Lycée Louis-le-Grand

Année 2003–2004

Logique propositionnelle

option informatique

1 Sommaire

- formules de la logique propositionnelle ;
- sémantique : l'évaluation des formules ;
- tautologies, formules satisfiables, contradictions ;
- tables de vérité ;
- fonctions booléennes ;
- équivalence des formules ;
- formes normales conjonctives et disjonctives.

2 Formules de la logique propositionnelle

Les formules propositionnelles sont définies à l'aide de constantes, variables et connecteurs.
Les constantes sont V et F ; les variables forment un ensemble dénombrable, et on les désignera ici par des lettres romaines : p, q, ; on dispose d'un connecteur unaire $\neg$, et de quatre connecteurs binaires : $\vee$, $\wedge$, $\Rightarrow$ et $\Leftrightarrow$.
L'ensemble des formules se définit alors récursivement :

- toute constante est une formule ;
- toute variable est une formule ;
- si e est une formule et si $\oplus$ est un connecteur unaire, $(\oplus e)$ est une formule ;
- si e et f sont des formules et si $\otimes$ est un connecteur binaire, $(e \otimes f)$ est une formule.

Comme d'habitude les règles de priorité entre opérateurs permettent d'éviter l'accumulation des parenthèses :

$$p \wedge q \vee r \Rightarrow p \wedge r \text{ est } (((p \wedge q) \vee r) \Rightarrow (p \wedge r)).$$

Le nombre de connecteurs n'est pas limité, certains introduisent le ou exclusif, ou le nand (la négation de la conjonction). On en reparlera plus tard.
On ne revient pas non plus sur la représentation par des arbres des formules de la logique propositionnelle.

On pourra utiliser le typage suivant en Caml :

```
type formule = V | F
    | Variable of string
    | Neg of formule
    | Ou of formule list
    | Et of formule list
    | Implique of formule * formule
    | Equivalent of formule * formule ;;
```

On notera qu'on a anticipé sur l'associativité de $\wedge$ et de $\vee$, qui s'éclaircira quand on parlera de sémantique.

3 Sémantique : évaluation des formules

Définir une sémantique sur les formules, c'est définir une fonction d'évaluation qui leur associe, pour chaque contexte, une valeur de vérité.
Plus précisément, on désigne par $\mathcal{B}$ l'ensemble des valeurs de vérité, que nous noterons ici $\{0, 1\}$; par $\mathcal{F}$ l'ensemble des formules propositionnelles et par $\mathcal{V}$ l'ensemble (dénombrable) des variables.
Un contexte μ est une application de $\mathcal{V}$ dans $\mathcal{B}$.
L'évaluation d'une formule $f \in \mathcal{F}$ dans un contexte μ se note $[\mu]f$, c'est un élément de $\mathcal{B}$. On pourra noter $eval_\mu : f \Rightarrow [\mu]f$ la fonction d'évaluation dans le contexte μ.

$eval_\mu$ se définit récursivement de la façon suivante :

- $[\mu]V = 1$ et $[\mu]F = 0$;
- pour toute variable p, $[\mu]p = \mu(p)$;
- si e est une formule, $[\mu](\neg e) = 1 - [\mu]e = \varepsilon^{\neg}([\mu]e)$;
- si e et f sont des formules :

$$
\begin{aligned}
[\mu](e \wedge f) &= \varepsilon^{\wedge}([\mu]e, [\mu]f) = ([\mu]e) \times ([\mu]f) \\
[\mu](e \vee f) &= \varepsilon^{\vee}([\mu]e, [\mu]f) = \max([\mu]e, [\mu]f) \\
[\mu](e \Rightarrow f) &= \varepsilon^{\Rightarrow}([\mu]e, [\mu]f) \\
[\mu](e \Leftrightarrow f) &= \varepsilon^{\Leftrightarrow}([\mu]e, [\mu]f)
\end{aligned}
$$

Pour tout connecteur binaire c, on fournit la fonction $\varepsilon^c : \mathcal{B} \times \mathcal{B} \Rightarrow \mathcal{B}$.
En particulier : $\varepsilon^{\Rightarrow}(0,0) = \varepsilon^{\Rightarrow}(0,1) = \varepsilon^{\Rightarrow}(1,1) = 1$ et $\varepsilon^{\Rightarrow}(1,0) = 0$;
$\varepsilon^{\Leftrightarrow}(0,0) = \varepsilon^{\Leftrightarrow}(1,1) = 1$ et $\varepsilon^{\Leftrightarrow}(1,0) = \varepsilon^{\Leftrightarrow}(0,1) = 0$.

