Université Libre de Bruxelles

INFO-F302
Informatique Fondamentale

Synthèse

Étudiants :
Hugo Callens

Enseignants :
E. Filiot

27 septembre 2023

1 Logique propositionnelle

1.1 Construction de formules

Le vocabulaire du langage de la logique propositionnelle est composé de :

1. de propositions x, y, z, ... ; ou X, Y, Z, ... ;
2. de deux constantes vrai ($\top$ ou 1) et faux ($\bot$ ou 0) ;
3. d'un ensemble de connecteurs logiques : $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$.
4. de parantheses ( ).

1.2 Sémantique

La sémantique d'une formule est la valeur de vérité de cette formule. La valeur de vérité d'une formule Φ formée àpd propositions d'un ensemble X, évaluée avec la fonction d'interprétation V, est notée $[\![\Phi]\!]_V$. La fonction $[\![\Phi]\!]_V$ est définie par induction sur la syntaxe de Φ de la façon suivante :

- $[\![\top]\!]_V = 1$; $[\![\bot]\!]_V = 0$; $[\![x]\!]_V = V(x)$
- $[\![\neg\Phi]\!]_V = 1 - [\![\Phi]\!]_V$
- $[\![\Phi_1 \vee \Phi_2]\!]_V = \max([\![\Phi_1]\!]_V, [\![\Phi_2]\!]_V)$
- $[\![\Phi_1 \wedge \Phi_2]\!]_V = \min([\![\Phi_1]\!]_V, [\![\Phi_2]\!]_V)$
- $[\![\Phi_1 \leftarrow \Phi_2]\!]_V = \max(1 - [\![\Phi_1]\!]_V, [\![\Phi_2]\!]_V)$
- $[\![\Phi_1 \leftrightarrow \Phi_2]\!]_V = \min([\![\Phi_1 \rightarrow \Phi_2]\!]_V, [\![\Phi_2 \rightarrow \Phi_1]\!]_V)$

Nous notons $V \vDash \Phi \Leftrightarrow [\![\Phi]\!]_V = 1$ soit "V satisfait Φ."

L'information contenue dans la définition est souvent représentée sous forme de table de verité :

Φ_1	Φ_2	$\Phi_1 \vee \Phi_2$	$\Phi_1 \wedge \Phi_2$	$\Phi_1 \rightarrow \Phi_2$	$\Phi_1 \leftrightarrow \Phi_2$
0	0	0	0	1	1
0	1	1	0	1	0
1	0	1	0	0	0
1	1	1	1	1	1

Dans l'implication suivante : $\Phi_1 \rightarrow \Phi_2$, la cas où Φ_1 est faux ne nous intéresse pas. Dans ce cas, l'implication est toujours vraie.

1.3 Validité et Stabilité

1.3.1 Définitions

Une formule propositionnelle Φ est **satisfaisable** $\Leftrightarrow$ il existe une fonction d'interprétation V pour les propositions de Φ, telle que $V \vDash \Phi$.

Une formule propositionnelle Φ est **valide** $\Leftrightarrow$ pour toute fonction d'interprétation V pour les propositions de Φ, $V \vDash \Phi$.

1.3.2 Conséquence logique

Soit $\Phi_1, ..., \Phi_n, \Phi$ des formules. On dira que Φ est une **conséquence logique** de $\Phi_1, ..., \Phi_n$, noté $\Phi_1, ..., \Phi_n \vDash \Phi$, si $(\Phi_1 \wedge ... \wedge \Phi_n) \rightarrow \Phi$ est valide.

1.3.3 Equivalence

Deux formules, Φ et Ψ, sont **équivalentes** si la formule $\Phi \leftrightarrow \Psi$ est valide. On notera $\Phi \equiv \Psi$.

Pour toutes formules Φ_1, Φ_2, Φ_3 :

- $\neg\neg\Phi_1 \equiv \Phi_1$
- $\neg(\Phi_1 \wedge \Phi_2) \equiv (\neg\Phi_1 \vee \neg\Phi_2)$
- $\neg(\Phi_1 \vee \Phi_2) \equiv (\neg\Phi_1 \wedge \neg\Phi_2)$
- $\Phi_1 \wedge (\Phi_2 \vee \Phi_3) \equiv (\Phi_1 \wedge \Phi_2) \vee (\Phi_1 \wedge \Phi_3)$
- $\Phi_1 \vee (\Phi_2 \wedge \Phi_3) \equiv (\Phi_1 \vee \Phi_2) \wedge (\Phi_1 \vee \Phi_3)$
- $\Phi_1 \rightarrow \Phi_2 \equiv (\neg\Phi_1 \vee \Phi_2)$

1.3.4 Lien entre satisfaisabilité et validité

Une formule Φ est valide $\Leftrightarrow \neg\Phi$ est insatisfaisable.

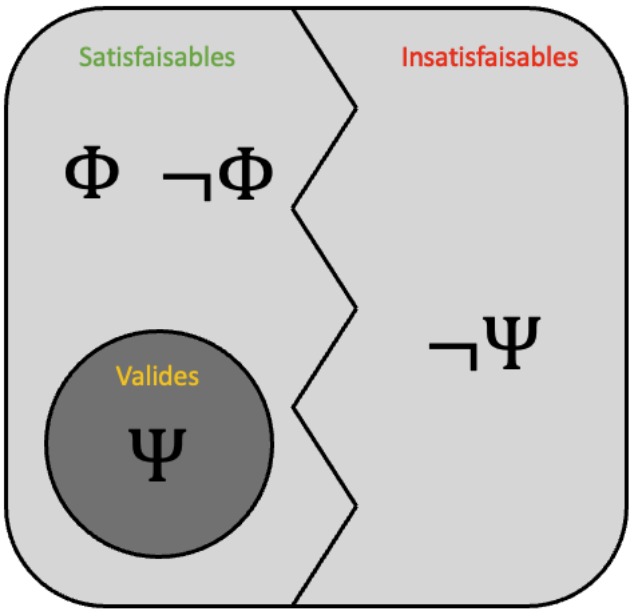

FIGURE 1 – Lien entre satisfaisabilité et validité

1.3.5 Tableaux sémantiques

Un littéral est une proposition x ou la négation d'une proposition $\neg x$.

La méthode des tableaux sémantiques est un algorithme pour tester la satisfaisabilité d'une formule. Elle consiste à construire un arbre dont les noeuds sont des formules et les feuilles sont des littéraux. On construit l'arbre de la façon suivante :

- On place la formule à tester à la racine de l'arbre.
- On applique les règles suivantes jusqu'à ce que l'arbre soit complet :
  - Si la formule à tester est une constante, on arrête.
  - Si la formule à tester est une conjonction, on ajoute les deux conjoncteurs comme fils de la formule à tester.
  - Si la formule à tester est une disjonction, on ajoute un fils avec le premier disjoncteur et un autre fils avec le deuxième disjoncteur.
  - Si la formule à tester est une implication, on ajoute un fils avec la négation de l'antécédent et un autre fils avec le conséquent.
  - Si la formule à tester est une équivalence, on ajoute un fils avec la négation de la première formule et un autre fils avec la deuxième formule.
  - Si la formule à tester est une négation, on ajoute un fils avec la négation de la formule à tester.