Université de Clermont1
IUT d'Informatique
1ière année - 2002-2003
Benoît GUGGER - Denis RICHARD - Jerzy TOMASIK

CALCUL PROPOSITIONNEL

TABLES DE VÉRITÉ
TABLEAUX DE BETH
PREUVES FORMELLES (MODUS PONENS)
RÉSOLUTION DE ROBINSON

CALCUL PROPOSITIONNEL

On considère des propositions comme :

- *La bastille fut prise en 1789*;
- *3 fois 5 font 16*;
- *La 12-millième décimale de* $\sqrt{2}$ *est 4*;
- *7 est inversible dans* $\mathbb{Z}/_{8\mathbb{Z}}$.

On les note par des lettres, A, B, C,..., X, X_1,Y_2, ... et on leur *attribue des valeurs de vérité* **VRAI (= 1) ou FAUX (= 0)**.

Le calcul propositionnel consiste d'abord en la donnée d'un ensemble PA de lettres (dites ***propositions atomiques*** ou ***variables propositionnelles***) et de toutes les applications $v : PA \to \{0,1\}$ qui attribuent à chaque variable propositionnelle une ***valeur de vérité***.

1 Connecteurs et Formes Propositionnelles

Soient A et B des propositions atomiques. On va former de nouvelles propositions (dites aussi ***formes propositionnelles***) en formalisant les *opérations logiques usuelles* de la langue naturelle.

OPÉRATION	NOM DE L'OPÉRATION	FORMALISATION
le contraire (non A)	négation	$\rceil A$ linéaire
A ou B	disjonction	$A \vee B$ binaire
A et B	conjonction	$A \wedge B$ "
si A alors B	implication	$A \to B$ "
A équivalent à B	équivalence	$A \leftrightarrow B$ "
(A ou B) mais pas (A et B)	disjonction exclusive	$A \veebar B$ "

Définition 1.1 *L'ensemble des* ***formes propositionnelles*** *se définit comme suit : toute proposition atomique est une forme propositionnelle (en abrégé* **FP***.)*;
si A *est une FP. alors* $\rceil$A *aussi*;

si A *et* B *sont des FP. alors* A $\vee$ B, A $\wedge$ B, A $\to$ B, A $\leftrightarrow$ B, A $\veebar$ B *aussi*;
toute forme propositionnelle est obtenue par applications (éventuellement répétées) des règles 1, 2 et 3 ci-dessus.

Problème d'ambiguïté

Est-ce que $A \to B \to C$ veut dire :

$$(A \to B) \to C \quad \text{ou bien} \quad A \to (B \to C)\,?$$

Les *parenthèses permettent de lever l'ambiguïté.*

Exemple de CONSTRUCTION d'une FP. :

$$\begin{array}{llll} A & & & \\ \rceil(A) & B & & \\ \rceil(A) \to & B & & A \\ (\rceil(A) \to & (B)) & \vee A & C \\ ((\rceil(A) \to & (B)) & \vee A) \leftrightarrow & C(*) \end{array}$$

Il est commode de formaliser cette construction en utilisant la notion d'**ARBRE**, de **RACINE**, de **NOEUD** et de **FEUILLE**.

1.1 Allégement d'écriture

On introduit les conventions suivantes :

Inutile de mettre les lettres elles-mêmes entre parenthèses ;
On écrit $\rceil\rceil\rceil\rceil A$ pour $\rceil(\rceil(\rceil(\rceil(A))))$;

on introduit une **PRIORITÉ** sur les **CONNECTEURS**

① $\vee$; ② $\leftrightarrow$; ③ $\to$; ④ $\vee$; ⑤ $\wedge$; ⑥ $\rceil$

Ce qui veut dire qu'on rétablit les *parenthèses dues à* ① , puis à ②, ... puis à ⑥ dans cet ordre. Pour un même connecteur *priorité "à droite"*.

Exemple : $(*)$ ci-dessus s'écrit

$(\rceil A \to B) \vee A \leftrightarrow C$
mais $\rceil A \leftrightarrow B \vee A \leftrightarrow C$ est une écriture de $(\rceil A \leftrightarrow (B \vee A)) \leftrightarrow C$

Exercice : Parenthéser $\quad A \leftrightarrow B \wedge \rceil C \vee \rceil D \to E \wedge F$

Remarque : Malgré les conventions, les parenthèses restent presque toujours nécessaires pour utiliser les formes propositionnelles.