4 Vocabulaire

Une formule logique e est

- une tautologie si pour tout contexte μ on a $[\mu]e = 1$;
- une contradiction si pour tout contexte μ on a $[\mu]e = 0$;
- satisfiable s'il existe au moins un contexte μ pour lequel $[\mu]e = 1$.

On n'a pas encore vraiment moyen de les distinguer...

5 Tables de vérité

Par induction structurelle sur les formules on peut facilement montrer le

Théorème

Soit e une formule logique. L'ensemble $\mathcal{V}(e)$ des variables qui y figurent est un ensemble fini. Si μ_1 et μ_2 sont deux contextes qui coïncident sur $\mathcal{V}(e)$, alors $[\mu_1]e = [\mu_2]e$.

Autrement dit il suffit de connaître les valeurs des contextes sur les variables qui apparaissent effectivement dans la formule, ce qui est assez évident, non ? C'est ainsi que pour tester si une formule e telle que $n = |\mathcal{V}(e)|$ est une tautologie, il faudra tester les 2^n possibilités qui correspondent aux valeurs d'un contexte sur les variables.

Dresser la table de vérité de e, c'est dresser un tableau à 2^n lignes et (au moins) $n+1$ colonnes. Les n premières colonnes contiennent les valeurs des variables qui apparaissent dans e, et la dernière la valeur de e, chaque ligne correspondant à un contexte différent.
En pratique, on ajoute souvent des colonnes pour les valeurs des sous-expressions de l'expression considérée.
On lit sur la dernière colonne si e est une tautologie, une contradiction, ou bien satisfiable.
Cet algorithme pour décider si une formule est une tautologie est en $O(2^n)$.
On ne connaît pas de meilleur algorithme. On ne sait pas s'il en existe.

Voici par exemple la table de vérité de $(p \wedge q) \vee (\neg p \wedge r)$:

p	q	r	$(p \wedge q) \vee (\neg p \wedge r)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

6 Fonctions booléennes

Une fonction booléenne à p variables est simplement une application de $\mathcal{B}^p$ dans $\mathcal{B}$.
Si e est une formule et si $n = |\mathcal{V}|$, on peut lui associer une fonction booléenne de n variables, notée ε^e : chaque n-uplet de booléens définit un contexte (on peut affecter n'importe quelle valeur aux variables qui ne figurent pas dans e) dans lequel on évalue l'expression e.

Remarque Remarquons que $\varepsilon^{(p \wedge q)} = \varepsilon^{\wedge}$, et de même $\varepsilon^{(p \otimes q)} = \varepsilon^{\otimes}$ pour tout connecteur binaire $\otimes$ et $\varepsilon^{(\neg p)} = \varepsilon^{\neg}$.

En fait, toute fonction booléenne admet ce type de représentation.

L'idée est la suivante : soit $\varphi : \mathcal{B}^n \Rightarrow \mathcal{B}$ une fonction booléenne, et nommons p_1, p_2, $\ldots$, p_n les variables.
On dresse la table des valeurs de φ: c'est un tableau à 2^n lignes. On cherche à construire une formule, où n'interviennent que les variables p_i, dont ce tableau soit la table de vérité.
On a le choix entre deux approches : on regarde les lignes pour lesquelles le résultat vaut 1, et on dit *il faut et il suffit que l'on soit dans l'un ou l'autre de ces cas*.
Ou bien, on regarde les lignes pour lesquelles le résultat vaut 0, et on dit *il faut et il suffit que l'on ne soit dans aucun de ces cas*.

