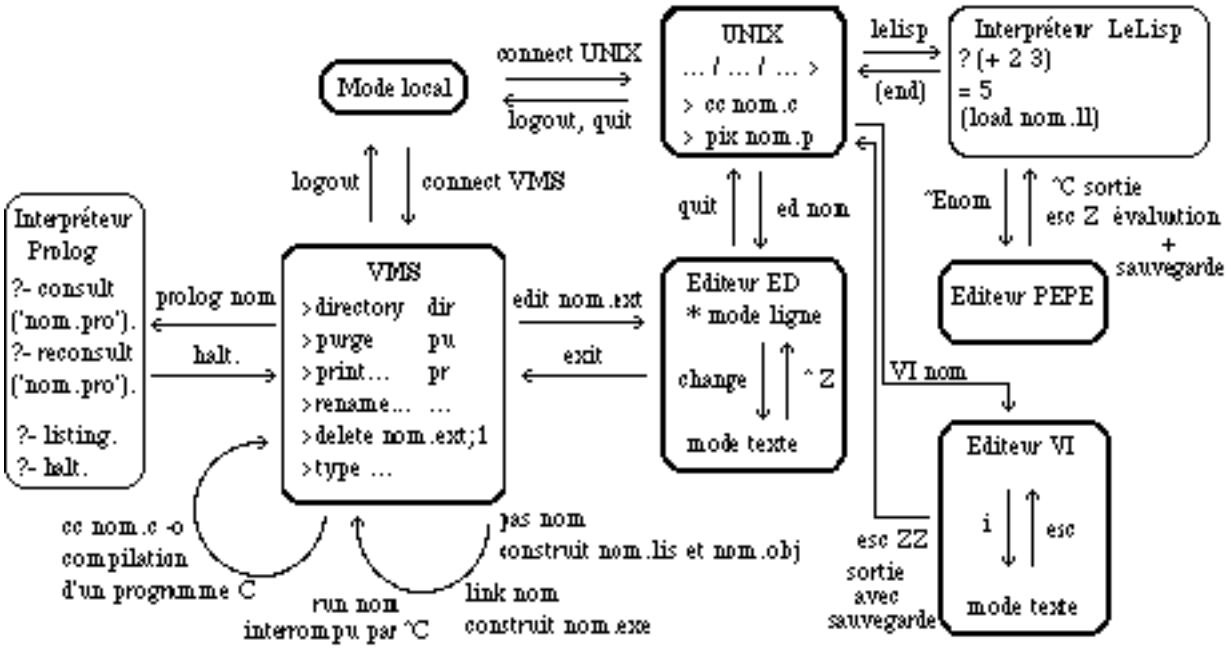
Rappelons qu'un système d'exploitation est un ensemble de programmes écrits en assembleur (mais presque tout en C pour UNIX) permettant l'utilisation d'un ordinateur, par opposition aux programmes dits d'application, venant par dessus. A première vue l'essentiel d'un système d'exploitation est un ensemble de commandes qui seront analysés par un interpréteur (le command.com du MSDOS, le SHELL d'UNIX).

Lancement du système :
local > connect *nom de machine*
On a aussi en mode local quelques commandes : help, show us, show session, show services, resume, backward, disconnect all, ...
login : *votre nom d'utilisateur*
password : *votre mot de passe*, on est alors placé dans un répertoire personnel

logout, quit, disconnect, bye	sortie du système
passwd	pour créér un nouveau mot de passe
set prompt " ... "	pour créér un autre prompt
who	liste des usagers actifs
tty	donne le chemin de connexion
ls	donne le catalogue d'un répertoire
cat *nom de fichier*	visualisation du fichier
cat *nom de fichier* > /dev/lp	impression du fichier (sur certains systèmes lp, lpr ou print)
lpr *nomimprimante nomfichier*	impression (il y a une imprimante par défaut)
cp *nom autrenom*	réalise une copie du premier dans un autre répertoire éventuellement sous un autre nom
mv *anciennom nouveaunom*	changement de nom
ln *anciennom nouveaunom*	réalise une duplication sous un autre nom sans effacement
rm *nom de fichier*	suppression
mkdir *nom*	création d'un nouveau répertoire
rmdir *nom*	destruction d'un répertoire
mail	accès au courrier électronique
ftp	accès au système de transfert de fichiers

↑ Revenir à la commande antérieure

↓ Passer à la commande suivante



BIBLIOGRAPHIE

Aarts E.H.L. Korst J.H.M *Simulated annealing and Boltzmann machines*, Wiley Chichester, 1989

Arsac J. *Quelques sujets de réflexion à propos de la création de programmes*, Séminaire d'informatique théorique 82-83 LITP (Universités Paris VI et Paris VII)

Axelrod R. *The genetic algorithms for the prisoner dilemma problem*, Genetic algorithms and simulated annealing Morgan Kaufmann Pub. 1987

Baldwin J.F. *Evidential support logic programming*, Fuzzy sets and systems, n°24, 1987

Baldwin J.F. Martin T. Pilsworth B. *Fril, fuzzy and evidential reasoning in artificial intelligence*, Research studies press, 1994

Belouze B. Glaymann M. Haug P. Herz J. *Les carrés magiques*, Publication de l'APMEP, 1975

Bonabeau E. Theraulaz G. *Intelligence collective*, Hermès, 1994

Bouchon B. *Logique floue*, Que sais-je ? PUF, 1993

Boussard J.C. Mahl R. *Programmation avancée*, Eyrolles, 1984

Bouteloup J. *Carrés magiques, carrés latins et eulériens*, Ed. du choix, 1991

Breaud-Pouliquin N. *Lisp sur Apple II*, Editions PSI, 1982

Chi R.S.Y. *Buddhist formal logic*, Motival Banarsidass, Delhi 1984

Colmerauer A. Giannesini F. Kanoui H. Pasero R. Van Caneghem M. *Prolog*, Inter Editions, 1985

