LIFLC – Logique classique

CM3 – Logique propositionnelle

Licence informatique UCBL – Automne 2018–2019

https://liris.cnrs.fr/ecoquery/dokuwiki/doku.php?id=enseignement:logique:start

Formules propositionnelles

Ensemble infini dénombrable $\mathcal{V}$ de variables propositionnelles notées $p, q, p', p_1, \ldots$

Ensemble des formules $\mathcal{F}$

Le plus petit ensemble stable par les règles suivantes

- Si $p \in \mathcal{V}$, p est une formule
- Si A est une formule, alors $(\neg A)$ est une formule
- Si A et B sont des formules, alors $(A \vee B)$, $(A \wedge B)$ et $(A \Rightarrow B)$ sont des formules

Abbréviations

Les parenthèses peuvent être omises dans certains cas
Ordre de priorité croissante : $\Rightarrow$, $\vee$, $\wedge$ and $\neg$

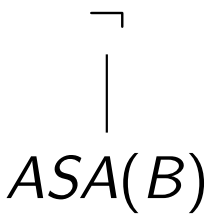
Exemple

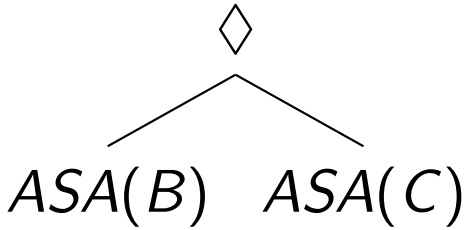
$$p \vee q \Rightarrow \neg p \wedge r \qquad = \qquad ((p \vee q) \Rightarrow ((\neg p) \wedge r))$$

Arbre de Syntaxe Abstraite

Définition

Soit A une formule. $ASA(A)$ est défini par :

- Si $p \in \mathcal{V}$, $ASA(p) = p$
- $ASA(\neg B) =$ 

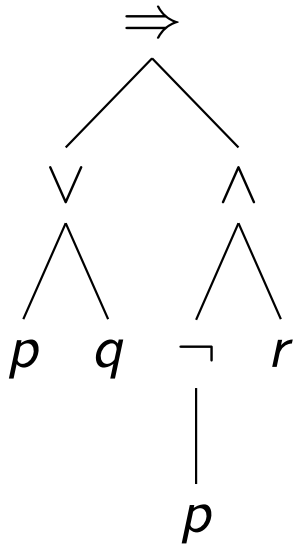
- $ASA(B \Diamond C) =$ 

  Avec $\Diamond \in \{\vee, \wedge, \Rightarrow\}$

En anglais : AST : Abstract Syntax Tree

Exemple

$$ASA(p \vee q \Rightarrow \neg p \wedge r) =$$



Sémantique

Donner du sens à une formule ?

$\rightsquigarrow$ on sait évaluer des expressions booléennes

x	$f_\neg(x)$
1	0
0	1

x	y	$f_\vee(x,y)$	$f_\wedge(x,y)$	$f_\Rightarrow(x,y)$
1	1	1	1	1
1	0	1	0	0
0	1	1	0	1
0	0	0	0	1

Les formules contiennent des *variables*, pas des booléens

Interprétations et extensions aux formules

Donner une valeur booléennes aux variables

Interprétation de variables

Fonction $I : \mathcal{V} \to \mathcal{B}$

Extension naturelle aux formules :

Interprétation de formules

$eval(A, I)$ définie par cas également notée $[\![A]\!]_I$

- si $p \in \mathcal{V}$, $eval(p, I) = I(p)$
- $eval(\neg B, I) = f_\neg(eval(B, I))$
- $eval(B \vee C) = f_\vee(\ eval(B, I)\ ,\ eval(C, I)\ )$
- $eval(B \wedge C) = f_\wedge(\ eval(B, I)\ ,\ eval(C, I)\ )$
- $eval(B \Rightarrow C) = f_\Rightarrow(\ eval(B, I)\ ,\ eval(C, I)\ )$

Satisfiabilité - Validité

Modèle

Une interprétation I est un *modèle* d'une formule A, si $eval(A, I) = 1$.

On note $I \models A$.