Utilisons par exemple la première approche, et notons 1_φ l'ensemble des n-uplets de booléens sur lesquels φ vaut 1. Autrement dit : $1_\varphi = \varphi^{-1}(\{1\})$.
Si p est une variable, notons $p^0 = \neg p$ et $p^1 = p$.
Avec ces notations on dispose du

Théorème

La fonction φ est représentée par la formule f suivante, c'est-à-dire que $\varepsilon^f = \varphi$ avec :

$$f = \bigvee_{(b_1,b_2,\ldots,b_n)\in 1_\varphi} \left(\bigwedge_{i=1}^{n} p_i^{b_i} \right).$$

La démonstration est laissée au lecteur attentif et . . . courageux.

En utilisant la deuxième approche, on obtiendrait évidemment le théorème dual :

Théorème

La fonction φ est représentée par la formule g suivante, c'est-à-dire que $\varepsilon^g = \varphi$ avec :

$$g = \bigwedge_{(b_1,b_2,\ldots,b_n)\in 0_\varphi} \left(\bigvee_{i=1}^{n} p_i^{1-b_i} \right).$$

On a bien sûr noté $0_\varphi = \varphi^{-1}(\{0\})$ l'ensemble des n-uplets de variables où s'annule φ.
Là encore la démonstration se ferait en vérifiant que la table de vérité de g coïncide avec le tableau de valeurs de φ.

7 Équivalence des formules

Deux formules e et f sont dites équivalentes, et on note $e \equiv f$, si pour tout contexte μ on a $[\mu]e = [\mu]f$.
Pour vérifier algorithmiquement cette équivalence, on utilise le théorème suivant

Théorème

Deux formules e et f sont équivalentes si et seulement si la formule $e \Leftrightarrow f$ est une tautologie.

C'est une conséquence assez immédiate de ce que $\varepsilon^{\Leftrightarrow}(x, y) = 1$ si et seulement si $x = y$.

On vérifie aisément les résultats suivants, pour deux variables p et q quelconques :

$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$
$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$
$$p \Rightarrow q \equiv \neg p \vee q$$
$$p \Rightarrow q \equiv \neg q \Rightarrow \neg p$$
$$p \Leftrightarrow q \equiv (p \Rightarrow q) \wedge (q \Rightarrow p)$$
$$p \Leftrightarrow q \equiv (p \wedge q) \vee (\neg p \wedge \neg q)$$

On aimerait en déduire les mêmes résultats quand p et q désignent non seulement des variables mais même des sous-formules...

Substitution dans une formule

Soit f une formule, p_1, p_2, …, p_k des variables **deux à deux distinctes**, et e_1, e_2, …, e_k des formules quelconques.
On définit inductivement la formule obtenue par substitution dans f de e_i à chaque p_i, qu'on note $\{e_1/p_1, e_2/p_2, \ldots, e_k/p_k\}f$ de la façon suivante :

- $\{e_1/p_1, \ldots, e_k/p_k\}V = V$ et $\{e_1/p_1, \ldots, e_k/p_k\}F = F$;
- si p est une variable,

$$\{e_1/p_1, \ldots, e_k/p_k\}p = \begin{cases} e_i, & \text{si } p = p_i; \\ p, & \text{si } p \text{ n'est aucune des variables } p_1, \ldots, p_k; \end{cases}$$

- $\{e_1/p_1, \ldots, e_k/p_k\}(\neg f) = \neg(\{e_1/p_1, \ldots, e_k/p_k\}f)$;
- pour tout connecteur binaire $\otimes$,

$$\{e_1/p_1, \ldots, e_k/p_k\}(e \otimes f) = (\{e_1/p_1, \ldots, e_k/p_k\}e) \otimes (\{e_1/p_1, \ldots, e_k/p_k\}f).$$

On démontre alors par induction structurelle le

Théorème

Soit p_1, p_2, $\ldots$, p_k des variables deux à deux distinctes, f une formule quelconque, et enfin μ un contexte quelconque.
On dispose alors de $[\mu](\{e_1/p_1, e_2/p_2, \ldots, e_k/p_k\}\, f) = [\mu']\, f$, où μ' est le contexte défini par $\mu'(p) = \mu(p)$ si p n'est pas une des variables p_i, et $\mu'(p_i) = [\mu]\, e_i$ pour $1 \leqslant i \leqslant k$.

On est maintenant en mesure de démontrer un nouveau théorème qui permet de généraliser au cas des sous-formules les équivalences de formules déjà vues plus haut.

