LOGIQUE PROPOSITIONNELLE

Vade-mecum de l’étudiant de Licence 1

Nicolas Clerbout.

Lille 3

Résumé : Le but de ce fascicule est de fournir aux étudiants de licence 1 un support pour le cours de logique. Il est donc essentiellement consacré à la logique propositionnelle. Il contient cependant un certain nombre de concepts fondamentaux pour tout étudiant en logique. La maîtrise de ces notions est par ailleurs essentielle à tout travail sérieux en philosophie contemporaine du langage et en épistémologie.

Note : Ce texte a été rédigé pour fournir aux étudiants un outil de travail et de révision. Il ne saurait remplacer le cours même.

Table

1 LOGIQUE ET SIGNIFICATION

1.1 Validité d'un argument ; schéma d'argument

Un argument peut être considéré comme une suite de phrases, dont les premières sont appelées prémisses et dont la dernière est la conclusion. En général, on considère que les prémisses d'un argument fournissent une *raison* de croire à la vérité de sa conclusion. En ce sens, argumenter c'est transférer la vérité des prémisses vers la conclusion.

Une classe particulière d'arguments intéresse le logicien : les arguments *valides*. Un argument est valide ssi (si, et seulement si) il n'est pas possible que sa conclusion soit fausse alors que ses prémisses sont vraies. En d'autres termes : si les prémisses d'un argument valide sont vraies, alors sa conclusion doit être vraie également. Les arguments valides sont ceux qui *garantissent la préservation de la vérité*.

On dit que la conclusion d'un argument valide est une c*onséquence logique* de ses prémisses.

Il est important de noter que cette *définition est indifférente au fait que les prémisses soient, en réalité, vraies ou fausses*. Voici deux exemples d'arguments. Le premier est valide alors que ses prémisses et la conclusion sont fausses. Le deuxième n'est pas valide alors que les prémisses et la conclusion sont vraies.

Ex 1 :

Tous les joueurs professionnels de tennis sont des femmes.
<u>Cristiano Ronaldo est un joueur professionnel de tennis.</u>
Cristiano Ronaldo est une femme.

Ce qui est important, c'est que si on accepte les prémisses (si on croit à leur vérité), on *doit* aussi accepter la conclusion de l'argument. Le premier exemple montre donc que la vérité de ses prémisses n'est pas *nécessaire* à la validité d'un argument.

Ex. 2 :

Tous les hommes sont des mammifères.
<u>Quelques mammifères sont bipèdes.</u>
Tous les hommes sont bipèdes.

Dans l'ex. 2, la vérité des prémisses n'est pas *suffisante* pour que l'argument soit valide : accepter les prémisses n'oblige pas à accepter la conclusion (même si ici, la conclusion est vraie). On pourrait *imaginer* une situation où les prémisses de l'ex. 2 sont vraies mais la conclusion fausse (il suffirait que quelques hommes ne soient pas bipèdes).

La vérité ou la fausseté des prémisses n'affecte donc pas la validité d'un argument. Puisque ce n'est pas la vérité des prémisses qui importe, qu'est-ce qui détermine la validité d'un argument ?

L'argument suivant est valide :

Ex. 3 :

Camille gagne la partie d'échecs ou Benjamin gagne la partie d'échecs.
Benjamin ne gagne pas la partie d'échecs.
Camille gagne la partie d'échecs.

On n'a même pas besoin de savoir si Camille et Benjamin jouent ou non aux échecs. La validité de l'argument n'a en fait rien avoir avec Camille et Benjamin. Car on peut remplacer leurs noms par celui d'autres personnes, ainsi que remplacer « gagner la partie d'échec » par autre chose, et conserver la validité de l'argument :

Ex. 4 :

Jean paye l'addition ou Antoine paye l'addition.
Antoine ne paye pas l'addition.
Jean paye l'addition.

Il paraît évident que la validité de l'argument ne dépend finalement que du fait qu'il consiste en la disjonction de deux phrases dans la première prémisse, que la deuxième de ces phrases est niée dans la deuxième prémisse et enfin que la conclusion est la première des phrases. C'est en vertu de cette *forme* particulière que ces deux arguments sont valides. On peut représenter schématiquement cette forme par :

A ou B.
Non B.
A.

On appelle une telle schématisation un *schéma d'argument*. Les lettres A et B peuvent être remplacées par n'importe quelle phrase, on obtiendra un argument valide.

Par contre, on ne peut remplacer « ou » et « non » par n'importe quelle expression et être sûr de sauvegarder la validité du schéma d'argument.

Ce ne sont donc que certains types d'expressions qui sont importants quand on veut déterminer la validité d'un argument. Par exemple, la signification de « ou » tient une part importante dans la validité du schéma ci-dessus. En explorant la validité des schémas d'argument où cette expression apparaît, on explore du même coup la signification de ce mot. La même chose vaut évidemment pour toutes les expressions qui peuvent jouer un rôle déterminant dans la validité de schémas d'argument.

Pour être plus précis : la *signification* des phrases A et B n'est pas déterminante pour la validité de l'argument[1], mais celles de « A ou B » et de « Non A » le sont. Or il se trouve que pour connaître la signification de ces phrases, on s'appuie sur la signification de la disjonction pour la première et sur la signification de la négation pour la deuxième.

Dans le cadre qui nous intéresse ici, les expressions dont la signification est importante pour la validité d'argument sont certaines conjonctions grammaticales (ou, et, si ...alors) et la négation.

1.2 Constantes logiques

Dans les schémas d'arguments tels qu'on les a présentés, il y a un petit nombre de mots dont la signification est essentielle à la validité de l'argument. Ces termes sont les *constantes logiques*. Lorsqu'un argument est valide, on peut altérer la signification de tous les termes qui le composent, si on ne touche pas aux constantes logiques (et que l'on respecte la substitution uniforme), alors le nouvel argument est encore valide.