2 Sémantique

2.1 Tables de vérité

Soit v une application de PA dans $\{0,1\}$. Autrement dit, *on veut connaître la valeur de vérité d'une FP. à partir des valeurs de vérité des lettres qui servent à la constituer*, c'est-à-dire qu'on veut étendre v à $\hat{v} : FP \to \{0,1\}$ de façon à ce que la restriction de $\hat{v}$ à PA soit v.
Pour cela, on formalise par des **TABLES DE VÉRITÉ** le sens que le raisonnement usuel donne aux connecteurs.

Sont **très naturels** les cas suivants :

$\rceil$	A
1	0
0	1

A	$\vee$	B
0	0	0
0	1	1
1	1	0
1	1	1

A	$\wedge$	B
0	0	0
0	0	1
1	0	0
1	1	1

A	$\leftrightarrow$	B
0	1	0
0	0	1
1	0	0
1	1	1

A	$\veebar$	B
0	0	0
0	1	1
1	1	0
1	0	1

La table de vérité la plus difficile à justifier concerne l'**IMPLICATION**.

Intituivement pour que A n'*implique pas* B, il faut que A soit vrai et que B soit faux. Donc $\rceil(A \to B)$ a la même table que $A \wedge \rceil B$;

par suite $A \to B$ doit intuitivement avoir la même table que $\rceil(A \wedge \rceil B)$ ou encore que $\rceil A \vee B$.

A	$\to$	B
1	1	1
1	0	0
0	1	1
0	1	0

Exercice : Construire les tables de vérité de
$\rceil(A \vee B \to A) \wedge B$
$(A \to C) \wedge (B \to C) \to (A \vee B \to C)$

Définition 2.1 *Les FP. qui prennent toujours la valeur vrai* $(= 1)$ *sont dites* ***tautologies****.*
Celles qui prennent toujours la valeur faux $(= O)$ *sont dites* ***antilogies****.*

2.2 Méthode de QUINE

On introduit deux nouvelles variables propositionnelles $\mathbb{I}$ et $\mathbb{O}$ et on étend toute application de vérité v à $\mathbb{O}$ ou $\mathbb{I}$ en posant toujours $v(\mathbb{O}) = 1$ et $v(\mathbb{I}) = 1$. Au lieu d'écrire $v(A) = 0$ (resp. $v(A) = 1$), on peut écrire $A \equiv \mathbb{O}$ (resp. $A \equiv \mathbb{I}$).

Un exemple suffit à exposer la méthode de QUINE : soit $\rceil(A \vee B \to A) \wedge B$

$A \equiv \mathbb{O}$		$A \equiv \mathbb{I}$
$\rceil(\mathbb{O} \vee B \to \mathbb{O}) \wedge B$		$\rceil(\mathbb{I} \vee B \to \mathbb{I}) \wedge B$
$\rceil(B \to \mathbb{O}) \wedge B$		$\rceil(\mathbb{I} \to \mathbb{I}) \wedge B$
$B \equiv \mathbb{O}$	$B \equiv \mathbb{I}$	$\rceil\mathbb{I} \wedge B$
$\rceil(\mathbb{O} \to \mathbb{O}) \wedge \mathbb{O} \wedge \mathbb{I}$	$\rceil(\mathbb{I} \to \mathbb{O})$	$\mathbb{O} \wedge B$
$\rceil\mathbb{I} \wedge \mathbb{O}$	$\rceil\mathbb{O} \wedge \mathbb{I}$	$\mathbb{O}$
$\mathbb{O}$	$\mathbb{I} \wedge \mathbb{I}$	
$\mathbb{I}$		

Remarque : *Méthode plus rapide que les tables de vérité en pratique*
(même complexité théorique).

2.3 Vérifier-Prouver : Les preuves formelles par la règle Modus Ponens.

Les tables de vérité permettent d'affirmer qu'une proposition est une **TAUTOLOGIE** en *calculant* tous les "cas possibles", c'est-à-dire en vérifant *pour tous les exemples.*

C'est comme si, pour vérifier que

$\forall n \leq 263\ [1^3 + 2^3 \cdots + n^3 = (1 + 2 + \cdots + n)^2]$,

on faisait $n = 1$, puis $n = 2$, jusqu'à $n = 263$.

Mais on peut aussi ***démontrer*** (ou ***prouver***) l'égalité ci-dessus par ***récurrence*** pour $n \in \mathbb{N}^*$. On le fera en arithmétique. Nous allons introduire une notion de ***preuve formelle*** en *calcul propositionnel.*

Pour cela, comme en géométrie euclidienne à partir du **POSTULAT d'EUCLIDE**, nous allons introduire :
des **AXIOMES**,
une manière de **DÉMONTRER**.

