NOM : PECATTE Prénom : Timothée Jury :

Algèbre ← Entourez l'épreuve → Analyse

Sujet choisi : 916 - Formules du calcul propositionnel : représentation, formes normales, satisfiabilité. Applications.

Autre sujet : 922

I. Formules du calcul propositionnel : définitions

I.1. Syntaxe et représentations :

Les formules du calcul propositionnel peuvent être définies selon plusieurs syntaxes, mais on verra qu'elle représente la même formule en un sens qu'on précisera.

Déf 1: (Variable propositionnelle). On suppose donné un ensemble de variable V.

Déf 2: (Formules). On note $\mathcal{F}$ l'ensemble des termes sur la signature $\Sigma = \{\wedge_2, \vee_2, \neg_1, \Rightarrow_2, \bot_0\} \cup V$.

Rem 3: Cette définition des formules assure un caractère univoque aux formules du calcul prop, utile pour faire des inductions.

Déf 4: On considère l'alphabet $\mathcal{A} = \{(,), \wedge, \vee, \neg, \Rightarrow, \bot\} \cup V$ et on définit $\mathcal{F}'$ comme le langage reconnu par la grammaire :

$$S \to (S \wedge S) \mid (S \vee S) \mid \neg(S) \mid (S \Rightarrow S) \mid x \in V \text{ pour } x \in V$$

Ex 5: $((x \wedge y) \Rightarrow \neg(z)) \in \mathcal{F}'$

Thé 6: (Lecture unique). On a un isomorphisme entre $\mathcal{F}$ et $\mathcal{F}'$ (via un parsing unique dans $\mathcal{F}'$)

Rem 7: On aurait aussi pu penser la notation polonaise qui évite l'usage des parenthèses et qui est aussi biunivoque à $\mathcal{F}$ mais moins facile pour l'homme.

Rem 8: Dans la même idée, on définit parfois des règles de priorité sur les symboles pour pouvoir retirer des parenthèses tout en conservant un parsing unique.

Ex 9: (Exemple $\wedge$ est prioritaire sur $\Rightarrow$) $x \wedge y \Rightarrow \neg z$

I.2. Sémantique et définissabilité

Déf 10: (Valeurs de vérité). Une distribution de valeurs de vérité est une application δ de V dans $\{0,1\}$

Rem 11: Une formule contient un nombre fini de variables, il n'y a qu'un nombre fini de distributions de valeurs de vérité de ces variables.

Déf 11: Etant donné une évaluation $\delta : V \to \{0,1\}$, on définit par induction le prolongement $\overline{\delta}$ de δ sur $\mathcal{F}$ par :

- $\overline{\delta}(x) = \delta(x)$ si $x \in V$
- $\overline{\delta}(A \wedge B) = \min(\overline{\delta}(A), \overline{\delta}(B))$
- $\overline{\delta}(\neg A) = 1 - \overline{\delta}(A)$
- $\overline{\delta}(A \vee B) = \max(\overline{\delta}(A), \overline{\delta}(B))$
- $\overline{\delta}(A \Rightarrow B) = \overline{\delta}(B \vee \neg A)$.

Déf 12: (i) Si deux distributions δ, δ' on a $\overline{\delta}(F) = 1$, on note $\delta \models F$, F est satisfiable

(ii) Si on a $\delta \models F$ pour tout δ, on dit que F est une tautologie, note $\models F$

(iii) Si on a $\delta \not\models F$ ssi $\delta \models \bot$ pour tout δ, on dit que F est contradictoire, note $F \models \bot$

Ex 13: On a $A \Rightarrow B \equiv B \vee \neg A$ pour toute formule A, B

Rem 14: La valeur d'une formule dépend de l'évaluation de ses variables, et d'après la rem 11, il y en a qu'un nombre fini. On peut donc voir F comme une fonction de $\{0,1\}^n$ dans $\{0,1\}$, où n est le nombre de variable de F via : $\tilde{F}(a_1, \dots, a_n) = \delta(F)$ avec $\delta(x_i) = a_i$.

