Chapitre 0

Logique

Hugo Salou MPI*

Dernière mise à jour le 13 mai 2023

Table des matières

0 Motivation

Considérons la grille de Sudoku 2×2 ci-dessous.

3			2
	4	1	
	3	2	
4			1

Figure 1 – Grille de Sudoku 2×2

On modélise ce problème : on considère $P_{i,j,k}$ une variable booléenne, c'est à dire un élément de $\{\boldsymbol{V}, \boldsymbol{F}\}$, définie telle que

$$P_{i,j,k} : \text{“}m(i,j) \stackrel{?}{=} k\text{” avec } (i,j,k) \in [\![1,4]\!]^3 \text{ ..}$$

On peut définir des contraintes logiques (des expressions logiques) pour résoudre le Sudoku. Les opérateurs ci-dessous seront définis plus tard.

$$\begin{aligned} & P_{113} \\ \wedge\ & P_{1,4,2} \\ \wedge\ & P_{2,2,4} \\ \wedge\ & P_{2,3,1} \\ & \vdots \\ \wedge\ & P_{1,2,1} \to (\neg P_{1,2,2} \wedge \neg P_{1,2,3} \wedge \neg P_{1,2,4}) \\ & \vdots \end{aligned}$$

Pour résoudre le Sudoku, on peut essayer chaque cas possible. Mais, ces possibilités sont très nombreuses.

En mathématiques, on utilise une certaine logique. Il en existe d'autre, certaines où tout est vrai, certaines où il est plus facile de montrer des théorèmes, etc. On va définir une logique ayant le moins d'opérateurs possibles.

1 Syntaxe

Définition : On suppose donné un ensemble $\mathscr{P}$ de variables propositionnelles.

Définition : On définit alors l'ensemble des formules de la logique propositionnelle par induction nommée avec les règles :

— $\neg|^1$;
— $\wedge|^2$;
— $\vee|^2$;
— $\to|^2$;
— $\leftrightarrow|^2$;
— $\top|^0$;
— $\bot|^0$;
— $V|^0_{\mathscr{P}}$.

On nomme l'ensemble des formules $\mathscr{F}$.

Exemple :

$$\vee(\wedge(\rightarrow(V(P),\top(\,),\neg(\bot(\,))),\vee(\leftrightarrow(\top(\,),\top(\,)),V(r)).$$

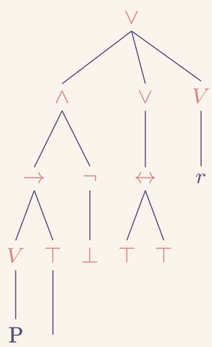

Figure 2 – Arbre syntaxique d'une expression logique

Pour simplifier la syntaxe, on écrit plutôt

$$((p \rightarrow \top) \wedge \neg\bot) \vee ((\top \leftrightarrow \top) \vee r).$$

Définition (taille d'une formule) **:** On définit, par induction, la taille notée "taille" comme

$$\begin{aligned}
\text{taille} : \mathscr{F} &\longrightarrow \mathbb{N} \\
p \in \mathscr{P} &\longmapsto 1 \\
\top &\longmapsto 1 \\
\bot &\longmapsto 1 \\
\neg G &\longmapsto 1 + \text{taille}(G) \\
G \rightarrow H &\longmapsto 1 + \text{taille}(G) + \text{taille}(H) \\
G \leftrightarrow H &\longmapsto 1 + \text{taille}(G) + \text{taille}(H) \\
G \wedge H &\longmapsto 1 + \text{taille}(G) + \text{taille}(H) \\
G \vee H &\longmapsto 1 + \text{taille}(G) + \text{taille}(H).
\end{aligned}$$

Définition (Ensemble des variables propositionnelles) **:** On définit inductivement

$$\begin{aligned}
\text{vars} : \mathscr{F} &\longrightarrow \wp(\mathscr{P}) \\
p \in \mathscr{P} &\longmapsto \{p\} \\
\top, \bot &\longmapsto \varnothing \\
\neg G &\longmapsto \text{vars}(G) \\
G \odot H &\longmapsto \text{vars}(G) \cup \text{vars}(H)
\end{aligned}$$ [1]

où $\odot$ correspond à $\cup$, $\cap$, $\rightarrow$ ou $\leftrightarrow$.

1. Le $\wp(E)$ représente ici l'ensemble des parties de E.

Définition : On appelle *substitution* une fonction de $\mathscr{P}$ dans $\mathscr{F}$ qui est l'identité partout sauf sur un ensemble fini de variables. On la note alors

$$(p_1 \mapsto H_1,\ p_2 \mapsto H_2, \ldots, p_n \mapsto H_n)$$

qui est la substitution

$$\begin{aligned} \mathscr{P} &\longrightarrow \mathscr{F} \\ p &\longmapsto \begin{cases} H_i & \text{si } p = p_i \\ p & \text{sinon.} \end{cases} \end{aligned}$$

Exemple :

La fonction

$$\sigma = (p \mapsto p \vee q,\ r \mapsto p \wedge \top)$$

est une substitution. On a $\sigma(p) = p \vee q$, $\sigma(r) = p \wedge \top$, $\sigma(q) = q$ et, pour toute autre variable logique a, $\sigma(a) = a$.

Définition (Application d'une substitution à une formule) **:** Étant donné une formule $G \in \mathscr{F}$ et une substitution σ, on définit inductivement $G[\sigma]$ par

$$\begin{cases} \top[\sigma] = \top \\ \bot[\sigma] = \bot \\ p[\sigma] = \sigma(p) \\ (\neg G)[\sigma] = \neg(G[\sigma]) \\ (G \odot H)[\sigma] = (G[\sigma]) \odot H[\sigma] \end{cases}$$

où $\odot$ correspond à $\cup$, $\cap$, $\to$ ou $\leftrightarrow$.