En logique propositionnelle, on considère généralement un ensemble de quatre constantes logiques, correspondant aux fonctions grammaticales de conjonction, disjonction, implication et négation.

1.3 Extensionalité et compositionnalité

Les arguments valides décrivent la forme des jugements *analytiques*, c'est-à-dire des jugements dont la vérité ne dépend *que de la signification des termes*. Si je juge que les prémisses sont vraies (jugement dont la forme

[1] Ce qui importe, en revanche, c'est l'*identité* de ces phrases : pour transformer un schéma d'argument en argument valide en langue naturelle, il faut remplacer *toutes les occurrences* d'une lettre donnée par *la même phrase*. On parle alors de *substitution uniforme*. La seule chose que les lettres A,B,C… conservent des phrases auxquelles elles se substituent, c'est justement la marque de leur identité.

est quelconque), alors le jugement par lequel je reconnais la vérité de la conclusion est analytique. La détermination de la validité d’un argument est donc un pur processus d’analyse de sa signification. Cette signification est structurée en vertu d’un principe généralement attribué à Frege, appelé le *principe de compositionnalité de la signification*. Selon ce principe, la signification d'une expression composée est entièrement déterminée par la signification des expressions plus simples qui la composent. Comme on l’a vu, les expressions les plus simples d'un langage dont la signification est cruciale pour déterminer la validité des arguments sont les *constantes logiques*. Ce sont elles qui permettent de construire des expressions composées à partir d’expressions simples. C’est pour cette raison qu’on les nomme aussi *opérateurs* : ils *transforment* une ou des expressions simples en une expression plus complexe. La *signification* d’une constante logique n’est rien d’autre que le rôle qu'elle joue dans la validité d'un argument.

Le problème qui se pose alors est celui de la signification des expressions simples à partir desquelles on construit les expressions complexes. L’unité fondamentale de signification à partir de laquelle on peut construire un argument est la *proposition*. Les énoncés qui composent un argument *ont pour contenu*[2] (ils dénotent) *des propositions*. Une proposition est l’entité linguistique minimale capable d’être vraie ou fausse. Une proposition *décrit* un certain état de fait, un certain état possible du monde. Frege cependant a introduit la distinction entre le *sens* de la proposition, qui est cette description, et sa référence ou *dénotation*, qu’on peut comprendre grossièrement comme la réponse à la question suivante : le monde est-il tel que la proposition le dit ? Si c’est le cas, alors la proposition dénote le vrai, sinon elle dénote le faux. Le vrai et le faux sont donc des entités extralinguistiques (un peu à la manière des formes idéales platoniciennes).

La notion de signification, entendue comme *sens*, est certainement très complexe. Mais l’on doit à Frege un second principe permettant de simplifier considérablement le problème de la signification : le principe *d’extensionalité*. L’extension d’un concept, c’est ce qu’il dénote, c'est-à-dire

---

[2] Il faut distinguer avec précision trois concepts : l'énoncé, la phrase, la proposition.
L'énoncé est un acte de langage. Si plusieurs locuteurs disent « Tous les hommes sont des mammifères », ils produisent différents énoncés. Les énoncés sont donc des événements singuliers.
Lorsque l’on veut désigner un énoncé, on lui donne un nom qui décrit le type d’acte en quoi cet énoncé consiste. La description de ce type général et abstrait est une phrase. Ainsi on dira que plusieurs locuteurs utilisent la même phrase « Tous les hommes sont des mammifères ».
Enfin il reste le concept délicat de proposition. On peut définir une proposition comme étant le contenu de signification d’une phrase. Ainsi, les phrases « Le chat mange la souris », « La souris est mangée par le chat » et « The cat eats the mouse » sont différentes, mais dénotent la même proposition parce qu'elles ont le même contenu de signification.

l'ensemble de tout ce qui peut être subsumé sous lui. L'idée centrale du principe d'extensionalité est que, pour ce qui concerne la validité logique, on peut réduire la signification d'une proposition à son extension : sa valeur de vérité. L'analyse logique de la signification d'une proposition complexe devient une analyse des rapports déterminés par les constantes logiques entre les valeurs de vérité des propositions simples qui la composent. Par exemple, la référence de la proposition[3] dénotée par l'expression « la porte est ouverte et il fait beau » ne dépend de rien d'autre que de la valeur de vérité de « la porte est ouverte » et de « il fait beau ».

2 SYNTAXE DE LA LOGIQUE PROPOSITIONNELLE

L'objet de la syntaxe est de définir la liste des signes qui composent le langage et les règles de leur usage correct.

2.1 Syntaxe d'un langage pour la logique propositionnelle

Un langage $\mathcal{L}$[4] pour la logique propositionnelle a comme symboles primitifs :

- un ensemble dénombrable de *variables propositionnelles*,[5] éventuellement indexées : $\{p, q, r, p_1, ..., p_n, q_1, ...\}$
- les *connecteurs* : $\wedge$, $\vee$, $\rightarrow$, $\neg$ (correspondant respectivement à la conjonction (et), la disjonction (ou), le conditionnel (si… alors) et la négation (non). $\neg$ est un connecteur à une place (unaire) et $\wedge$, $\vee$ et $\rightarrow$ des connecteurs à deux places (binaires)).
- les *parenthèses* : (et).

A partir de ce vocabulaire de base, on peut former des *expressions*, c'est-à-dire des chaînes de symboles primitifs du langage

Ex. 5

- p
- $\neg(\neg p)$

---

[3] C'est-à-dire sa valeur de vérité.

[4] On appelle $\mathcal{L}$ le langage-objet.

[5] Les variables propositionnelles ont pour rôle de dénoter les propositions logiquement simples, c'est-à-dire dépourvue de constantes logiques, telles que « la marquise sortit à cinq heures » ou « le bon sens est la chose du monde la mieux partagée ».