AXIOMES POUR LE CALCUL PROPOSITIONNEL

(S1)	$A \to (B \to A)$	} (contient $A \to A$)
(S2)	$(A \to (A \to B)) \to (A \to B)$	} (contient $A \to A$)
(S3)	$(A \to B) \to ((B \to C) \to (A \to C))$	(transitivite de $\to$)
(S4)	$(A \leftrightarrow B) \to (A \to B)$	} (liens entre $\to$ et $\leftrightarrow$)
(S5)	$(A \leftrightarrow B) \to (B \to A)$	} (liens entre $\to$ et $\leftrightarrow$)
(S6)	$(A \to B) \to ((B \to A) \to (A \leftrightarrow B))$	} (liens entre $\to$ et $\leftrightarrow$)
(S7)	$(\rceil B \to \rceil A) \to (A \to B))$	contraposition
(S8)	$A \vee B \leftrightarrow (\rceil A \to B)$	$\vee$ à partir de $\rceil$ et $\to$
(S9)	$A \wedge B \leftrightarrow \rceil(A \to \rceil B)$	$\wedge$ à partir de $\rceil$ et $\to$

Règle de démonstration

$$\frac{H, H \to C}{C}$$ Règle dite **Modus Ponens** (MP)

qui *veut dire* que de H (une *hypothèse*) et de $H \to C$ (un *raisonnement*), on peut conclure C.

Une **PREUVE FORMELLE** d'une proposition T est une suite finie $(A_1, \ldots, A_n)$ telle que :
- A_n est T
- pour $1 \leq i \leq n$:

  - ou bien A_1 est une *instance* d'axiome,
  - ou bien il existe $j < i$ et $k < i$ tels que A_i se déduit de A_j et A_k en utilisant la règle **Modus Ponens**.

Exemples de preuves formelles

(*) Preuve de $A \to A$

$A \to (A \to A)$	par S1
$(A \to (A \to A)) \to (A \to A)$	par S2
$(A \to A)$	par Modus Ponens.

Remarque : *La preuve formelle de chaque axiome est la suite réduite à cet axiome.*

- Preuve de $A \vee \rceil A$ (*le tiers exclu*)

$(A \vee \rceil A) \leftrightarrow (\rceil A \to \rceil A)$	par S8
$((A \vee \rceil A) \leftrightarrow (\rceil A \to \rceil A)) \to ((\rceil A \to \rceil A) \to (A \vee \rceil A))$	par S5
$(\rceil A \to \rceil A) \to (A \vee \rceil A)$	par Modus Ponens
$\rceil A \to \rceil A$	(démontré ci-dessus en (*))
$A \vee \rceil A$	par Modus Ponens.

En fait, les preuves formelles peuvent être difficiles à trouver par cette méthode. Par exemple, la simple et naturelle proposition $A \to \rceil\rceil A$ ne se prouve pas aisément.

Nous laissons cela au lecteur et prouvons ci-dessous la réciproque : $\rceil\rceil A \to A$

$\rceil\rceil A \to (\rceil\rceil\rceil A \to \rceil\rceil A)$	par (S1)
$(\rceil\rceil\rceil A \to \rceil\rceil A) \to (\rceil A \to \rceil\rceil\rceil A)$	par (S7)
$(\rceil A \to \rceil\rceil\rceil A) \to (\rceil\rceil A \to A)$	par (S7)

On peut ensuite montrer (exercice pour le lecteur) en utilisant S3 plusieurs fois que :

$$\neg\neg A \to (\neg\neg A \to A).$$

La preuve s'achève alors ainsi :

$(\neg\neg A \to (\neg\neg A \to A)) \to (\neg\neg A \to A)$	par (S2)
$\neg\neg A \to A$	par Modus Ponens.

On va introduire un mode de démonstration plus simple.