Exemple :

Avec $G = p \wedge (q \vee \top)$ et $\sigma = (p \mapsto p,\ q \mapsto r \wedge \top)$, on a

$$G[\sigma] = q \wedge \big((r \wedge \top) \vee \bot\big).$$

Définition : On appelle parfois *clés* d'une substitution de σ, l'ensemble des variables propositionnelles sur lequel elle n'est pas l'identité.

Définition : On définit la *composée* de deux substitutions σ et σ' par

$$\begin{aligned} \sigma \cdot \sigma' : \mathscr{P} &\longrightarrow \mathscr{F} \\ p &\longmapsto (p[\sigma])[\sigma]. \end{aligned}$$

Exemple :

Avec $\sigma = (p \mapsto q)$ et $\sigma = (q \mapsto r)$, on a

$$\sigma' \cdot \sigma = (p \mapsto r,\ q \mapsto r).$$

En effet,

$$\sigma' \cdot \sigma(x) = \begin{cases} r & \text{si } x = p \\ r & \text{si } x = q \\ x & \text{sinon.} \end{cases}$$

Exemple :
Avec $\sigma = (p \mapsto q \wedge \top)$, $\sigma' = (q \mapsto \bot,\ r \mapsto p)$, on a

$$\begin{aligned} \sigma' \cdot \sigma(x) &= \begin{cases} \bot \wedge \top & \text{si } x = p \\ \bot & \text{si} x = q \\ p & \text{si } x = r \\ x & \text{sinon} \end{cases} \\ &= (p \mapsto \bot \wedge \top,\ q \mapsto \bot,\ r \mapsto p). \end{aligned}$$

Remarque :
L'opération $\cdot$ est associative.

Propriété : Soient σ et σ' deux substitutions, on a, pour toute formule $H \in \mathscr{F}$,

$$(H[\sigma])[\sigma'] = H[\sigma' \cdot \sigma].$$

Preuve :
Notons P_G la propriété "$(G[\sigma])[\sigma'] = G[\sigma' \cdot \sigma]$"

Montrons que, pour toute formule $G \in \mathscr{F}$, P_G est vraie par induction :
- $(\top[\sigma])[\sigma'] =^{(\text{def})} \top = \top[\sigma' \cdot]$
- $(p[\sigma])[\sigma'] = p[\sigma' \cdot \sigma]$
- à faire à la maison : le cas $\neg$ et un cas $\wedge$.

□

Définition : On appelle *relation sous-formule*, la relation $\diamond$ définie dans la section 3 du chapitre -1.

2 Sémantique

2.1 Algèbre de Boole

Définition : On note $\mathbb{B} = \{\boldsymbol{V}, \boldsymbol{F}\}$ l'*ensemble des booléens.*

Définition : Sur $\mathbb{B}$, on définit les opérateurs

a	b	$a \cdot b$
$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$
$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$
$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{F}$
$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$

Table 1 – Opération $\cdot$ sur les booléens

a	b	$a + b$
$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$
$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{V}$
$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$
$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$

Table 2 – Opération $+$ sur les booléens

a	$\bar{a}$
$\boldsymbol{F}$	$\boldsymbol{V}$
$\boldsymbol{V}$	$\boldsymbol{F}$

Table 3 – Opération $\bar{\square}$ sur les booléens

Remarque :

Nom	$\cdot$	$+$
Commutativité	$a \cdot b = b \cdot a$	$a + b = b + a$
Neutre	$\boldsymbol{V} \cdot a = a$	$\boldsymbol{F} + a = a$
Absorbant	$\boldsymbol{F} \cdot a = \boldsymbol{F}$	$\boldsymbol{V} \cdot a = \boldsymbol{V}$
Associativité	$(a \cdot b) \cdot c = a \cdot (b \cdot c)$	$a + (b + c) = (a + b) + c$
Idempotence	$a \cdot a = a$	$a + a = a$
Distributivité	$a \cdot (b + c) = a \cdot b + a \cdot c$	$a + (b \cdot c) = (a + b) \cdot (a + c)$
Complémentaire	$a \cdot \bar{a} = \boldsymbol{F}$	$a + \bar{a} = \boldsymbol{V}$
Morgan	$\overline{a \cdot b} = \bar{a} + \bar{b}$	$\overline{a + b} = \bar{a} \cdot \bar{b}$

Table 4 – Règles dans $\mathbb{B}$

2.2 Fonctions booléennes

Définition (Environnement propositionnel) **:** On appelle *environnement propositionnel* une fonction de $\mathscr{P}$ dans $\mathbb{B}$.

Définition : On appelle *fonction booléenne* une fonction de $\mathbb{B}^{\mathscr{P}}$ dans $\mathbb{B}$. On note l'ensemble des fonctions booléennes $\mathbb{F}$.

Remarque :
Si $|\mathscr{P}| = n$, alors $|\mathbb{B}^{\mathscr{P}}| = 2^n$ et donc $|\mathbb{F}| = 2^{2^n}$.

Exemple :
La fonction

$$f : \begin{pmatrix} (p \mapsto \boldsymbol{F},\, q \mapsto \boldsymbol{F}) \mapsto \boldsymbol{F} \\ (p \mapsto \boldsymbol{F},\, q \mapsto \boldsymbol{V}) \mapsto \boldsymbol{V} \\ (p \mapsto \boldsymbol{V},\, q \mapsto \boldsymbol{F}) \mapsto \boldsymbol{V} \\ (p \mapsto \boldsymbol{V},\, q \mapsto \boldsymbol{V}) \mapsto \boldsymbol{V} \end{pmatrix} \in \mathbb{F}$$

est une fonction booléenne.