- $\neg)\vee pq\rightarrow(\wedge$

Mais toutes les expressions ne nous intéressent pas : seules celles qui sont capables de dénoter une proposition, c'est-à-dire dont la structure est identique à celle d'une proposition, nous intéressent. Ces expressions « correctes » sont désignées sous le terme *expressions bien formées* (*e.b.f.*) ou *formules*.

Définition 1 : *e.b.f.*

La définition d'une *e.b.f.* dans un langage $\mathcal{L}$ quelconque pour la logique propositionnelle est donnée récursivement par les conditions suivantes :

(i) Les variables propositionnelles du vocabulaire de $\mathcal{L}$ sont des *e.b.f.*

(ii) Si A est une *e.b.f.*, alors $\neg$A est une *e.b.f.*

(iii) Si A et B sont des *e.b.f.*, alors $(A\wedge B)$, $(A\vee B)$ et $(A\rightarrow B)$ sont des *e.b.f.*

(iv) Rien d'autre n'est une *e.b.f.* .

Dans cette définition, on utilise les lettres capitales du début de l'alphabet (A,B,C…) comme des variables désignant une *expression* quelconque. Mais en général, ces capitales désignent des *formules* quelconques (c'est-à-dire des *e.b.f.*).

(ii) et (iii) sont les clauses inductives de la définition : partant d'un cas connu (A,B) elle permettent de conclure à un nouveau cas. Une clause telle que (iv) est parfois appelée la clause de clôture d'une définition : elle permet de clore l'ensemble des expressions bien formées : une expression ne peut être une formule qu'en vertu de l'une des clauses (i) à (iii).

Exercice 1 :

Les expressions suivantes sont-elles bien formées ? Pourquoi ?

$((p\rightarrow q)\vee(q\rightarrow\neg\neg p))$

$((p)\vee(q\wedge r))$

$(p_1\rightarrow((p_2\rightarrow q)))$

$(\neg(p\vee q)\rightarrow\neg\neg q)\rightarrow r$

2.2 Autres notions syntaxiques

2.2.1 Connecteur principal et notion de sous-formule

Une formule obtenue par application de la clause (ii) est une négation, et $\neg$ en est le *connecteur principal*. De façon analogue, une formule obtenue par application de la clause (iii) est une conjonction, une disjonction ou un conditionnel (ou « implication matérielle ») et a pour connecteur principal (respectivement) $\wedge$, $\vee$, ou $\rightarrow$.

La notion de sous-formule immédiate est donnée par les conditions suivantes :

- une variable propositionnelle n'a pas de sous-formule immédiate (c'est pourquoi on les appelle aussi *formules atomiques* ou *atomes*).
- A est la (seule) sous-formule immédiate de $\neg A$.
- A et B sont les (seules) sous-formules immédiates de $(A\wedge B)$, $(A\vee B)$ et $(A \rightarrow B)$.

On peut alors définir implicitement la notion de sous-formule par les règles (où A, B et C sont des *e.b.f.*) suivantes.

Définition 2 : sous-formule

- Si A est une sous-formule immédiate de B, ou si A est identique à B, alors A est une sous-formule de B.
- Si A est une sous-formule de B et B une sous-formule de C, alors A est une sous-formule de C.

2.2.2 Décomposition unique.

Il est possible de prouver que chaque formule ne peut être formée que d'une unique manière. En d'autres termes, il est possible de prouver que pour toute formule A, une et une seule des conditions suivantes est remplie :

(1) A est une variable propositionnelle
(2) Il y a une unique formule B telle que $A = \neg B$.
(3) Il y a une unique paire de formules A_1, A_2 et un unique connecteur binaire *- parmi ceux du langage-objet - tels que $A = (A_1 * A_2)$

Remarque :

Dans la deuxième condition, B est une sous-formule de A ; dans la troisième condition, A_1 et A_2 sont des sous-formules de A.

Le fait que chaque formule peut être décomposée d'une seule manière provient en fait de la définition d'une *e.b.f.* Les trois conditions ci-dessus permettent par exemple d'éviter qu'une conjonction puisse aussi être une disjonction. Elles permettent également, par exemple, de prouver que $(A_1\wedge A_2)$ est identique à $(B_1\wedge B_2)$ pourvu que $A_1 = B_1$ et $A_2 = B_2$.

On peut illustrer la décomposition d'une formule donnée au moyen d'un arbre de formation. Voici un exemple avec la formule $((p\wedge(q\vee r))\rightarrow(\neg s\wedge t))$.

```
               ((p∧(q∨r))→(¬s∧t))
                  /          \
           (p∧(q∨r))        (¬s∧t)
          /       \          /    \
         p       (q∨r)     ¬s      t
                 /   \      |
                q     r     s
```

Le fait qu'il n'y ait qu'une décomposition est lié à l'utilisation des parenthèses : grâce à celles-ci, on peut facilement reconnaître le connecteur principal d'une formule, et identifier les différentes sous-formules de l'expression.

L'omission de la paire de parenthèses la plus extérieure d'une formule ne provoque pas d'ambiguïté, et l'on convient en général qu'on peut les supprimer. Ainsi, notre exemple

$$((p\wedge(q\vee r))\rightarrow(\neg s\wedge t))$$

peut s'écrire :

$$(p\wedge(q\vee r))\rightarrow(\neg s\wedge t)$$

Exercice 2 :

Réaliser l'arbre de formation de chacune des formules suivantes.

$(p\vee(q\vee\neg r))\rightarrow r$

$(a\rightarrow b)\vee\neg(a\rightarrow b)$

Remarque :

On peut souligner le lien entre la définition d'une *e.b.f.* et l'arbre de formation de la façon suivante : il suffit de faire le lien entre la lecture « de bas en haut » de l'arbre de formation d'une formule *F* et la démonstration que *F* est une formule.