2.4 Autre mode de démonstration syntaxique : Les tableaux de BETH (1956)

On va introduire des arbres, dits **TABLEAUX** de **BETH** et un **FORMALISME** adéquat :

$\overline{A}$ veut dire : *je nie A*

A (dans un tableau) veut dire : *J'affirme A*

$\perp$ veut dire : *c'est absurde*

pour les connecteurs $\neg$, $\vee$, $\wedge$, $\to$ on introduit des :
(Ne pas confondre $\bar{A}$ et $\neg A$)

Règles d'affirmation	**Règles de réfutation**
$\neg A$ $\overline{A}$	$\overline{\neg A}$ A
$A \vee B$ $\swarrow \quad \searrow$ $A \qquad B$	$\overline{A \vee B}$ $\overline{A}$ $\overline{B}$
$A \wedge B$ A B	$\overline{A \wedge B}$ $\swarrow \quad \searrow$ $\overline{A} \qquad \overline{B}$
$A \to B$ $\swarrow \quad \searrow$ $\overline{A} \qquad B$	$\overline{A \to B}$ A $\overline{B}$

Règle $\perp$
A
$\overline{A}$
$\perp$

Avec ces règles, pour démontrer une forme propositionnelle F, on nie F (en écrivant $\overline{F}$) et on développe un arbre dit **TABLEAU** de **BETH** dont $\overline{F}$ est la racine à l'aide des règles précédentes. Une branche contenant $\perp$ comme feuille est dite **CLOSE** et s'arrête en $\perp$.

Définition 2.2 *La proposition* A *est démontrable ssi* A *est la première ligne d'un tableau, (ou racine de ce tableau considéré comme arbre) dont toutes les branches sont closes.*

Remarque : Cela veut dire qu'en niant A et en raisonnant de façon AUTOMATIQUE, on obtient toujours l'ABSURDE.

Exemples :

Preuve de $A \vee \rceil A$

$$\dfrac{\dfrac{}{A \vee \rceil A}}{\dfrac{\dfrac{A}{\rceil A}}{\bot}}$$

Preuve de $(A \rightarrow B \wedge \rceil B) \rightarrow \rceil A$

$$\dfrac{\dfrac{}{(A \rightarrow B \wedge \rceil B) \rightarrow \rceil A}}{\dfrac{\dfrac{(A \rightarrow B) \wedge \rceil B}{\rceil A}}{\dfrac{\dfrac{A}{A \rightarrow B}}{\dfrac{\rceil B}{B}}}}$$

$$\dfrac{\overline{A} \qquad B}{\bot \qquad \bot}$$

Preuve de $(A \rightarrow B) \wedge A \rightarrow B$

$$\dfrac{\dfrac{}{(A \rightarrow B) \wedge A \rightarrow B}}{\dfrac{\dfrac{(A \rightarrow B) \wedge A}{B}}{\dfrac{A \rightarrow B}{A}}}$$

$$\dfrac{\overline{A} \qquad B}{\bot \qquad \bot}$$

CONCLUSION :

En résumé, on a vu deux méthodes pour **VÉRIFIER** qu'une proposition prend toujours la valeur vraie :
1ère méthode : par **TABLE de VÉRITÉ**
2ème méthode : celle de **QUINE**.

On a vu aussi deux méthodes de démonstration formelle (l'une axiomatique avec usage de modus ponens, l'autre par l'utilisation des tableaux de Beth sans axiomes mais avec beaucoup de règles).

On démontre (et nous ne le faisons pas ici) en LOGIQUE MATHÉMATIQUE le résultat fondamental suivant dit :

THÉORÈME DE COMPLÉTUDE DU CALCUL PROPOSITIONNEL

Une **PROPOSITION** est une **TAUTOLOGIE** si et seulement si elle a une **PREUVE FORMELLE** ou si et seulement si elle est démontrable par la méthode des **TABLEAUX** de **BETH**.

Exercice : Montrer que les *deux* méthodes de démonstration formelles sont équivalentes.

Les cours de 1ère année s'arrête ici concernant le calcul propositionnel, voici ce qu'on pourrait ajouter en 2ème année, 3ème année, en licence d'informatique, ou en école d'ingénieur.

3 La résolution de ROBINSON

On a vu les modes de démonstration liés :

- aux preuves formelles obtenues par axiomatisation et par utilisation de la règle **MODUS PONENS** ;

- aux tableaux de **BETH**.

On introduit maintenant une nouvelle notion de preuve formelle, importante par son utilisation actuelle en INFORMATIQUE, et qui est connue sous le nom de preuve par **RÉSOLUTION de ROBINSON**.