2.3 Interprétation d'une formule comme une fonction booléenne

Définition (Interprétation) **:** Étant donné une formule $G \in \mathcal{F}$ et un environnement propositionnel $\rho \in \mathbb{B}^{\mathcal{P}}$, on définit l'*interprétation* de G dans l'environnement ρ par

— $[\![\top]\!]^\rho = \boldsymbol{V}$;
— $[\![\bot]\!]^\rho = \boldsymbol{F}$;
— $[\![p]\!]^\rho = \rho(p)$ où $p \in \mathcal{P}$;
— $[\![\neg G]\!]^\rho = \overline{[\![G]\!]^\rho}$;
— $[\![G \wedge H]\!]^\rho = [\![G]\!]^\rho \cdot [\![H]\!]^\rho$;
— $[\![G \vee H]\!]^\rho = [\![G]\!]^\rho + [\![H]\!]^\rho$;
— $[\![G \to H]\!]^\rho = \overline{[\![G]\!]^\rho} + [\![H]\!]^\rho$;
— $[\![G \leftrightarrow H]\!]^\rho = \left(\overline{[\![G]\!]^\rho} + [\![H]\!]^\rho\right) \cdot \left(\overline{[\![H]\!]^\rho} + [\![G]\!]^\rho\right)$.

Exemple :
Avec $\rho = (p \mapsto \boldsymbol{V}, q \mapsto \boldsymbol{F})$, et $G = (p \wedge \top) \vee (q \wedge \bot)$, on a

$$\begin{aligned} [\![G]\!]^\rho &= [\![(p \wedge \top) \vee (q \wedge \bot)]\!]^\rho \\ &= [\![p \wedge \top]\!]^\rho + [\![q \wedge \bot]\!]^\rho \\ &= [\![p]\!]^\rho \cdot [\![\top]\!]^\rho + [\![q]\!]^\rho \cdot [\![\bot]\!]^\rho \\ &= \rho(p) \cdot \boldsymbol{V} + \rho(q) \cdot \boldsymbol{F} \\ &= \boldsymbol{V} + \boldsymbol{F} \\ &= \boldsymbol{V}. \end{aligned}$$

Définition (Fonction booléenne associée à une formule) **:** Étant donné une formule G, on note

$$\begin{aligned} \mathbb{F} \ni [\![G]\!] : \mathbb{B}^{\mathcal{P}} &\longrightarrow \mathbb{B} \\ \rho &\longmapsto [\![G]\!]^\rho\,. \end{aligned}$$

Exemple :
La fonction booléenne associée à $p \vee q$ est

$$f : \begin{pmatrix} (p \mapsto \boldsymbol{F},\, q \mapsto \boldsymbol{F}) \mapsto \boldsymbol{F} \\ (p \mapsto \boldsymbol{F},\, q \mapsto \boldsymbol{V}) \mapsto \boldsymbol{V} \\ (p \mapsto \boldsymbol{V},\, q \mapsto \boldsymbol{F}) \mapsto \boldsymbol{V} \\ (p \mapsto \boldsymbol{V},\, q \mapsto \boldsymbol{V}) \mapsto \boldsymbol{V} \end{pmatrix} \in \mathbb{F}.$$

La fonction booléenne associée à $p \vee (q \wedge \top)$ est aussi f ; tout comme $(p \vee \bot) \vee (q \wedge \top)$.

2.4 Liens sémantiques

Définition : On dit que G et H sont *équivalents* si et seulement si $[\![G]\!] = [\![H]\!]$. On note alors $G \equiv H$.

Définition (Conséquence sémantique) **:** On dit que H est *conséquence sémantique* de G dès lors que

$$\forall \rho \in \mathbb{B}^{\mathscr{P}},\ \big([\![G]\!]^\rho = \boldsymbol{V}\big) \implies \big([\![H]\!]^\rho = \boldsymbol{V}\big).$$

On le note $G \models H$.

Propriété : On a

$$G \equiv H \iff (G \models H \text{ et } H \models G).$$

Preuve « $\implies$ » On suppose $G \equiv H$. Soit $\rho \in \mathbb{B}^{\mathscr{P}}$. On suppose $[\![G]\!]^\rho = \boldsymbol{V}$ alors $[\![H]\!]^\rho = \boldsymbol{V}$ car $[\![G]\!] = [\![H]\!]$. On suppose maintenant $[\![H]\!]^\rho = \boldsymbol{V}$, et alors $[\![G]\!]^\rho = \boldsymbol{V}$ car $[\![G]\!] = [\![H]\!]$.

« $\impliedby$ » On suppose $G \models H$ et $H \models G$. Soit $\rho \in \mathbb{B}^{\mathscr{P}}$. On suppose $[\![G]\!]^\rho = \boldsymbol{V}$ alors $[\![H]\!]^\rho = \boldsymbol{V}$ car $H \models H$ et donc $[\![G]\!] = [\![H]\!]$. On suppose maintenant $[\![H]\!]^\rho = \boldsymbol{V}$ alors $[\![G]\!]^\rho = \boldsymbol{V}$ car $G \models H$. Par contraposée, si $[\![G]\!]^\rho = \boldsymbol{F}$, alors $[\![H]\!]^\rho = \boldsymbol{F}$. On en déduit que $[\![G]\!] = [\![H]\!]$.

$\square$

Remarque :
$\models$ n'est pas une relation d'ordre.

Remarque :
La relation $\equiv$ est une relation d'équivalence. De plus, si $G \equiv G'$ et $H \equiv H'$, alors

— $G \wedge H \equiv G' \wedge H'$;
— $G \vee H \equiv G' \vee H'$;
— $G \to H \equiv G' \to H'$;
— $G \leftrightarrow H \equiv G' \leftrightarrow H'$;
— $\neg G \equiv \neg G'$.

Une telle relation est parfois appelée une *congruence*.

Définition : On dit d'une formule $H \in \mathscr{F}$ qu'elle est
- *valide* ou *tautologique* dès lors que $\forall \rho \in \mathbb{B}^{\mathscr{P}}$, $[\![H]\!]^\rho = \boldsymbol{V}$;
- *satisfiable* dès lors qu'il existe $\rho \in \mathbb{B}^{\mathscr{P}}$, $[\![H]\!]^\rho = \boldsymbol{V}$;
- *insatisfiable* dès lors qu'il n'est pas satisfiable.

On dit de $\rho \in \mathbb{B}^{\mathscr{P}}$ tel que $[\![H]\!]^\rho = \boldsymbol{V}$ que ρ est un *modèle* de H.