Si l'on reprend l'exemple $((p\wedge(q\vee r)\rightarrow(\neg s\wedge t))$, comparer la démonstration suivante avec la lecture « de bas en haut » de son arbre de formation.

(1) p, q, r, s et t sont des variables propositionnelles, donc ce sont des formules (par (i))
(2) s est une formule, donc $\neg s$ est une formule (par (ii))
(3) q et r sont des formules, donc $(q\vee r)$ est une formule (par (iii))
(4) $\neg s$ et t sont des formules, donc $(\neg s\wedge t)$ est une formule (par (iii))
(5) p et $(q\vee r)$ sont des formules, donc $(p\wedge(q\vee r))$ est une formule (par (iii))
(6) $(p\wedge(q\vee r))$ et $(\neg s\wedge t)$ sont des formules, donc $((p\wedge(q\vee r)\rightarrow(\neg s\wedge t))$ est une formule (par (iii)).

De plus, l'arbre de formation d'une formule comporte toutes ses sous-formules, ici : p, q, r, s, t, $\neg s$, $(q\vee r)$, $(\neg s\wedge t)$, $(p\wedge(q\vee r))$ et $(p\wedge(q\vee r))$.

2.2.3 Détermination systématique des *e.b.f.*

Une *e.b.f.* génère toujours un arbre de formation dont les feuilles (c'est-à-dire les formules à l'extrémité des branches) sont des variables propositionnelles. Une expression qui génère un arbre portant au moins une feuille qui n'est pas une var. prop. n'est pas une *e.b.f.*

3 SÉMANTIQUE (1) : VALUATIONS ET MODÈLES

3.1 La notion logique de sémantique.

En général, la *sémantique* étudie les rapports entre les signes d'un langage et les objets. Du point de vue extensionnel de la logique mathématique, la sémantique renvoie à l'étude de la façon dont les formules dénotent des valeurs de vérité. Les valeurs de vérité sont des objets qui n'appartiennent pas à notre langage logique $\mathcal{L}$, mais dont on se sert pour parler de $\mathcal{L}$. Dans la logique classique que nous étudions ici, il n'y a que deux valeurs de vérité, le vrai et le faux, auxquels réfèrent respectivement les signes 1 et 0.

Remarque :

Seules les formules sont susceptibles de « recevoir » une valeur de vérité. Les expressions qui ne sont pas bien formées n'en reçoivent pas.

3.2 Interprétation, valuation, valuation booléenne.

On associe une (et une seule à la fois) valeur de vérité aux variables propositionnelles au moyen d'une *fonction d'interprétation*. Il y a 2^n interprétations possibles, n étant le nombre de variables propositionnelles du langage, puisqu'il y a deux manières de choisir la valeur de la première var. prop., puis deux manières de choisir pour le deuxième, etc...

Pour exprimer qu'une interprétation i assigne la valeur 1 à la variable propositionnelle p, on écrit :

$$i(p)=1$$

La notion de *valuation* est une extension de celle d'interprétation, permettant d'assigner une valeur de vérité à des formules plus complexes que les variables propositionnelles.

Définition 3 : valuation

Une valuation d'un ensemble Σ de formules est une fonction v qui aux éléments de Σ (donc : à des formules) associe un élément de l'ensemble des valeurs de vérité.

La notion de valuation recouvre celle d'interprétation pour le cas des variables propositionnelles, mais permet de considérer le cas de formules plus complexes.

Le but est de passer à l'ensemble de toutes les formules d'un langage, que l'on va appeler l'ensemble **EBF**.

Mais nous ne nous intéressons pas à n'importe quelle valuation. On veut une fonction de valuation qui traduise correctement nos intuitions quant à la signification de nos connecteurs.[6] Par exemple, une valuation qui associe la valeur 1 à $\neg$A et à ne rend pas compte de la signification intuitive de la négation.

Définition 4 : valuation booléenne

Une valuation v de **EBF** est dite booléenne si, et seulement si elle obéit aux conditions suivantes (où A et B sont des éléments de **EBF**) :

---

[6] La sémantique d'un langage formel tel que L ne peut être analyser les arguments en langue naturelle qu'à la condition que la traduction formelle de ces arguments respecte les intuitions d'un locuteur compétent de cette langue naturelle.

- $v(\neg A) = 1$ si, et seulement si $v(A) = 0$
- $v(A \wedge B) = 1$ si, et seulement si $v(A) = 1$ et $v(B) = 1$
- $v(A \vee B) = 1$ si, et seulement si $v(A) = 1$ ou $v(B) = 1$
- $v(A \rightarrow B) = 1$ si, et seulement si $v(A) = 0$ ou $v(B) = 1$

On a donc un outil qui permet, à partir d'une interprétation quelconque, de déterminer la valeur de vérité de toute formule du langage. On peut prouver que chaque interprétation peut être étendue à exactement une valuation booléenne.

L'important est que la variation des interprétations détermine une variation de l'assignation de valeurs de vérité à toutes les formules du langage. Par exemple, à partir d'une interprétation v telle que $v(p) = 1$ et $v(q) = 0$, on a : $v(p \rightarrow q) = 0$. Mais à partir d'une autre interprétation v' telle que $v'(p) = 0$ et $v'(q) = 0$, on a : $v'(p \rightarrow q) = 1$.

3.3 Modèles

Un modèle *pour une formule A* est une valuation booléenne v telle que $v(A) = 1$. Un modèle *pour un ensemble de formules* Σ est une valuation booléenne v telle que pour toute formule $A \in \Sigma$, $v(A) = 1$.

4 SÉMANTIQUE (2) : VALIDITÉ ET DÉCISION

4.1 Satisfiabilité, validité, tautologie

Grâce aux valuations booléennes, on peut maintenant définir ce que sont une formule satisfiable, un ensemble de formules satisfiable et une tautologie.