Présentation arborescente des formules

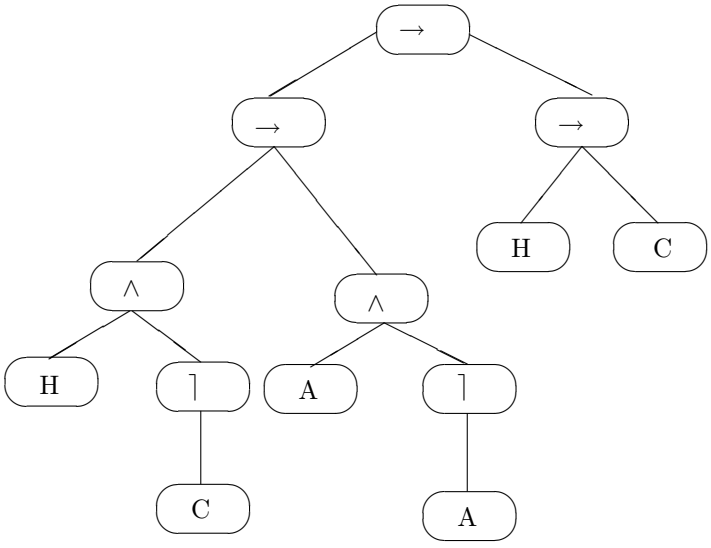
Définition 3.1 *Une* ***formule*** *du* ***calcul propositionnel*** *est un* ***arbre*** A *tel que :*

ou bien A *est réduit à un sommet (ou noeud)* ***terminal*** *dont le* ***label*** *(ou* ***étiquette****) est une* ***variable propositionnelle*** *(donc une formule) ;*

ou bien la ***racine*** *(****sommet commun*** *à toutes les* ***branches*** *de* A*) a pour étiquette l'un des symboles* $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ *et possède deux successeurs et chacun des* ***sous-arbres droit*** *et* ***gauche*** *sont (par définition récursif) des formules ;*

ou bien la racine de A *est étiquetée par le symbole* $\rceil$ *et alors cette racine possède un seul successeur dont le sous-arbre correspondant est une formule.*

Exemple :



$$[(H \wedge \rceil C) \rightarrow (A \wedge \rceil A)] \rightarrow (H \rightarrow C)$$

(arbre ou formule formalisant le raisonnement par l'absurde).

3.1 Complétude fonctionnelle des symboles $\{\rceil, \wedge, \vee\}$:

Formes normales disjonctives et formes normales conjonctives

Considérons la formule F $\quad [(\rceil A \vee B) \wedge C] \rightarrow (A \wedge \rceil B)$

$\rceil$	A	$\vee$	B	$\wedge$	C	$\rightarrow$	A	$\wedge$	$\rceil$	B
0	1	1	1	1	1	**0**	1	0	0	1
0	1	1	1	0	0	**1**	1	0	0	1
0	1	0	0	0	1	**1**	1	1	1	0
0	1	0	0	0	0	**1**	1	1	1	0
1	0	1	1	1	1	**0**	0	0	0	1
1	0	1	1	0	0	**1**	0	0	0	1
1	0	1	0	1	1	**0**	0	0	1	0
1	0	1	0	0	0	**1**	0	0	1	0

Considérons la table de vérité de F (construite en utilisant les priorités).

A) Formes Normales Conjonctives :

Le résultat $(= 0)$ de la distribution de vérité $(1, 1, 1)$ de la première ligne est $(B, C) = O$

• La formule $\rceil A \vee \rceil B \vee \rceil C$ ne prend la valeur 0 que pour cette distribution $(1, 1, 1)$ de cette première ligne.

• De même, en ligne 5, la distribution est $(B, C) = (0, 1, 1)$ mais qui donne encore $\mathcal{V}(F) = 0$ et la formule $A \vee \rceil B \vee \rceil C$ ne prend la valeur 0 que pour la même distribution $(0, 1, 1)$.

• Enfin la septième ligne correspond à la distribution $(B, C) = (0, 0, 1)$ et a pour résultat 0 ; la formule $A \vee B \vee \rceil C$ ne prend la valeur 0 que pour cette seule même distribution de vérité $(0, 0, 1)$.

La **Conjonction** notée **FNC(F)** $\quad (\rceil A \vee \rceil B \vee \rceil C) \rightarrow (A \vee \rceil B \vee \rceil C) \rightarrow (A \vee B \vee \rceil C)$ est donc logiquement équivalente à F.

Un **Littéral** est, par définition, une variable propositionnelle ou sa négation.
On dit que **FNC(F)** est une **FORME NORMALE CONJONCTIVE** pour **F** ; c'est une conjonction de **DISJONCTIONS** de **LITTÉRAUX**.

Théorème 3.1 (Complétude fonctionnelle par FNC)
Toute forme propositionnelle F *est équivalente à une formule qui est conjonction de disjonctions de litteraux, et est dite* ***Forme Normale Conjonctive*** *de* F.