Théorème

Soit p_1, p_2, $\ldots$, p_k des variables deux à deux distinctes, f et g deux formule quelconques, et enfin e_1, e_2, e_k des formules quelconques.
On suppose que $f \equiv g$.
Alors : $\{e_1/p_1, e_2/p_2, \ldots, e_k/p_k\}\, f \equiv \{e_1/p_1, e_2/p_2, \ldots, e_k/p_k\}\, g$.

✧ En effet, pour tout contexte μ:

$$[\mu](\{e_1/p_1, \ldots, e_k/p_k\}\, f) = [\mu']f = [\mu']g = [\mu](\{e_1/p_1, \ldots, e_k/p_k\}\, g).$$

✦

8 Formes normales conjonctives, disjonctives

Un littéral est ou bien une variable, ou bien la négation d'une variable. Donc une formule du genre p ou $\neg p$.
Notons au passage que la négation d'un littéral est équivalente à un littéral.
Une clause est une disjonction de littéraux, c'est-à-dire une formule du genre $\ell_1 \vee \cdots \vee \ell_r$ où chaque ℓ_i est un littéral.
Une forme normale conjonctive est une conjonction de clauses, c'est-à-dire une formule du genre $c_1 \wedge \cdots \wedge c_k$ où chaque c_j est une clause.
Une forme normale disjonctive est une disjonction d'une conjonction de littéraux.

Nous voulons montrer que toute formule est équivalent à une forme normale conjonctive (*resp.* disjonctive).
Énonçons rapidement quelques lemmes.

Lemme Toute négation d'un littéral est équivalente à un littéral.

Lemme La négation d'une disjonction (*resp.* d'une conjonction) de littéraux est équivalente à une conjonction (*resp.* une disjonction) de littéraux.

Lemme La négation d'une disjonction de conjonctions (*resp.* d'une conjonction de disjonctions) de littéraux est équivalente à une conjonction de disjonctions (*resp.* une disjonction de conjonctions) de littéraux.

Nous sommes en mesure de démontrer par induction structurelle le

Théorème

Toute formule f est équivalente à une forme normale conjonctive $f^\times$ et à une forme normale disjonctive f^+.

✧ On peut se contenter des connecteurs binaires $\wedge$ et $\vee$, comme il a été prouvé plus haut.
Le cas des constantes est facile : $V^+ = V^\times = p \vee \neg p$, $F^+ = F^\times = p \wedge \neg p$.
Le cas des variables aussi : $p^+ = p^\times = p$.
Le cas de la négation tout autant, grâce aux lemmes précédents.
Reste le cas d'une formule $e = f \vee g$ ou $e = f \wedge g$, pour laquelle on connaît déjà f^+, $f^\times$, g^+, $g^\times$.

On dispose évidemment de $(f \vee g)^+ = f^+ \vee g^+$ et de $(f \wedge g)^\times = f^\times \wedge g^\times$.
Considérons maintenant $(f \vee g)^\times$. On dispose de $f^\times = \bigwedge_i f_i$ et $g^\times = \bigwedge_i g_i$, où les f_i et les g_i sont des disjonctions de littéraux.
Mais $f \vee g \equiv f^\times \vee g^\times = (\bigwedge_i f_i) \bigvee (\bigwedge_j g_j) \equiv \bigwedge_{i,j} (f_i \vee g_j)$ où on reconnaît une conjonction de disjonctions de littéraux.
De même $f \wedge g \equiv f^+ \wedge g^+ = (\bigvee_i f'_i) \bigwedge (\bigvee_j g'_j) \equiv \bigvee_{i,j} (f'_i \wedge g'_j)$ où on reconnaît une disjonction de conjonctions de littéraux. ✦

Utilisation des tables de vérité

Le lecteur attentif aura remarqué que les théorèmes que nous avons énoncés pour la représentation des fonctions booléennes par des formules, faisaient intervenir des formes normales conjonctives ou disjonctives : associant à une formule e la fonction booléenne ε^e, et appliquant les théorèmes en question, on en déduit de nouvelles formules, sous formes normales, qui seront équivalentes à e.
C'est, en pratique, la méthode la plus naturelle et la plus rapide pour obtenir une forme normale (qu'elle soit disjonctive ou conjonctive) d'une formule propositionnelle.