Définition 5 : formule satisfiable
Une formule A appartenant à l'ensemble **EBF** d'un langage L pour la logique propositionnelle est (vérifonctionnellement) *satisfiable* si, et seulement si il existe au moins une valuation booléenne v telle que $v(A) = 1$.

Définition 6 : ensemble de formules satisfiable
Un ensemble Σ de formules est (vérifonctionnellement) *satisfiable* si, et seulement si il y a au moins une valuation booléenne v telle que pour chaque formule F de Σ, $v(F) = 1$.

Ex. 6 :

L'ensemble E = {a, $((b \rightarrow a) \vee c)$, $\neg c$, $(b \vee c)$} est-il satisfiable ?

E est satisfiable si, et seulement si il y a une valuation booléenne v telle que

(a) $v(a) = 1$
(b) $v((b \rightarrow a) \vee c) = 1$
(c) $v(\neg c) = 1$
(d) $v(b \vee c) = 1$

Or

(1) $v(\neg c) = 1$ si, et seulement si $v(c) = 0$.
(2) $v((b \rightarrow a) \vee c) = 1$ si, et seulement si $v(b \rightarrow a) = 1$ ou $v(c) = 1$
(3) $v(b \vee c) = 1$ si, et seulement si $v(b) = 1$ ou $v(c) = 1$.

Dans (2) et (3), on rejette le cas où $v(c) = 1$ à cause de (1).

Dans (2), $v(b \rightarrow a) = 1$ si, et seulement si $v(b) = 0$ ou $v(a) = 1$. Par (a), on sait que $v(a) = 1$, donc la clause $v(b) = 1$ de (3) n'engendre pas de contradiction.

Conclusion, les formules de E sont vraies sous une valuation booléenne v telle que $v(a) = 1$, $v(b) = 1$ et $v(c) = 0$. L'ensemble E est donc satisfiable.

Ex. 7 :

{a, $\neg a$}

Cet ensemble est satisfiable si, et seulement si il existe une valuation booléenne v telle que

$v(a) = 1$
$v(\neg a) = 1$

Or $v(\neg a) = 1$ si, et seulement si $v(a) = 0$. CONTRADICTION.

L'ensemble n'est donc pas satisfiable.

Définition 7 : tautologie

Une formule A appartenant à l'ensemble **EBF** d'un langage L pour la logique propositionnelle est une *tautologie* si, et seulement si pour toute valuation booléenne v, $v(A) = 1$. Notation : $\vDash A$

Définition 8 : formule insatisfiable

Une formule A appartenant à l'ensemble **EBF** d'un langage L pour la logique propositionnelle est *insatisfiable* si, et seulement si il n'existe pas de valuation booléenne v telle que $v(A) = 1$. On dit que A est contradictoire.

Théorème 1
Une formule A est une tautologie si, et seulement si $\neg A$ est insatisfiable.

Preuve :
Supposons que A est une tautologie. Alors pour toute valuation booléenne v : $v(A) = 1$. Donc $v(\neg A) = 0$ pour toute valuation booléenne de **EBF**. Conversement : si $\neg A$ est insatisfiable, alors pour toute valuation booléenne v, $v(\neg A) = 0$, donc $v(A) = 1$ pour toute valuation booléenne de **EBF**, c'est-à-dire est valide. ■

En logique propositionnelle, A est valide si, et seulement si A est une tautologie.

Définition 9 : conséquence logique
Une formule A est une *conséquence logique* d'un ensemble de formules Γ si A est vraie dans toute valuation booléenne qui satisfait Γ. Notation : $\Gamma \models A$.

Remarque :
L'ensemble Γ peut ne contenir qu'une seule formule... Ou être vide (dans ce dernier cas, A est en fait une tautologie).

Définition 10 : équivalence logique
Deux formules A et B sont logiquement équivalentes si, et seulement si A et B sont vraies dans les mêmes valuations booléennes, ou si chacune est une conséquence logique de l'autre. Notation : $A \equiv B$.

Ex. 8 : lois de De Morgan
$\neg(A \wedge B) \equiv (\neg A \vee \neg B)$
$\neg(A \vee B) \equiv (\neg A \wedge \neg B)$

Théorème 2 : (théorème de déduction)
$\Gamma, A \models B$ si, et seulement si $\Gamma \models (A \rightarrow B)$.

Preuve :
Supposons $\Gamma, A \models B$. Pour toute valuation v qui satisfait à la fois Γ et A, v satisfait B.
Prenons une valuation qui satisfait Γ. Supposons que cette valuation ne satisfait pas $(A \rightarrow B)$. Alors cette valuation satisfait A mais pas B. Mais si elle

satisfait Γ et A, elle doit satisfaire B. L'hypothèse de cette valuation mène à une contradiction.

La converse est similaire. ■

4.2 Méthode de décision

Les méthodes de décision pour la validité donnent une réponse correcte (oui ou non) à toutes les questions concernant la validité des formules.

Il s'agit donc de procédures permettant de déterminer si une formule est valide ou non. Puisqu'en logique propositionnelle, une formule valide est une tautologie, de telles procédures doivent permettre de déterminer si une *e.b.f.* quelconque est une tautologie ou non.

4.2.1 Tables de vérité

L'une d'entre elles, très connue, est la méthode des tables de vérité.