B) Formes Normales Disjonctives :

De même, si l'on considère maintenant des distributions de vérité correspondant aux lignes où $\mathcal{V}(F) = 1$ et si l'on établit la correspondance

$$\begin{array}{llll}
\mathcal{V}(A,B,C) = (1,1,0) & \text{avec} & A \wedge B \wedge \rceil C & \text{notée } F_2 \\
\mathcal{V}(A,B,C) = (1,0,1) & \text{avec} & A \wedge \rceil B \wedge C & \text{notée } F_3 \\
\mathcal{V}(A,B,C) = (1,0,0) & \text{avec} & A \wedge \rceil B \wedge \rceil C & \text{notée } F_4 \\
\mathcal{V}(A,B,C) = (0,1,1) & \text{avec} & \rceil A \wedge B \wedge C & \text{notée } F_6 \\
\mathcal{V}(A,B,C) = (1,1,1) & \textit{avec} & A \wedge B \wedge C & \text{notée } F_8
\end{array}$$

alors F est logiquement équivalent à la formule **FND(F)** suivante :

$$F_2 \vee F_3 \vee F_4 \vee F_6 \vee F_8$$

qui est une disjonction de conjonctions de littéraux encore dite **FORME NORMALE DISJONCTIVE** de **F**.

Théorème 3.2 (Complétude fonctionnelle par FND)
Toute forme propositionnelle F *est équivalente à une formule qui est disjonction de conjonctions de littéraux et est dite* ***Forme Normale Disjonctive*** *de* F.

3.2 Compacité propositionnelle

Soit $F(A_1, \ldots, A_n)$ une forme propositionnelle dont l'ensemble **VAR(F)** des variables propositionnelles estÊ $\{A_1, \ldots, A_n\}$, une distribution de vérité
$\mathcal{V}(A_1, \ldots, A_n) \in \{0,1\}^n$ est dite **MODÈLE** de F si $\mathcal{V}(F) = 1$ pour cette distribution.

Un **MODÈLE** d'un ensemble Σ (fini ou infini) de formes propositionnelles est une distribution de vérité modèle de chaque formule de Σ.

On dit que F est **CONSÉQUENCE** de Σ si toute distribution qui satisfait Σ satisfait aussi F ; on note $\Sigma \models F$.

Un ensemble de formules Σ est contradictoire s'il n'a pas de modèle.

Théorème 3.3 *Soit* Σ *un ensemble de formes propositionnelles et* F *une formule ; alors* $\Sigma \models$ F *si et seulement si* $\Sigma \cup \{\neg \mathrm{F}\}$ *est contradictoire.*

Théorème 3.4 de Compacité
Un ensemble Σ *de formules du calcul propositionnel est contradictoire si et seulement s'il admet un sous-ensemble* ***fini*** *contradictoire.*

3.3 La méthode de résolution

On va définir une nouvelle notion de preuve (la troisième de ce chapitre) fondée sur un objet dit **ARBRE** de **RÉSOLUTION** dont les noeuds sont des **CLAUSES**. On doit donc d'abord définir les clauses et montrer comment transformer une **forme propositionnelle** (vue comme arbre) en un ensemble fini de **clauses** de longueurs ≤ 3.

Une **CLAUSE** est une disjonction de littéraux ; sa longueur est le nombre de littéraux qui y figurent.

On veut associer à chaque formule F un ensemble $C(F)$ de **CLAUSES** de longueur ≤ 3 tel que : F est **CONTRADICTOIRE** si et seulement si $C(F)$ est **CONTRADICTOIRE**.

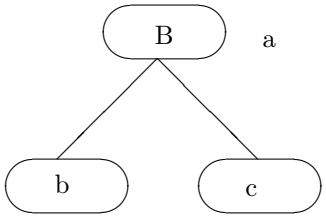
Pour ce faire, on transforme l'arbre F de façon à ce que *chaque noeud* a *reçoive un label* p(a) *qui soit une variable propositionnelle* et qui vérifie :

si a est une feuille, $p(a) = a$;

si a n'est ni la racine, ni une feuille (on dit que a est un noeud **INTERNE**)
alors $p(a) \in$ VAR(F).

Si a et b sont des noeuds internes $p(a) \neq p(b)$.

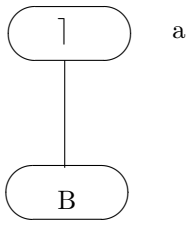
Les clauses élémentaires sont construites pour que si B est un connecteur binaire et si on a la configuration suivante :



alors : $\quad p(a) \leftrightarrow (p(b)Bp(c))$.

La conjonction des disjonctions constituant $C(F)$ soit logiquement équivalente à $[p(a) \leftrightarrow (p(b)Bp(c))]$.