Exemple :
- $p \vee \neg p$ est une tautologie. En effet, soit $\rho \in \mathscr{B}^{\mathscr{P}}$, on a
$$[\![p \vee \neg p]\!]^\rho = [\![p]\!]^\rho + \overline{[\![p]\!]^\rho} = \boldsymbol{V}.$$
- p est satisfiable mais non valide. En effet,
$$[\![p]\!]^{(p \mapsto \boldsymbol{V})} = \boldsymbol{V} \qquad \text{et} \qquad [\![p]\!]^{(p \mapsto \boldsymbol{F})} = \boldsymbol{F}.$$

— $p \wedge \neg p$ est insatisfiable. En effet, soit $\rho \in \mathbb{B}^{\mathscr{P}}$, on a

$$[\![p \wedge \neg p]\!]^\rho = [\![p]\!]^\rho \cdot \overline{[\![p]\!]^\rho} = \boldsymbol{F}.$$

Définition : Si Γ est un ensemble de formules, on écrit $\Gamma \models H$ pour dire que

$$\forall \rho \in \mathbb{B}^{\mathscr{P}},\ (\forall G \in \Gamma,\ [\![G]\!]^\rho = \boldsymbol{V}) \implies [\![H]\!]^\rho = \boldsymbol{V}.$$

Remarque :
Si Γ est fini, alors on a

$$\Gamma \models H \iff \Big(\bigwedge_{G \in \Gamma} G\Big) \models H.$$

On doit faire la preuve, pour $n \geqslant 1$,

$$\{G_1, G_2, \ldots, G_n\} \models H \iff (\cdots((G_1 \wedge G_2) \wedge G_3) \cdots \wedge G_n) \models H.$$

3 Le problème Sat – Le problème Validité

On définit le problème Sat comme ayant pour donnée une formule H et pour question “H est-elle satisfiable ?” et le problème Valide comme ayant pour donnée une formule H et pour question “H est-elle valide ?”

3.1 Résolution par tables de vérité

a	b	c	$a \wedge b$	$\neg b$	$\neg c$	$\neg b \vee \neg c$	$(a \wedge b) \to (\neg b \vee \neg c)$
$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$
$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{V}$
$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$
$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$
$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{V}$
$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{V}$
$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$
$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$

Table 5 – Table de vérité de $(a \wedge b) \to (\neg b \vee \neg c)$

Exemple :

Le problème Sat lit la colonne résultat, on cherche un $\boldsymbol{V}$. Le problème Valide lit la colonne résultat et vérifie qu'il n'y a que des $\boldsymbol{V}$.

Remarque :
Deux formules sont équivalent si et seulement si elles ont la même colonne résultat.

On essaie d'énumérer toutes les possibilités : si $|\mathscr{P}| = n \in \mathbb{N}$, alors le nombre de classes d'équivalences pour $\equiv$ est au plus 2^{2^n}. On cherche donc un meilleur algorithme.

4 Représentation des fonction booléennes

4.1 Par des formules ?

p	q	r	S
$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{V}$
$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$
$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{F}$
$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$
$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{F}$	$\boldsymbol{V}$
$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$	$\boldsymbol{F}$
$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{F}$	$\boldsymbol{V}$
$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{V}$	$\boldsymbol{F}$

Table 6 – Table de vérité d'une formule inconnue

On regarde les cas où la sortie est $\boldsymbol{V}$ et on crée une formule permettant de tester cette combinaison de p, q et r uniquement. On unie toutes ces formules par des $\vee$. Dans l'exemple ci-dessus, on obtient

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

Théorème : Soit $f : \mathbb{B}^{\mathscr{P}} \to \mathbb{B}$ une fonction booléenne avec $\mathscr{P}$ fini. Il existe une formule $H \in \mathscr{F}$ telle que $[\![H]\!] = f$.

Avant de prouver ce théorème, on démontre d'abord les deux lemme suivants et on définit lit_ρ.

Définition : Soit $\rho \in \mathbb{B}^{\mathscr{P}}$. On définit

$$\mathrm{lit}_\rho(p) = \begin{cases} p & \text{si } \rho(p) = \boldsymbol{V}\,; \\ \neg p & \text{sinon.} \end{cases}$$

Lemme :

$$\forall \rho \in \mathbb{B}^{\mathscr{P}},\ \exists G \in \mathscr{F},\ (\forall \rho' \in \mathbb{B}^{\mathscr{P}},\ [\![G]\!]^{\rho'} = \boldsymbol{V} \iff \rho = \rho').$$

On prouve ce lemme :

Preuve :
$\mathscr{P}$ est fini. Notons donc $\mathscr{P} = \{p_1, \dots, p_n\}$ ses variables. Soit alors $\rho \in \mathbb{B}^{\mathscr{P}}$, on définit

$$H_\rho = \bigwedge_{i=1}^{n} \mathrm{lit}_\rho(p_i).$$

Montrons que $[\![H_\rho]\!]^{\rho'} = \boldsymbol{V} \iff \rho = \rho'$. Soit $\rho' \in \mathbb{B}^{\mathscr{P}}$.