Satisfiabilité

Une formule est *satisfiable* si elle admet un modèle.

Validité

Une formule est *valide* si toute interprétation est un modèle de cette formule.

On note $\models A$

Attention à la surcharge de notation

Satisfiabilité - Validité

Modèle

Une interprétation I est un *modèle* d'une formule A, si $eval(A, I) = 1$.

On note $I \models A$.

Satisfiabilité

Une formule est *satisfiable* si elle admet un modèle.

Validité

Une formule est *valide* si toute interprétation est un modèle de cette formule.

On note $\models A$ Attention à la surcharge de notation

SAT

Problème SAT

Consiste à déterminer si une formule est SATisfiable.

Pas d'algorithme connu ayant une complexité au pire exponentielle dans la taille de la formule.

Le problème est dit NP-Complet :

- Il existe un algorithme Non déterministe Polynomial (dans NP)
  - On peut vérifier si une interprétation est un modèle en temps polynomial dans la taille de la formule
- Il permet de répondre aux autres problèmes NP (NP-difficile)
  - En encodant l'entrée dans une formule avec un algorithme polynomial

Validité *vs* satisfiabilité

Théorème

Une formule A est valide si et seulement si ¬A n'est pas satisfiable[a].

a. on dit aussi *insatisfiable*

Équivalence de formule

Deux formules sont *équivalentes* si elles ont le même sens. Techniquement :

Équivalence

Deux formules A et B sont équivalentes (noté $A \equiv B$)
si pour toute interprétation I :

$$eval(A, I) = eval(B, I)$$

Remarque :

- deux formules peuvent être différentes mais équivalentes
  ($\equiv$ n'est pas $=$)

Exemple

$$\neg p \vee q \equiv p \Rightarrow q$$

Conséquence logique

Définition

Un ensemble de formule $\mathcal{A}$ a pour conséquence logique une formule B (noté $\mathcal{A} \models B$) si l'affirmation suivante est juste :

- Pour toute interprétation I telle que pour toute $A \in \mathcal{A}$, $I \models A$
- on a également $I \models B$

Remarque : on omet souvent les accolades lorsque $\mathcal{A}$ est donné en extension

Exemple

$$\neg p, q \models p \Rightarrow q$$

Théorème

$A \equiv B$ si et seulement si $A \models B$ et $B \models A$

Complétude fonctionnelle

Théorème

Soit f une fonction booléenne à n arguments[a]*. Il existe une formule A_f ayant pour variables $\{p_1, \ldots, p_n\}$ telle que pour toute interprétation I :*

$$eval(A_f, I) = f(I(p_1), \ldots, I(p_n))$$

a. *i.e.* $f : \mathcal{B}^n \to \mathcal{B}$

On dit que $\{\vee, \wedge, \neg\}$ est *fonctionnellement complet*.

Substitutions

Définition

Une substitution est une fonction partielle $\sigma : \mathcal{V} \to \mathcal{F}$ dont le domaine est fini.

Notation en extension

Si $dom(\sigma) = \{p_1, \ldots, p_n\}$ et si $\sigma(p_i) = A_i$ pour $1 \leq i \leq n$,
alors on note σ :

$$[p_1 := A_1, \ldots, p_n := A_n]$$

Application d'une substitution

Définition

L'application de σ à A, notée $A\sigma$, est définie par :

- $p\sigma = \sigma(p)$ si $p \in dom(\sigma)$
- $p\sigma = p$ si $p \notin dom(\sigma)$
- $(\neg B)\sigma = (\neg(B\sigma))$
- $(B \Diamond C)\sigma = (B\sigma) \Diamond (C\sigma)$ avec $\Diamond \in \{\vee, \wedge, \Rightarrow\}$

Substitutions et sémantique

Théorème

Soit A et B deux formules et σ une substitution.

$$\text{Si } A \equiv B, \text{ alors } A\sigma \equiv B\sigma$$

Théorème (Principe de substitution)

Soient A et B deux formules et $p \in \mathcal{V}$.

$$\text{Si } A \equiv B, \text{ alors pour toute formule } C, \quad C[p := A] \equiv C[p := B]$$