De même la clause correspondant à :

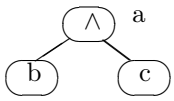
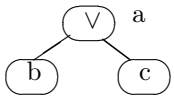
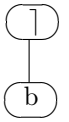
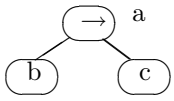
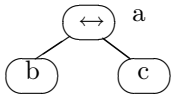


est logiquement équivalente à la formule $[\rceil p(a) \leftrightarrow p(b)]$.

Soit F une formule alors

$$\subset (F) = p(\text{racine de } F)\ U \bigcup_{a \text{ noeud interne}} \subset_a (F)$$

Définition des CLAUSES ÉLÉMENTAIRES

Si F est	alors $C(F)$ est
	$\{\rceil p(a) \vee p(b)$ $\rceil p(a) \vee p(c)$ $\rceil p(b) \vee \rceil p(c) \vee p(a)\}$
	$\{\rceil p(a) \vee p(b) \vee p(c)$ $\rceil p(b) \vee p(a)$ $\rceil p(c) \vee p(a)\}$
	$\{p(b) \vee p(a)$ $\rceil p(a) \vee \rceil p(b)\}$
	$\{\rceil p(a) \vee \rceil p(b) \vee p(c)$ $p(a) \vee p(b)$ $p(a) \vee \rceil p(c)\}$
	$\{\rceil p(a) \vee \rceil p(b) \vee p(c)$ $\rceil p(a) \vee p(b) \vee \rceil p(c)$ $p(a) \vee \rceil p(b) \vee \rceil p(c)$ $p(a) \vee p(b) \vee p(c)\}$

Remarque : Soit F est une formule considérée comme un arbre dont il est la racine. On constate alors que, pour toute distribution de vérité σ, si toutes les clauses $C_a(F)$ sont satisfaites *alors* les valeurs des variables $p(a)$ sont celles qu'on trouverait au cours des étapes du calcul de la valeur de vérité de F.

Les clauses seront notées, par convention sans expliciter les disjonctions et sous forme ensembliste :

$$\begin{array}{lll} \{\urcorner p(a), p(b)\} & \text{pour} & \urcorner p(a) \vee p(b) \\ \{\urcorner p(a), p(b), p(c)\} & \text{pour} & \urcorner p(a) \vee p(b) \vee p(c) \\ \text{etc.} & & \end{array}$$

Si u est un littéral alors $\urcorner u$ sera noté $\bar{u}$.

La **CLAUSE VIDE** sera notée $\perp$.

On dira que la **CLAUSE** c **RÉSOUD** les clauses c_1 et c_2 (ou encore que c est une **RÉSOLVANTE** de c_1 et c_2, noté $\mathrm{RES}(c_1, c_2)$) s'il existe un littéral u tel que :

le littéral u appartient à c_1

le littéral $\bar{u}$ appartient à c_2

$c = \{c_1/\{u\}\} \cup \{c_2/\{\bar{u}\}\}$

c'est-à-dire que c est la réunion de c_1 privé de u avec c_2 privé de $\bar{u}$.

Comme dans la méthode de **MODUS PONENS**, une ***preuve formelle*** par ***résolution*** de la clause c_n à ***partir*** d'un ensemble C de **CLAUSES** est une suite finie $(c_1, c_2, \ldots, c_n)$ telle que, pour tout c_i :

ou bien $c_i \in C$;

ou bien il existe $j < i$ et $k < i$ tels que $c_i = \mathrm{RES}(c_j, c_k)$.

On note $C \overset{\vdash}{}_R c_n$.

On dit que c_n est le **BUT** ou la **CONCLUSION** de la **PREUVE**. Si $c_n = \perp$, on dit que la preuve est une **RÉFUTATION** de C.

Comme dans le cas des arbres de Beth, la preuve à partir de C pourra se faire au moyen d'un arbre dit **ARBRE de RÉSOLUTION** R ainsi défini :

si $c_n \in C$, alors R est réduit à c_n ;

si $c_i = \mathrm{RES}(c_j, c_k)$ alors l'arbre de racine c_i possède c_j comme sous arbre gauche et c_k comme sous arbre droit ;

les feuilles de R sont étiquetées par les éléments de C.

