Notes de cours Option Info MP/MP* : Logique

Formules logiques

Définition: On considère:

- Les constantes booléennes Vrai et Faux, on note $\mathcal{B}$ leur ensemble.
- Les connecteurs logiques $\wedge, \vee, \Rightarrow, \Leftrightarrow$.
- Un ensemble dénombrable de symboles appelées variables propositionnelles.

Les formules logiques se définissent inductivement par:

- Les constantes booléennes sont des formules logiques.
- Les variables propositionnelles sont des formules logiques.
- Si A est une formule logique alors $(\neg A)$ est une formule logique.
- Si A, B sont des formules logiques et $\oplus$ est un connecteur logique, alors $(A \oplus B)$ est une formule logique.

Remarque: On pourra ainsi faire des preuves par induction sur les formules, en considérant:
- les cas de base : constantes booléennes, variables propositionnelles
- les cas d'induction : si le résultat est vrai pour des formules A et B, il reste vrai pour $\neg A$ et $A \oplus B$ où $\oplus$ est un connecteur.

Définition: Étant donnée une formule F à n variables booléennes $v_0, \ldots, v_{n-1}$, une instanciation μ pour F est un n-uplet $(b_0, \ldots, d_{n-1}) \in \mathcal{B}^n$. Evaluer la formule F pour l'instanciation μ revient à substituer à chaque v_i la valeur b_i et faire le calcul suivant les régles de la logique (voir tables). On note le résultat $F(\mu)$. Une formule logique à n variables définit ainsi une fonction de $\mathcal{B}^n$ dans $\mathcal{B}$.

Question: Rappeler les tables de vérité des connecteurs logiques usuels.

Définitions:

- On dit qu'une formule B est *conséquence logique* d'une formule A si toute valuation satisfaisant A satisfait également B. On note $A \models B$. On peut aussi parler de conséquence logique d'un ensemble de formule Γ (et noter aussi $\Gamma \models F$).

  Donner un exemple:

- On dit que deux formules A et B sont *équivalentes* si elles définissent la même fonction booléenne. On note $A \equiv B$. (notons que $A \equiv B$ ssi ($A \models B$ et $B \models A$).)

  Donner un exemple:

- Une formule logique équivalente à Vrai est appelée une *tautologie*. On introduira le symbole $\top$ pour désigner une tautologie.

  Donner un exemple:

- Une formule logique équivalente à Faux est appelée une *contradiction* (ou *antilogie*). On introduira le symbole $\perp$ pour désigner une antilogie.

  Donner un exemple:

- Une formule logique qui n'est pas une contradiction est dite satisfiable.
- Satisfaire une formule logique F, c'est déterminer une instanciation μ telle que l'évaluation $F(\mu)$ donne Vrai.
- La table de vérité d'une formule F est l'ensemble des $F(\mu)$ pour $\mu \in \mathcal{B}^n$.

Question: Combien de lignes la table de vérité comporte-t-elle si la formule a n variables ?

Exercice: Donner la table de vérité de la formule $a \Rightarrow (b \vee c)$.

Remarque: La satisfiabilité d'une formule logique est un problème difficile. Dans le cas général, on ne connaît pas de meilleure méthode que construire la table de vérité, ce qui demande un nombre de calculs ..... en le nombre de variables.

Représentation sous forme d'arbre

On peut représenter une formule sous forme arborescente dont les feuilles sont les variables et les nœuds les connecteurs logiques. La *hauteur* de la formule est la hauteur de l'arbre. La *taille* de la formule est celle de l'arbre associé, c'est-à-dire le nombre total de nœuds, correspondant donc au nombre total de symboles de la formule (variables + connecteurs).

On pourra souvent remplacer les preuves par induction par des preuves par récurrence sur la hauteur de la formule.

On parlera aussi de *sous-formule*, correspondant à un sous-arbre de l'arbre associé.

Régles de calcul

- Double négation: pour toute formule A: $\neg(\neg A) \equiv A$.
- Lois de De Morgan: si A et B sont des formules logiques:
  - $\neg(A \wedge B) \equiv (\neg A) \vee (\neg B)$
  - $\neg(A \vee B) \equiv (\neg A) \wedge (\neg B)$
- Tiers exclu: si A est une formule:
  - $A \vee (\neg A) \equiv Vrai$
  - $A \wedge (\neg A) \equiv Faux$
- Distributivité: $\wedge$ est distributive sur $\vee$ et réciproquement.

Formes normales conjonctives et disjonctives

Définitions: On appelle littéral une variable ou la négation d'une variable. On appelle clause une disjonction (un "ou") de littéraux. Une formule est dite sous forme normale conjonctive (FNC) si c'est une conjonction (un "et") de clauses.

Définitions: On appelle clause duale une conjonction de littéraux. Une formule est dite sous forme normale disjonctive (FND) si c'est une disjonction de clauses duales.

La forme normale ..... est particulièrement adaptée pour construire la table de vérité et tester la satisfiabilité d'une formule. Réciproquement, si on construit la table de vérité, on en déduit facilement la forme normale .....

Question: Comment obtenir la FNC de F à partir de la FND de $\neg F$?

Question: Donner des exemples de situations où la forme normale conjonctive apparaît naturellement.

Proposition: Chaque formule est équivalente à une formule sous forme normale conjonctive et à une formule sous forme normale disjonctive. (preuve par induction ou par récurrence sur la hauteur de la formule laissée en exercice)

Remarque: Il n'y a pas unicité de la FNC ou FND équivalente à une formule donnée.

Exercice: Donner pour chaque formule F une FNC et une FND équivalente F ainsi qu'une FNC et une FCD équivalente à $\neg F$:

1. $F_1 = a \Rightarrow (b \vee c)$
2. $F_2 = a \Rightarrow (b \wedge c)$
3. $F_1 = (a \vee b) \Rightarrow (c \wedge d)$

Faire également l'exercice de logique du sujet CCP 2015.

Le problème n-SAT

Nous avons vu un algorithme permettant de tester la satisfiabilité d'une formule à l'aide de la table de vérité, qui a un coût ..... en le nombre de variables dans le cas général. Nous allons voir qu'on peut faire mieux pour un certain type de formules. Restreignons-nous tout d'abord aux formules sous FNC (nous avons vu que chaque formule est équivalente à une formule sous FNC). On appelle clause d'ordre n une clause qui comporte au plus n littéraux, et FNC d'ordre n une FNC qui ne comporte que des clauses d'ordre n. Le problème n-SAT consiste à décider si une FNC d'ordre n est satisfiable.

Le problème 2-SAT

(à faire après les graphes)

Représentation en Caml : TP sur la satisfiabilité