— Si $\rho = \rho'$, alors

$$\llbracket H_\rho \rrbracket^{\rho'} = \left\llbracket \bigwedge_{i=1}^{n} \mathrm{lit}_\rho(p_i) \right\rrbracket^{\rho'} = \mathop{\bullet}_{i=1}^{n} \llbracket \mathrm{lit}_\rho(p_i) \rrbracket^{\rho'}$$

Soit $i \in \llbracket 1, n \rrbracket$. Si $\rho(p_i) = \boldsymbol{V}$ alors $\rho'(p_i) = \boldsymbol{V}$, or, $\mathrm{lit}_\rho(p_i) = p_i$ et donc $\llbracket \mathrm{lit}_\rho(p_i) \rrbracket^{\rho'} = \llbracket p_i \rrbracket^{\rho'} = \boldsymbol{V}$; sinon si $\rho(p_i) = \boldsymbol{F}$, alors $\rho'(p_i) = \boldsymbol{F}$, or, $\mathrm{lit}_\rho(p_i) = \neg p_i$ et donc

$$\llbracket \mathrm{lit}_\rho(p_i) \rrbracket^{\rho'} = \llbracket \neg p_i \rrbracket^{\rho'} = \overline{\llbracket p_i \rrbracket^{\rho'}} = \overline{\rho'(p_i)} = \bar{\boldsymbol{F}} = \boldsymbol{V}.$$

et comme ceci étant vrai pour tout $i \in \llbracket 1, n \rrbracket$, on a

$$\mathop{\bullet}_{i=1}^{n} \llbracket \mathrm{lit}_\rho(p_i) \rrbracket^{\rho'} = \boldsymbol{V}.$$

— Sinon ($\rho \neq \rho'$), soit donc $p_i \in \mathscr{P}$ tel que $\rho(p_i) \neq \rho'(p_i)$. Si $\rho(p_i) = \boldsymbol{V}$ alors $\rho'(p_i) = \boldsymbol{F}$ et donc $\mathrm{lit}_\rho(p_i) = p_i$ et $\llbracket \mathrm{lit}_\rho\, p_i \rrbracket^{\rho'} = \rho'(p_i) = \boldsymbol{F}$; sinon si $\rho(p_i) = \boldsymbol{F}$, alors $\rho'(p_i) = \boldsymbol{V}$ et donc $\mathrm{lit}_\rho(p_i) = \neg p_i$ et $\llbracket \mathrm{lit}_\rho(p_i) \rrbracket^{\rho'} = \llbracket \neg p_i \rrbracket^{\rho'} = \overline{\llbracket p_i \rrbracket^{\rho'}} = \bar{\boldsymbol{V}} = \boldsymbol{F}$.

On en déduit donc que

$$\llbracket H_\rho \rrbracket^{\rho'} = \mathop{\bullet}_{j=1}^{n} \llbracket \mathrm{lit}_\rho(p_j) \rrbracket^{\rho'} = \boldsymbol{F}$$

car il existe $i \in \llbracket 1, n \rrbracket$ tel que $\llbracket \mathrm{lit}_\rho(p_i) \rrbracket^{\rho'} = \boldsymbol{F}$. □

On peut donc maintenant prouver le théorème :

Lemme : Considérons alors la formule

$$H = \bigvee_{\substack{\rho \in \mathbb{B}^{\mathscr{P}} \\ f(\rho) = \boldsymbol{V}}} H_\rho.$$

On a $\llbracket H \rrbracket = f$.

Preuve : — Soit $\rho \in \mathbb{B}^{\mathscr{P}}$ tel que $f(\rho) = \boldsymbol{V}$, on a donc

$$\llbracket H \rrbracket^{\rho} = \left\llbracket \bigvee_{\substack{\rho' \in \mathbb{B}^{\mathscr{P}} \\ f(\rho') = \boldsymbol{V}}} H_{\rho'} \right\rrbracket^{\rho}.$$

H_ρ apparaît donc dans cette disjonction. Or, $\llbracket H_\rho \rrbracket = \boldsymbol{V}$ et donc $\llbracket H \rrbracket^\rho = \boldsymbol{V}$.

Si $f(\rho) = \boldsymbol{F}$, alors on a vu que $\forall \rho'$ tel que $f(\rho') = \boldsymbol{V}$, alors $\rho' \neq \rho$ et donc $\llbracket H_{\rho'} \rrbracket^\rho = \boldsymbol{F}$ et donc

$$\left\llbracket \bigvee_{\substack{\rho' \in \mathbb{B}^{\mathscr{P}} \\ f(\rho') = \boldsymbol{V}}} H_{\rho'} \right\rrbracket = \boldsymbol{F}.$$

Finalement $\llbracket H \rrbracket = f$.

□

Le théorème est prouvé directement à l'aide des deux lemmes précédents.

On connaît donc la réponse à la question du nom de ce paragraphe, à savoir “peut-on représenter les fonctions booléennes par des formules ?” Oui.

4.2 Par des formules sous formes normales ?

Définition : On dit d'une formule de la forme

- p ou $\neg p$ avec $p \in \mathscr{P}$, que c'est un *littéral* ;
- $\bigwedge_{i=1}^{n} \ell_i$ où les ℓ_i sont des littéraux que c'est une *clause conjonctive* ;
- $\bigvee_{i=1}^{n} \ell_i$ où les ℓ_i sont des littéraux que c'est une *clause disjonctive* ;
- $\bigwedge_{i=1}^{n} D_i$ où les D_i qui sont des clauses disjonctives est appelée une *forme normale conjonctive* ;
- $\bigvee_{i=1}^{n} C_i$ où les C_i qui sont des clauses conjonctives est appelée une *forme normale disjonctive*.

Remarque :
On prend, comme convention, que $\bigwedge_{i=1}^{0} G_i = \top$ et $\bigvee_{i=1}^{0} G_i = \bot$.

Exemple :

La formule $\overbrace{(p \wedge \neg q)}^{\text{clause conjonctive}} \vee \overbrace{(r \wedge p)}^{\text{clause conjonctive}}$ est donc une clause normale disjonctive.

Remarque :
On écrit FMD pour une forme normale disjonctive et FNC pour une forme normale conjonctive.

Exemple :
La formule $p \wedge q \wedge \neg r$ est une clause conjonctive donc une FNC mais c'est aussi une FND.

Exemple :
La formule $\top$ est une clause conjonctive de taille 0, donc c'est une FND. Mais, c'est aussi une clause conjonctive de taille 0, donc c'est une FNC. De même, la formule $\bot$ est une FNC et une FND.

Théorème : Toute formule est équivalente à une formule sous FND et à une formule sous FNC.