Déf 15: On dit qu'une fonction $f : \{0,1\}^n \to \{0,1\}$ est définissable ssi il existe une formule F t.q. $\tilde{F} = f$.

On dit qu'un système de connecteurs est complet si toute fonction de $\{0,1\}^n$ dans $\{0,1\}$ est définissable, pour tout n.

Prop 16:
- $\{\wedge, \vee, \neg, \Rightarrow, \bot\}$ est complet
- $\{\vee, \wedge, \neg\}$ est complet
- Si on a $C_1 \subset C_2$ des ensembles de connecteurs t.q. C_2 soit complet et t.q. $\forall F \in \mathcal{F}_{C_2}$, $\exists F' \in \mathcal{F}_{C_1}$ t.q. $F \equiv_2 F'$, alors C_1 est complet.

Application 17: Le système {NAND, NOR} est complet, et c'est celui qui est utilisé dans les ordinateurs car il est plus pratique d'un point de vue matériel

I.3. Preuves et complétude :

Déf 18 : Déduction naturelle

Déf 19 : On note $\vdash F$ si F peut être déduit sans hypothèse supplémentaire

Th 20 : On a $\vdash F$ ssi $\models F$ (Correction et complétude)

Rem 21 : La preuve repose essentiellement sur le fait que F est complétement caractérisée par sa fonction $\bar{F}$ associée.

Coro 22 : On a $\vdash F \Rightarrow G$ ssi pour toute évaluation δ tq $\bar{\delta}(F)$, on a $\bar{\delta}(G)$. Il suffit donc de tester les tables de vérité pour savoir si $\vdash F \Rightarrow G$.

Coro 23 : $\vdash F$ est décidable

II. Résultats avancés :

II.1. Formes normales :

Pb 24 : Y-a-t-il des représentants canoniques des classes d'équivalence $\equiv$?

Déf 25 : (Littéral). Un littéral est une formule qui est soit une variable propositionnelle, soit la négation d'une variable propositionnelle.

Rem 26 : Comme on a les lois de Morgan, on peut toujours se ramener au cas où les $\neg$ sont sur les littéraux.

Déf 27 : (FNC). Une forme normale conjonctive est une formule F sous la forme :

$$F = \bigwedge_{i \in I} \bigvee_{j \in J_i} l_j \qquad \text{où } (l_j) \text{ sont des littéraux}$$

Déf 28 : (FND). Une forme normale disjonctive est une formule F sous la forme :

$$F = \bigvee_{i \in I} \bigwedge_{j \in J_i} l_j \qquad \text{où } (l_j) \text{ sont des littéraux}$$

Prop 29 : Pour toute formule F, il existe $F_\wedge$ en FNC et $F_\vee$ en FND tq $F \equiv F_\wedge \equiv F_\vee$

Rem 30 : La preuve de la prop 16 qui donne que $\{\vee, \wedge, \neg\}$ est complet donne en fait la forme normale conjonctive équivalente.

II.2. Théorème de compacité

Déf 31 : On dit que un ensemble Σ de formules est satisfiable s'il existe une distribution de vérité qui satisfait toutes les formules de Σ. On dit que Σ est contradictoire s'il n'est pas satisfiable.

Prop 32 : Soit Σ un ensemble de formules du calcul propositionnel, F une formule ; les deux assertions suivantes sont équivalentes :
(i) $\Sigma \vdash F$
(ii) L'ensemble $\Sigma \cup \{\neg F\}$ est contradictoire.

Th 33 : Un ensemble de formules du calcul propositionnel est contradictoire si et seulement si il admet un sous-ensemble fini contradictoire.

Th 33 bis : Un ensemble de formules du calcul propositionnel est satisfiable si et seulement si il est finiment satisfiable, ie tout sous ensemble fini est satisfiable.