Cette méthode repose sur un principe d'exhaustion des cas : pour savoir si une formule est vraie pour toute valuation booléenne, il suffit de remarquer les points suivants :

- Si deux valuations booléennes v et v' sont en accord sur toutes les variables propositionnelles contenues dans une formule A, alors $v(A) = v'(A)$.
- Etant donnée une formule A, l'ensemble de toutes les valuations booléennes est donc divisible en sous ensembles, tels que toutes les valuations contenues dans un même sous ensemble soient en accord sur les variables propositionnelles contenues dans A.
- Toute valuation booléenne appartient à un (et un seul) sous ensemble.
- Démontrer qu'une formule est vraie dans toute valuation booléenne revient donc à montrer qu'elle est vraie pour toute valuation appartenant à un des sous-ensembles.
- Chaque sous-ensemble est caractérisé par une certaine distribution de valeurs de vérité sur les variables propositionnelles de la formule. Considérer tous les sous ensembles revient donc à considérer toutes les distributions possibles.

La méthode est en conséquence (pour une formule A contenant n var. prop.) :

1. faire un tableau de 2^n lignes, avec une colonne pour chaque var. prop., puis une colonne pour chaque connecteur de la formule.

2. distribuer systématiquement toutes les combinaisons possibles de valeurs de vérité sur les var. prop.
3. en suivant le rang des connecteurs[7], calculer la valeur de vérité de chaque sous-formule de la formule principale en fonction des valeurs de ses sous formules immédiates, jusqu'à calculer la valeur pour *A*.

Ex.9 :
Par exemple, pour la formule ((p∨q)→r), la table de vérité est la suivante

p	q	r	((p	∨	q)	→	r)
1	1	**1**		**1**		1	
1	1	**0**		**1**		0	
1	0	**1**		**1**		1	
1	0	**0**		**1**		0	
0	1	**1**		**1**		1	
0	1	**0**		**1**		0	
0	0	**1**		**0**		1	
0	0	**0**		**0**		1	
(1)	(1)	(1)		(2)		(3)	

(La dernière ligne précise l'ordre du calcul).

Les trois premières colonnes listent la totalité des interprétations possibles pour les variables propositionnelles contenues dans la formule. Ensuite, on commence par calculer les valeurs de (p∨q) pour chaque interprétation. Puis on prend ces valeurs et celles de r (colonnes en gras) pour calculer les valeurs de vérité de la formule dans son entier (colonne en rouge).

Ainsi, on voit que la formule n'est pas une tautologie : il y a trois ensembles de valuations dans lesquelles elle n'est pas vraie. Si la colonne en rouge n'avait comporté que des 1, la formule aurait été une tautologie. Si elle n'avait comporté que des 0, la formule aurait été contradictoire.

[7] Le rang des connecteurs est défini de la façon suivante. Si la ou les sous formules immédiates d'une formule *A* sont des var. prop., le connecteur principal de *A* est de rang 1. Si la sous formule immédiate de rang maximum de *A* est de rang *n*, alors le connecteur principal de *A* est de rang *n*+1.

Remarquons que pour n variables propositionnelles, il y a 2^n interprétations possibles. Le calcul dans une table de vérité s'arrête donc après un nombre fini d'étapes. C'est pourquoi on dit que la logique propositionnelle est *décidable* : ne serait-ce qu'avec la méthode des tables de vérité, on est en mesure de vérifier, pour toute formule, si elle est valide ou non. Mais les tables de vérité ne sont pas la seule méthode de décision à notre disposition. En fait, elles ont un gros inconvénient : leur longueur. Pour une formule à n variables propositionnelles, la table de vérité comprendra 2^n lignes. Pour un nombre élevé de var. prop., les tables de vérité prennent rapidement des proportions contraignantes. Nous allons maintenant introduire une autre procédure : celle des arbres (sémantiques), aussi appelée *méthode des tableaux*.

4.2.2 La méthode des tableaux (arbres sémantiques)

4.2.2.1 Formes argumentatives

On commence par rappeler que, dans toute valuation booléenne, on a :

1) a) Si $\neg A$ est vrai, alors A est faux
   b) Si $\neg A$ est faux alors A est vrai
2) a) Si $(A \wedge B)$ est vrai alors A et B sont tous les deux vrais
   b) Si $(A \wedge B)$ est faux, alors soit A soit B est faux
3) a) Si $(A \vee B)$ est vrai, alors soit A soit B est vrai
   b) Si $(A \vee B)$ est faux, alors A et B sont tous les deux faux
4) a) Si $(A \rightarrow B)$ est vrai, alors soit A est faux, soit B est vrai
   b) Si $(A \rightarrow B)$ est faux, alors A est vrai et B est faux

On commence par introduire dans notre langage-objet les symboles T et F, et on définit une formule signée comme une expression TA ou FA, où A est une *e.b.f.* non-signée.

L'idée des expressions signées est d'introduire la sémantique dans la syntaxe. Il devient possible alors de construire un calcul sur les formules qui respecte les contraintes sémantiques de la notion de valuation booléenne. A chaque formule TA (respectivement FA) correspond l'énoncé sémantique $v(A) = 1$ (respectivement $v(A) = 0$). On dit qu'une telle valuation est *fidèle* aux formules signées.

On introduit également la notion de conjugué d'une formule signée. Il s'agit de la formule signée obtenue en remplaçant « T » par « F » ou vice-versa dans la formule signée d'origine.

Avec la notion de formule signée, et les huit faits ci-dessus, on peut formuler les règles pour la construction, ou *règles de génération* des arbres.

Pour chaque connecteur, il y a deux règles : une pour une formule précédée par « T » et une pour une formule précédée par « F ».