Preuve :
Soit $G \in \mathscr{F}$ une formule. Soit $[\![G]\!]$ la fonction booléenne associée à G. Alors, par le théorème précédent, il existe une formule H telle que $[\![H]\!] = [\![G]\!]$ (i.e. $H \equiv G$) avec H construit dans la preuve précédente sous forme normale disjonctive. □

Exemple :
La formule $G = p \wedge (\neg q \vee p)$ a pour table de vérité la table suivante.

p	q	$[\![G]\!]$
F	***F***	***F***
F	***V***	***F***
V	***F***	***V***
V	***V***	***V***

Table 7 – Table de vérité de $p \land (\neg q \lor p)$

La forme normale disjonctive équivalente à G est $(p \land \neg q) \lor (p \land q)$.

Nous n'avons pas encore prouvé la deuxième partie du théorème mais, on essaie de trouver une formule sous FNC :

Exemple :
On reprend l'exemple de la table de vérité d'une fonction inconnue.

p	q	r	f	$\bar{f}$
F	***F***	***F***	***V***	***F***
F	***F***	***V***	***F***	***V***
F	***V***	***F***	***F***	***V***
F	***V***	***V***	***V***	***F***
V	***F***	***F***	***V***	***F***
V	***F***	***V***	***F***	***V***
V	***V***	***F***	***V***	***F***
V	***V***	***V***	***F***	***V***

Table 8 – Table de vérité d'une formule inconnue (2)

On analyse la formule $\bar{f}$ au lieu de f. Grâce à la première partie du théorème (et de la méthode pour générer cette FND), on a

$$\bar{f} = (\neg p \land \neg q \land r) \lor (\neg p \land q \land \neg r) \lor (p \land \neg q \land r) \lor (p \land q \land r).$$

Et, à l'aide des lois de De Morgan, on a

$$\bar{\bar{f}} = (p \lor q \lor \neg r) \land (p \lor \neg q \lor r) \land (\neg p \lor q \lor \neg r) \land (\neg p \lor \neg q \lor \neg r),$$

ce qui est une FNC.

À l'aide de cet algorithme, on prouve facilement la 2^nde^ partie du théorème.

Remarque :
Il est en fait possible de transformer une formule en FND en appliquant les règles suivantes à toutes les sous-formules jusqu'à obtention d'un point fixe.

- $\neg\neg H \rightsquigarrow H$;
- $\neg(G \land H) \rightsquigarrow G \lor H$;
- $\neg(G \lor H) \rightsquigarrow G \land H$;
- $(G \lor H) \land I \rightsquigarrow (G \land I) \lor (H \land I)$;
- $I \land (G \lor H) \rightsquigarrow (I \land G) \lor (I \land H)$;

- $H \land \top \rightsquigarrow H$;
- $\top \land H \rightsquigarrow H$;
- $H \lor \bot \rightsquigarrow H$;
- $\bot \lor H \rightsquigarrow H$;
- $\neg\top \rightsquigarrow \bot$;
- $\bot \land H \rightsquigarrow \bot$;
- $H \land \bot \rightsquigarrow \bot$;
- $\neg\bot \rightsquigarrow \top$;
- $\top \lor H \rightsquigarrow \top$;
- $H \lor \top \rightsquigarrow \top$.

Propriété : Soit $n \geqslant 2$ et H_n la formule $H_n = (a_1 \vee b_1) \wedge (a_2 \vee b_2) \wedge \cdots \wedge (a_n \vee b_n)$ avec $\mathscr{P}_n = \{a_1, b_1, a_2, b_2, \ldots, a_n, b_n\}$. Alors, par application de l'algorithme précédent on obtient

$$\bigvee_{P \in \wp(\llbracket 1,n \rrbracket)} \Big(\bigwedge_{j=1}^{n} \begin{cases} a_j \text{ si } j \in P \\ b_j \text{ sinon} \end{cases} \Big).$$

À faire :

Preuve (par récurrence) :

□

Remarque :
Qu'en est-il du problème Sat ? Le problème est-il simplifié pour les fnd ou les fnc ?

Oui, pour les fnd, le problème se simplifie. On considère, par exemple, la formule

$$\begin{array}{cc} & (\ell_{11} \wedge \ell_{12} \wedge \cdots \wedge \ell_{1,n_1}) \\ \vee & (\ell_{21} \wedge \ell_{22} \wedge \cdots \wedge \ell_{2,n_2}) \\ \vdots & \vdots \\ \vee & (\ell_{m,1} \wedge \ell_{m,2} \cdots \ell_{m,n_m}). \end{array}$$

On procède en suivant l'algorithme suivant : (À faire : Mettre l'algorithme à part) Pour i fixé, je lis la ligne i, puis je fabrique un environnement ρ.

Par exemple, pour $(p \wedge \neg q \wedge r \wedge \neg p) \vee (q \wedge r \wedge \neg q) \vee (p \wedge r)$, on a $\rho = (p \mapsto \boldsymbol{V}, r \mapsto \boldsymbol{V})$.

On en conclut que Sat peut être résolu en temps linéaire dans le cas d'une forme normale disjonctive. Le problème est de construire cette fnd.

Remarque :
Après s'être intéressé au problème Sat, on s'intéresse au problème Valide.

Par exemple, on considère la formule $(p \vee q \vee \neg r \vee \neg p) \wedge (p \vee \neg r \vee p \vee r) \wedge (q \vee r)$. On peut construire $\rho = (q \mapsto \boldsymbol{F}, r \mapsto \boldsymbol{F})$ est tel que $\llbracket H \rrbracket^\rho = \boldsymbol{F}$.

Si on ne peut pas construire un tel environnement propositionnel, la formule vérifie le problème Valide.

On en conclut que Valide peut être résolu en temps linéaire dans le cas d'une forme normale conjonctive. Le problème est de construire cette fnc.

5 Algorithme de Quine

L'objectif de cet algorithme est de résoudre le problème Sat. On commence par poser quelques lemmes, puis on donne l'algorithme.

Remarque :
Une forme normale peut être vue comme un ensemble d'ensembles de littéraux (c'est la représentation que nous allons utiliser en OCamL).