Application 34 : Un groupe abélien est sans torsion ssi il est totalement ordonnable (un ordre total compatible avec l'opération de groupe) → DEV 1

III. Applications :

III.1. Complexité :

Déf 35 : (P et NP). Un langage L est dans P s'il existe une machine de Turing M qui s'exécute en temps polynomial tq $w \in L$ ssi M accepte w.
De même, un langage L est dans NP s'il existe une machine de Turing M qui s'exécute en temps polynomial tq $w \in L$ ssi M accepte w (non-déterministe)

Question 36 : On a $P \subseteq NP$. Est-ce que l'inégalité est stricte ?

Déf 37 : $L \leq L'$ s'il existe une fonction bornée polynomialement tq $w \in L \Leftrightarrow f(w) \in L'$

Déf 38 : (i) L est dit NP-dur si pour tout $L' \in NL$, $L' \leq L$
(ii) L est dit NP-complet si $L \in NP$ et L est NP-dur

Prop 39 : Si L est NP-complet et que $L \in P$, alors $P = NP$.

Pb 40 : $SAT = \{ \langle F \rangle \mid F$ est une formule en CNF satisfiable $\}$

DEV2 — Th 41 : SAT est NP-complet (Cook)

Déf 42 : Un circuit booléen est un DAG où les "feuilles" sont des variables et les noeuds sont étiquetés par $\wedge, \vee, \neg$

Déf 43 : Une famille de circuit est une liste définie de circuit $(C_0, C_1, \dots)$, où C_n a n variables en entrée. On dit que $C = (C_0, C_1, \dots)$ décide un langage A sur l'alphabet $\{0,1\}^*$ si ona $w \in A$ ssi $C_n(w) = 1$ où $n = |w|$.

Déf 44 : (i) La complexité en taille d'une famille de circuit est définie par la fonction $T : n \longmapsto$ # noeuds de C_n :
(ii) La complexité en profondeur d'une famille de circuit est définie par la fonction $P : n \longmapsto$ taille du plus long chemin d'une feuille à la racine dans C_n.

Prop 45 : Il existe une famille de circuits booléen qui reconait $L = \{\langle x,y,z \rangle \mid xxy = z\}$ de complexité en taille théorie et de complexité en profondeur logarithmique.

III.2. Méthode de résolution :

Déf 46 : (Clause). On appelle clause un ensemble de littéraux sous une disjonction. On note $l_1, l_2, \dots, l_n$ une clause au lieu de $l_1 \vee l_2 \vee \dots \vee l_n$.

Déf 47 : Soit C, C_1, C_2 trois clauses. On dit que C est une résolvante de C_1 et C_2 s'il existe un littéral u tq :
(i) u appartient à C_1
(ii) $\bar{u}$ appartient à C_2
(iii) $C = (C_1 - \{u\}) \cup (C_2 - \{\bar{u}\})$

Déf 48 : Soit C un ensemble de clause. On appelle preuve par réfutation un arbre dont la racine est la clause vide, dont les feuilles sont des clauses de C, et dont un noeud est la résolvante de son fils gauche et droit.

Prop 49 : (Correction). Toute clause apparaissant dans un arbre de réfutation à partir de C est une conséquence de C.

Prop 50 : (Complétude). Soit C un ensemble de clause contradictoire. Alors il existe un arbre de résolution qui réfute C.

Déf 51 : On définit de même une clause dans le calcul des prédicats en supposant que les variables sont quantifiées universellement.

Déf 52 : Etant donné C_1, C_2, C trois clauses, on dit que C est une résolvante de C_1, C_2 si : $\exists S_1 \subseteq C_1, S_2 \subseteq C_2$ tq :
(i) $S_1 \cup \neg S_2$ est unifiable et σ est un unificateur principal
(ii) C est la clause obtenue en appliquant σ à $(C_1 - S_1) \cup (C_2 - S_2)$

Déf 53 : Définition similaire d'un arbre de résolution / réfutation.

Prop 54 : (Correction). Tout arbre de résolution à partir de Σ a pour but une conséquence de Σ.

Th 55 : (Complétude). Si un ensemble de clause Σ est contradictoire, alors il existe un arbre de réfutation de Σ. — DEV3

Réfs : • Stern : Fondements mathématiques de l'info
• Cori, Lascar
• Sipser