On les présente schématiquement ainsi :

1) $\dfrac{\mathrm{T}\neg A}{\mathrm{F}A}$ $\quad$ $\dfrac{\mathrm{F}\neg A}{\mathrm{T}A}$

2) $\dfrac{\mathrm{T}(A\wedge B)}{\begin{array}{c}\mathrm{T}A\\ \mathrm{T}B\end{array}}$ $\quad$ $\dfrac{\mathrm{F}(A\wedge B)}{\mathrm{F}A \mid \mathrm{F}B}$

3) $\dfrac{\mathrm{T}(A\vee B)}{\mathrm{T}A \mid \mathrm{T}B}$ $\quad$ $\dfrac{\mathrm{F}(A\vee B)}{\begin{array}{c}\mathrm{F}A\\ \mathrm{F}B\end{array}}$

4) $\dfrac{\mathrm{T}(A\rightarrow B)}{\mathrm{F}A \mid \mathrm{T}B}$ $\quad$ $\dfrac{\mathrm{F}(A\rightarrow B)}{\begin{array}{c}\mathrm{T}A\\ \mathrm{F}B\end{array}}$

La formule au dessus de la barre d'inférence est la prémisse de la règle, et la ou les formules en dessous sont la (les) conclusion(s).

La règle 1) dit qu'à partir de T$\neg A$ on peut directement inférer que FA et que de F$\neg A$ on peut inférer TA. La formulation de ces règles est une conséquence des faits sémantiques 1a) (respectivement 1b)).

La règle 2) dit qu'à partir de T($A\wedge B$) on peut inférer que A et B sont tous les deux vrais. Pour la deuxième partie de la règle, elle signifie que F($A\wedge B$) se ramifie (on dit parfois « branche ») en FB, FA. Une telle bifurcation signifie ceci : de la prémisse on peut conclure qu'au moins une des conclusions est vraie, mais sans pouvoir dire laquelle. Ces règles correspondent aux faits 2a) et 2b).

On peut comprendre de la même manière les règles 3) et 4).

Ces règles de génération sont aussi appelées *formes argumentatives*. En effet, la règle pour T($A\wedge B$) exprime le fait qu'un argument correct pour justifier T($A\wedge B$) est un argument pour justifier TA et un argument pour justifier TB. De la même manière, un argument correct pour F($A\wedge B$) est soit un argument pour FA, soit un argument pour FB. Et ainsi de suite.

Sémantiquement, les formes argumentatives on la signification suivante : toute valuation booléenne fidèle à la prémisse est aussi fidèle à au moins une des branches de la conclusion.

4.2.2.2 Tableaux sémantiques

On peut utiliser les formes argumentatives pour tirer de façon systématique toutes les conséquences d'une formule signée. La méthode est la suivante :

- On inscrit la formule dans le tableau.
- Lorsqu'une formule est dans le tableau et n'a pas fait l'objet de l'application d'une règle, on lui applique la règle correspondant à son connecteur principal et sa signature.
- Le tableau est *complet* lorsqu'il n'y a plus de règle à appliquer.

La méthode des tableaux est un algorithme qui permet de répondre de façon systématique à la question : la formule A est-elle une tautologie ?

En effet, si une formule A est une tautologie, sa négation est insatisfiable (théorème 1). Donc, si aucune valuation booléenne ne satisfait $\neg A$, on peut en déduire que A est une tautologie. Si l'on construit un tableau complet pour FA, on détermine des ensembles de formules (les ensembles de conséquences). Toute valuation booléenne fidèle à la racine doit être fidèle à au moins un de ces ensembles. On dit qu'une branche d'un tableau « ferme » ou « clôt » si, et seulement si elle contient une variable propositionnelle signée et sa conjuguée (par exemple Tp et Fp). Or aucune valuation booléenne ne peut être fidèle à de telles signatures. Si toutes les branches d'un arbre pour une formule signée FA ferment, on en conclut que :

- toute valuation booléenne fidèle à la racine doit être fidèle à des formules conjuguées ;
- qu'il est impossible qu'une valuation booléenne soit fidèle à des conjuguées ;
- qu'en conséquence aucune valuation booléenne n'est fidèle à la racine ;
- donc $\neg A$ est insatisfiable ;
- donc A est donc une tautologie.

Ex. 10 :

On veut déterminer si la formule $((p\vee(q\wedge r))\rightarrow((p\vee q)\wedge(p\vee r)))$ est une tautologie. On construit donc l'arbre pour $F((p\vee(q\wedge r))\rightarrow((p\vee q)\wedge(p\vee r)))$.

$F((p\vee(q\wedge r))\rightarrow((p\vee q)\wedge(p\vee r)))$
$T(p\vee(q\wedge r))$
$F((p\vee q)\wedge(p\vee r))$
$Tp \quad | \quad T(q\wedge r)$
$F(p\vee q) \mid F(p\vee r) \qquad Tq$
$Fp \quad Fp \qquad Tr$
$Fq \quad Fr \qquad F(p\vee q) \mid F(p\vee r)$
$Fp \quad Fp$
$Fq \quad Fr$
$\perp \quad \perp \qquad \perp \quad \perp$

Le « $\perp$ » à la fin de chaque branche sert à indiquer qu'elles sont fermées. En effet, dans chaque branche on trouve une variable propositionnelle signée et sa conjuguée.

Par cet arbre, on a montré que $F((p\vee(q\wedge r))\rightarrow((p\vee q)\wedge(p\vee r)))$ est insatisfiable (on n'en dérive que des contradictions). On a donc montré que $((p\vee(q\wedge r))\rightarrow((p\vee q)\wedge(p\vee r)))$ est une tautologie.

4.2.2.3 Tableaux et modèles

Soit une formule *A*. On construit l'arbre sémantique pour T*A*. Toute branche qui ne clôt pas détermine l'ensemble de toutes les valuations booléennes fidèles aux signatures des formules contenues dans la branche. En réalité, pour caractériser cet ensemble il suffit de retenir les signatures sur les var. prop. (c'est-à-dire l'ensemble des interprétations fidèles).

Parallèlement, une branche fermée dans l'arbre pour T*A* fournit un *contre-modèle* pour *A*, c'est-à-dire une interprétation telle que toute valuation booléenne qui lui correspondant donne la valeur 0 à *A*. Conversement, toute branche qui ne clôt pas dans un arbre pour F*A* fournit un modèle pour $\neg A$; toute branche qui clôt fournit un contre-modèle pour $\neg A$.