EXEMPLE :

L'ensemble $\{\{p,q\},\{p,r\},\varnothing\}$, a pour formule sous FNC associée $(p \vee \neg q) \wedge (q \vee r) \wedge \bot$.

L'ensemble $\{\{p,\neg q\},\{q,r\},\varnothing\}$ a pour formule sous FND associée $(p \wedge \neg q) \vee (q \wedge r) \vee \top$.

L'ensemble $\varnothing$ a pour formule sous FNC associée $\top$.

L'ensemble $\varnothing$ a pour formule sous FND associée $\bot$.

Lemme : Pour toute formule H, pour tout variable propositionnelle p et pour tout environnement propositionnel ρ, tel que $\rho(p) = \boldsymbol{V}$, alors

$$\left[\!\left[H[p \mapsto \top] \right]\!\right]^\rho = [\![H]\!]^\rho .$$

Preuve (par induction sur les formules) :

— Si $H = p$, avec $p \in \mathscr{P}$, alors

$$\left[\!\left[H[p \mapsto \top] \right]\!\right]^\rho = [\![\top]\!]^\rho = \boldsymbol{V} = [\![H]\!]^\rho.$$

— Si $H = \top$, alors $H[p \mapsto \top] = H$.

— Si $H = \neg H_1$ tel que $\left[\!\left[H_1[p \mapsto \top] \right]\!\right]^\rho = [\![H_1]\!]^\rho$, alors,

$$\begin{aligned}
\left[\!\left[H[p \mapsto \top] \right]\!\right]^\rho &= \left[\!\left[\neg(H_1[p \mapsto \top]) \right]\!\right]^\rho \\
&= \overline{\left[\!\left[H_1[p \mapsto \top] \right]\!\right]^\rho} \\
&= \overline{[\![H_1]\!]^\rho} \\
&= [\![\neg H_1]\!]^\rho \\
&= [\![H]\!]^\rho.
\end{aligned}$$

— Si $H = H_1 \wedge H_2$ avec H_1 et H_2 vérifiant l'hypothèse d'induction, alors

$$\begin{aligned}
\left[\!\left[H[p \mapsto \top] \right]\!\right]^\rho &= \left[\!\left[(H_1[p \mapsto \top]) \wedge (H_2[p \mapsto \top]) \right]\!\right]^\rho \\
&= \left[\!\left[H_1[p \mapsto \top] \right]\!\right]^\rho \cdot \left[\!\left[H_2[p \mapsto \top] \right]\!\right]^\rho \\
&= [\![H_1]\!]^\rho \cdot [\![H_2]\!]^\rho \\
&= [\![H_1 \wedge H_2]\!]^\rho.
\end{aligned}$$

□

REMARQUE :
Le résultat reste vrai en remplaçant $\boldsymbol{V}$ par $\boldsymbol{F}$ et $\top$ par $\bot$.

Lemme : Pour toute formule H, et pour toute variable propositionnelle p, H est satisfiable si, et seulement si $H[p \mapsto \top]$ est satisfiable ou $H[p \mapsto \bot]$ est satisfiable.

Preuve : " $\Longrightarrow$ " Soit $H \in \mathscr{F}$ une formule satisfiable. Il existe donc un environnement propositionnel ρ défini sur $\mathrm{vars}(H)$ tel que $[\![H]\!]^\rho = \boldsymbol{V}$.

— Si $\rho(p) = \boldsymbol{V}$, alors $\left[\!\left[H[p \mapsto \top] \right]\!\right]^\rho = \boldsymbol{V}$, d'après le lemme précédent. Donc, $H[p \mapsto \top]$ est satisfiable.

— Sinon, $\rho(p) = \boldsymbol{F}$ et donc, d'après le lemme précédent, $\left[\!\left[H[p \mapsto \bot] \right]\!\right]^\rho = \boldsymbol{V}$. La formule $H[p \mapsto \bot]$ est satisfiable.

"$\Longleftarrow$" Supposons $H[p \mapsto \top]$ satisfiable. Il existe donc un environnement propositionnel μ défini sur $\mathrm{vars}(H[p \mapsto \top])$ tel que $[\![H[p \mapsto \top]]\!]^\mu = \boldsymbol{V}$. Or, $\mathrm{vars}(H[p \mapsto \top]) = \mathrm{vars}(H) \setminus \{p\}$. On peut donc étendre μ en

$$\rho : \mathrm{vars}(H) \longrightarrow \mathbb{B}$$
$$x \longmapsto \begin{cases} \mu(x) & \text{si } x \neq p \\ \boldsymbol{V} & \text{sinon.} \end{cases}$$

Ainsi, $[\![H]\!]^\mu = [\![H[p \mapsto \top]]\!]^\rho = [\![H]\!]^\rho$ car $p \notin \mathrm{vars}(H[p \mapsto \top])$.

$\square$

Lemme : Une formule sans variables est

— satisfiable si et seulement si elle est équivalente à $\top$;