Davalo Naim *Réseaux de neurones*, Eyrolles 1985

De Kleer J. *An assumption-based truth maintenance system*, Artificial intelligence n°28, 1986

Delahaye *Dessins géométriques et artistiques*, Eyrolles, 1985

Delahaye J.P. *L'altruisme récompensé*, Pour la science, novembre 1992

Dony R. *Graphisme scientifique*, Masson, 1985

Doyle J. *A truth maintenance system*, Artificial intelligence n°12, 1979

Dubois D. Prade H. *Théorie des possibilités*, Masson, 1985

Farreny H. *Lisp*, Masson, 1984

Froidevaux C. Gandel M.C. Soria M. *Types de données et algorithmes*, Mc Graw Hill, 1990

Ganascia J.G. *Deux techniques d'arrentissage symbolique appliquées à la construction de bases de connaissances*, Thèse à l'Université d'Orsay 1987

Gacôgne L. *Les suffixes du français et l'analyse automatique de leurs enchainements*. Mémoires d'informatique fondamentale Paris VII-CNAM 1986

Gacôgne L. *Programmation*, Eyrolles 1988

Gacôgne L. *Applications de l'analyse*, Eyrolles 1990

Glover J.D. *Seminar in stochastic processes*, 1987

Goldberg D. E. *Genetic algorithms*, Addison Wesley 1989

Hacques G. *Mathématiques pour l'informatique*, Armand Colin 1971

Heudin J.C. *La vie artificielle*, Hermès 1994

Hofstadter D. *Gödel Escher et Bach* Inter-éditions 1985

Hopfield J.J. *Neural networks and physical systems with emergent computational abilities*, PNAS USA n°79 p.2554-2558, 1982

Ifrah G. *Histoire universelle des chiffres*, Robert Laffont, 1994

Jodouin J.F. *Les réseaux neuro-mimétiques*, Hermès, 1994

Kamada H. Yoshida *Visual control system using image processing and fuzzy system*, Tokyo 1990

Kantor J.M. *Mathématiques venues d'ailleurs*, Belin 1982

Kirkpatrick S. Gellat C.D. Vecchi M.P. *Optimization by simulated annealing*, Science n° 220 p.671-680 1983

Knuth D. *The art of computer programming*, Addison Wesley, 1973

Kohonen T. *Self organization and associative memory*, Springer Verlag 1989

Koza J. *Genetic programming*, MIT Press (Cambridge) 1992

Kurbel K. *Improving sequencing algorithms based on Hopfield nets by more efficient representation*, Congrès EUFIT 1994

Laurière J.L. *Intelligence artificielle*, tomes 1 et 2, Eyrolles 1988

Ledgard H.F. *Proverbes de programmations*, Dunod, 1979

Le Cun Y. *Modèles connexionistes de l'apprentissage*, Thèse Université Paris VI, 1987
Le Lionnais F. *Les nombres remarquables*, Hermann, 1983
Lippman S. *L'essentiel du C++*, Addisson Wesley, 1992
Livercy C. *Théorie des programmes*, Dunod, 1978
Mamdani E.H. *Application of fuzzy algorithms for contol of simple dynamic plant*, Proc.IEEE n°12, 1974
Mandelbrot, *Les objets fractals*, Flammarion, 1975
Mathieu P. *L'utilisation de la logique trivaluée dans les systèmes experts*, Thèse de l'Université des sciences et techniques de Lille, 1991
Meyer B. Baudoin C. *Méthodes de programmation*, Eyrolles 1984
Michalewicz Z. *Genetic algorithms + data structures = evolution programs*, Springer Verlag 1992
Miclet L. *Méthodes structurelles pour la reconnaissance des formes*, Eyrolles, 1984
Minoux M. *Programmation mathématique*, Dunod, 1983
Nehlig P. *Applications affines discrètes et antialiassage*, Thèse de l'Université L.Pasteur Strasbourg, 1992
Nougier J.P. *Méthodes de calcul numérique*, Masson, 1987
Platt J. *A resource allocating network for function interpolation*, Neural Computation vol.3, p.213-225, 1991
Poggio T. Girosi F. *Networks for approximation and learning*, Proceedings of IEEE p.1481-1497, 1990
Pomerol J.C. *Les systèmes experts*, Hermès, 1988
Queinnec C. *Les langages Lisp*, InterEditions, 1994
Quinlan J.R. *Induction of decision trees*, Machine Learning, p.81-106, 1986
Reeves C.R. *Modern heuristics techniques for combinatorial problems*, Blackwell scientific publications p.20-69, 1993
Reveilles J.P. *Geométrie discrète, calcul en nombres entiers et algorithmique*, Thèse d'état de l'Université L.Pasteur de Strasbourg 1991
Roy J.P. Kiremitdjian G. *Lire Lisp*, Cedic Nathan
Roy J.P. *Lisp et Prolog en terminale*, publ. de l'IREM Paris sud, 1984
Rumelhart D.E. Hinton G.E. Williams R.J. *Learning internal representations by error propagation*. Parallel distributed processing MIT Press p. 318-362, 1986
Stroustrup B. Ellis A. *The annotated C++ reference manual*, Addison Wesley, 1992
Van Caneghem M. Warren D. *Logic programming and its applications*, Ablex pub. Corporation, 1986
Wertz H. *Intelligence artificielle*, Masson, 1984
Wilf H.S. *Algorithmes et complexité*, Masson, 1989
Wirth N. *Introduction à la programmation systématique*, Masson, 1981