Dans l'exemple ci-dessus, toutes les branches pour $F((p\vee(q\wedge r))\rightarrow((p\vee q)\wedge(p\vee r)))$ ferment. Donc toute valuation booléenne est un contre-modèle de $\neg((p\vee(q\wedge r))\rightarrow((p\vee q)\wedge(p\vee r)))$, ce qui constitue une définition du concept de contradiction.

Exercice 3 :

En utilisant la méthode des arbres, vérifier la validité des formules suivantes :

$(p\rightarrow(q\rightarrow p))$
$((\neg p\vee\neg q)\rightarrow\neg(p\vee q))$
$(((p\vee q)\wedge(\neg p\rightarrow\neg q))\rightarrow q)$

4.2.2.4 Tableau pour un ensemble de formules.

On peut construire un arbre pour un ensemble de formules. Pour cela, on place toutes les formules appartenant à l'ensemble les unes après les autres « à la racine » de l'arbre.

Les notions de modèle et contre-modèle telles que nous les avons expliquées précédemment s'étendent de façon immédiate au cas d'un ensemble de formules.

5 COMPLÉTUDE FONCTIONNELLE

Nous avons interprété les connecteurs en terme de valuation booléenne. On peut représenter cette interprétation par les *tables de vérité des connecteurs* suivantes :

A	B	$\neg A$	$(A\wedge B)$	$(A\vee B)$	$(A\rightarrow B)$
1	1	0	1	1	1
1	0	0	0	1	0
0	1	1	0	1	1
0	0	1	0	0	1

Définition 11 :

L'ensemble des *littéraux* du langage L est composé de l'ensemble des var. prop. de $\mathcal{L}$ (littéraux *positifs*) et des négations des var. prop (littéraux *négatifs)*.

Définition 12 : forme normale disjonctive

Une formule est en *forme normale disjonctive* (FND, ou DNF en anglais) si, et seulement si c'est une disjonction de conjonctions de littéraux.

Ex. 11 :

$((p\wedge\neg q\wedge r)\vee(p\wedge r))$

Théorème 3 : FND

Toute formule A de la logique propositionnelle est équivalente à une formule A' en FND.

Preuve :

On construit la table de vérité pour A. A chaque fois que A reçoit la valeur 1, on traduit l'interprétation des var. prop. correspondant à la ligne en une conjonction de littéraux, positifs pour les var. prop. interprétées 1, et négatif pour les autres. On construit alors A' : la disjonction de toutes les conjonctions obtenues. Par construction, A' est en FND et v(A') = 1 pour toute interprétation pour laquelle v(A) = 1. Conversement : si v(A') = 1, alors l'un au moins des disjoints de A' a la valeur 1, et, encore par construction, v(A) = 1 à la ligne correspondante. ■

Ex. 12 :

((p→q)→¬q)

p	**q**	**((p→q)→¬q)**	
1	1	0	
1	0	1	(p∧¬q)
0	1	0	
0	0	1	(¬p∧¬q)
			(p∧¬q)∨(¬p∧¬q)

Exercice 4 :

Trouver une formule A' en FND équivalente à la formule (((p→q)∧(q→ p))→(¬r→p))

Conséquence immédiate de ce qui précède, on a le théorème suivant :

Théorème 4 : équivalence FND

Toute formule de la logique propositionnelle est équivalente à une formule où les seuls connecteurs utilisés sont ¬, ∧ et ∨.

Corollaire : par les lois de De Morgan, chaque connecteur binaire de L (avec la négation) peut exprimer les deux autres. Donc toute FND est équivalente à une formule ne contenant qu'un des connecteurs binaire et la négation.

Théorème 5 : complétude fonctionnelle

Soit f un connecteur vérifonctionnel à *n* places défini par sa table de vérité sur les atomes $p_1, ..., p_n$.

Soit *F* une formule dont f est le connecteur principal. Il y a une formule *F'* écrite seulement avec des occurrences de $\neg$ et $\wedge$ et des occurrences des atomes $p_1, ..., p_n$ telle que $F \equiv F'$.

Schéma de preuve :

On utile le théorème de l'équivalence FND, puis le corollaire. ■

Il est possible d'appliquer le théorème de complétude fonctionnelle à un système dont les seuls connecteurs sont $\{\neg, \vee\}$ (même schéma de preuve) et à un système dont les seuls connecteurs sont $\{\neg, \rightarrow\}$.

Pour le théorème des FND et celui de la complétude fonctionnelle, on s'est appuyé ici sur la méthode des tables de vérités. Les preuves de complétude fonctionnelle sont particulièrement simples via les tables, et c'est pourquoi on les a conservées. Elles restent cependant d'un usage très limité en dehors de ce problème particulier et c'est pourquoi il est important de maîtriser la technique des arbres sémantiques.

Exercice 5 :

Faire la table de vérité de $(((p_1 \wedge p_2) \vee (p_3 \wedge p_4) \rightarrow p_5) \rightarrow (p_5 \vee p_4)$ puis trouver une formule en FND équivalente.

RÉFÉRENCES

- LTF Gamut [1992] : *Logic, Language and Meaning*, vol 1 : Introduction to logic, University of Chicago Press, Chicago, en particulier les chapitres 1 et 2, pp. 1-64.

[*Ouvrage remarquable par sa clarté didactique et par la richesse de ses exemples. Les méthodes de preuves introduites (déduction naturelle) sont cependant différentes de celles présentées ici.*]

- R. Smullyan [1968] : *First Order Logic*, Dover Publications, New York, en particulier la première partie, pp. 3-40.

[*Extraordinaire de précision et de concision. Une initiation à l'élégance mathématique, tout en restant d'un abord assez facile.*]