— insatisfiable si et seulement si elle est équivalente à $\bot$;

```
type formule =
  | Top | Bot
  | Var   of string
  | Not   of formule
  | And   of formule * formule
  | Or    of formule * formule
  | Imply of formule * formule
  | Equiv of formule * formule

let substitution (f: formule) (x: string) (g: string): formule =
  ⟨code déjà fait en TP⟩

exception Found of string (* arret de la boucle *)
let get_var (f: formule): string option =
  let rec aux (f: formule): unit =
    match f with
    | Top | Bot       -> ()
    | Var(y)          -> raise(Found(y))
    | Not(f1)         -> aux f1
    | And(f1, f2)
    | Or(f1, f2)
    | Imply(f1, f2)
    | Equiv(f1, f2)   -> (aux f1; aux f2)
  in try
    aux f;
    None
  with
  | Found(y) -> Some(y)

let rec test_equiv (f: formule): bool option =
  match f with
  | Var(_) -> None
  | Top    -> Some(true)
  | Bot    -> Some(false)
  | Not(h) ->
            begin
              match test_equiv h with
              | None    -> None
              | Some(b) -> Some(not b)
            end
  | And(h1, h2) ->
            begin
              match test_equiv h1, test_equiv h2 with
              | Some(false), _ | _, Some(false) -> Some(false)
              | Some(true), Some(true)          -> Some(true)
              | _                               -> None
            end
  | Or(h1, h2) ->
            begin
              match test_equiv h1, test_equiv h2 with
              | Some(true), _ | _, Some(true) -> Some(true)
```

```
                  | Some(false), Some(false)       -> Some(true)
                  | _                              -> None
                end
    | Imply(h1, h2) ->
                begin
                  match test_equiv h1, test_equiv h2 with
                  | Some(false), _
                  | Some(true), Some(true)        -> Some(true)
                  | Some(true), Some(false)       -> Some(false)
                  | _                             -> None
                end
    | Equiv(h1, h2) ->
                begin
                  match test_equiv h1, test_equiv h2 with
                  | Some(true), Some(true)
                  | Some(false), Some(false)      -> Some(true)
                  | Some(true), Some(false)
                  | Some(false), Some(true)       -> Some(false)
                  | _                             -> None
                end

let rec quine (f: formule): bool =
  match get_var f, test_equiv f with
  | _, Some(b)     -> b
  | None, _        -> failwith "cas impossible"
  | Some(p), None  -> ?
```

Code 1 – Algorithme de Quine *version zéro*

Remarque :
Dans la suite, on s'intéresse aux formules sous forme CNF. Une CNF (ou même une DNF) peut être représentée au moyen d'un ensemble d'ensemble de littéraux. Ces ensembles sont finis. Par exemple, la formule $(p \vee \neg q) \wedge (r \vee p \vee p)$ est représenté par

$$\{\{p, \neg q\}, \{r, p\}\}.$$

Algorithme 1 Algorithme *Assume*

Entrée G une CNF, p une variable propositionnelle, et $b \in \mathbb{B}$.
Sortie Une CNF équivalente à $G[p \mapsto b]$.

```
Soit ℓ_V le littéral p si b = V, ¬p sinon.
Soit ℓ_F le littéral ¬p si b = V, p sinon.
pour C ∈ G faire
    si ℓ_V ∈ C alors
        On retire C de G.
    sinon
        si ℓ_F ∈ C alors
            On retire ℓ_F de C.
```

On peut donc donner l'algorithme de Quine final :

Algorithme 2 Algorithme de QUINE

Entrée Une CNF G

si $G = \varnothing$ **alors**
    retourner OUI
sinon
    si $\varnothing \in G$ **alors**
        retourner NON
    sinon si $\exists\{\ell\} \in G$ **alors**
        si $\ell = p$, avec $p \in \mathrm{vars}(G)$ **alors**
            QUINE($Assume(G, p, \boldsymbol{V})$)
        sinon si $\ell = \neg p$, avec $p \in \mathrm{vars}(G)$ **alors**
            QUINE($Assume(G, p, \boldsymbol{F})$)
    sinon
        $p \leftarrow h(G)$
        On essaie QUINE($Assume(G, p, \boldsymbol{V})$)
        On essaie QUINE($Assume(G, p, \boldsymbol{F})$)

6 Synthèse du chapitre

— **Formules** —

On définit l'ensemble des formules $\mathscr{F}$ par induction nommé avec les règles

- $\neg|^1$;
- $\wedge|^2$;
- $\vee|^2$;
- $\to|^2$;
- $\leftrightarrow|^2$;
- $\top|^0$;
- $\bot|^0$;
- $V|^0_{\mathscr{P}}$.

On définit inductivement $\mathrm{taille}(F)$, pour $F \in \mathscr{F}$, la taille de cette formule, *i.e.* le nombre d'opérateurs dans cette formule. On définit également l'ensemble des variables $\mathrm{vars}(F)$ d'une formule $F \in \mathscr{F}$.

Une *substitution* est une fonction de $\mathscr{P}$ dans $\mathscr{F}$, où elle est l'identité partout, sauf en nombre fini de variables, alors nommés *clés* de cette substitution. On note $F[\sigma]$ l'application d'une substitution σ à une formule $F \in \mathscr{F}$. On définit la composée de deux substitutions $\sigma \cdot \sigma'$, comme $\sigma \cdot \sigma' : p \mapsto (p[\sigma])[\sigma']$, cela ne correspond pas à la définition mathématique d'une composition de fonctions.

— **Fonctions booléennes** —

Un *environnement propositionnel* est une fonction de $\mathscr{P}$ dans $\mathbb{B}$. Une *fonction booléenne* est une fonction de $\mathbb{B}^{\mathscr{P}}$ dans $\mathbb{B}$. L'ensemble $\mathbb{F}$ est l'ensemble des fonctions booléennes.

On définit inductivement l'*interprétation* d'une formule $F \in \mathscr{F}$ dans un environnement ρ. On note ce booléen $[\![F]\!]^\rho$. On note également $[\![F]\!]$ l'application $\rho \mapsto [\![F]\!]^\rho$.

— **Liens sémantiques** —

On note $G \equiv H$ si, et seulement si $[\![G]\!] = [\![H]\!]$. On note $G \models H$ dès lors que, pour $\rho \in \mathbb{B}^{\mathscr{P}}$, si $[\![G]\!]^\rho = \boldsymbol{V}$, alors $[\![H]\!]^\rho = \boldsymbol{V}$. On étend cette définition pour un ensemble Γ de formules G. On a $G \equiv H$ si, et seulement si $G \models H$ et $H \models G$.

Une formule *valide* ou *tautologique* est une formule dont l'interprétation vaut toujours $\boldsymbol{V}$, peu importe l'environnement propositionnel. Une formule satisfiable est une formule dont l'interprétation vaut $\boldsymbol{V}$, pour un certain environnement propositionnel. Si $\rho \in \mathbb{B}^{\mathscr{P}}$ vérifie $[\![F]\!]^\rho = \boldsymbol{V}$, on dit que ρ est un *modèle* de F.