Algorithme de résolution : pour prouver F, on va :

1) considérer $\urcorner F$ (de même qu'on considère $\bar{F}$ ou qu'on nie F dans la méthode de Beth) ;

2) mettre $\urcorner F$ sous forme normale conjonctive (qui donne un ensemble de clauses C noté $\mathrm{FNC}(\urcorner F)$) ;

3) construire l'arbre de résolution R dont les feuilles appartiennent à C et vérifie que la racine de C est $\perp$.

Proposition 3.1 *Si* $\mathrm{C} \overset{\vdash}{}_{\mathrm{R}} \mathrm{c_n}$ *alors* $\mathrm{C} \models \mathrm{c_n}$. *(*$\mathrm{c_n}$ *est conséquence de* C*).*

Proposition 3.2 $\{\mathrm{c} \backslash \{\bar{\mathrm{U}}\} (\mathrm{c} \in \mathrm{C}$ *et* $\not\in \mathrm{c}\} \models \mathrm{C}$.

Théorème 3.5 (Complétude pour la résolution)
Sont équivalentes les conditions suivantes :

(i) $\overset{\vdash}{}_{\mathrm{R}}$ *(la formule* F *est un théorème prouvé par résolution) ;*

(ii) il existe un arbre de résolution qui réfute FNC(⌝ F) *;*

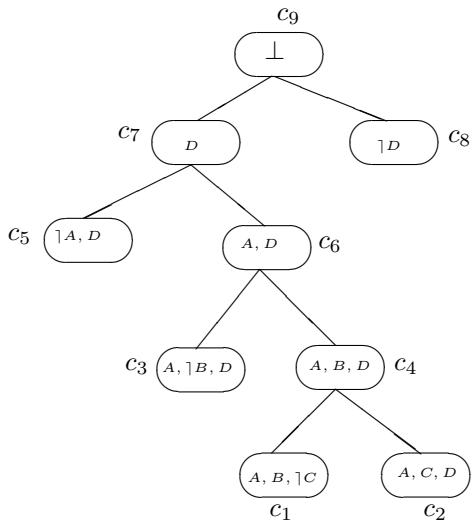
(iii) FNC(⌝F) *est contradictoire ;*

(iv) pour tout modèle σ*, on aura* $\sigma \models$ F *(la preuve utilise le théorème de compacité).*

Exemple de réfutation :

Soit FNC($\rceil F$) $= (A \vee B \vee \rceil C) \wedge (A \vee C \vee D) \wedge (A \vee \rceil B \vee D) \wedge (\rceil A \vee D) \wedge \rceil D$.

Arbre de réfutation



Preuve formelle :

$c_1 = \{A, B, \rceil C\}$ $\quad$ $c_2 = \{A, C, D\}$
$c_3 = \{A, \rceil B, D\}$ $\quad$ $c_4 = RES(c_1, c_2) = \{A, B, D\}$
$c_5 = \{\rceil A, D\}$ $\quad$ $c_6 = RES(c_3, c_4) = \{A, D\}$
$c_7 = RES(c_5, c_6) = \{D\}$ $\quad$ $c_8 = \{\rceil D\}$
$c_9 = \perp = RES(c_7, c_8)$.

RÉSOLUTION LINÉAIRE :

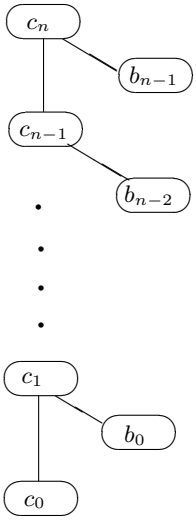
- On dit qu'une **PREUVE FORMELLE** obtenue par **RÉSOLUTION** à partir d'un ensemble C de clauses est **LINÉAIRE** si et seulement si cette preuve s'écrit $(c_0, c_1, \ldots, c_n)$ avec : $c_0 \in C$; pour tout i vérifiant $1 \leq i \leq n$, la clause c_i est résolvante de c_{i-1} et de $b_{i-1} \in C \cup \{c_0, \ldots, c_{i-1}\}$.

- On écrit $C \underset{RL}{\vdash} c_n$ dans ce cas.

Théorème 3.6 ***La méthode*** *de* ***résolution linéaie*** *est* ***complète****, c'est-à-dire, pour toute forme propositionnelle* $\underset{\text{RL}}{\vdash}$ F *si et seulement si, pour toute distribution de vérité* σ*, alors* $\sigma \models$ F.

- Si une **PREUVE DE RÉSOLUTION LINÉAIRE** $(c_0, \ldots, c_n)$ satisfait
  $c_0 \in C$,
  $b_{i-1} \in C$, alors on dit qu'elle est à **ENTRÉES DIRECTES**.

Son arbre de résolution se présente ainsi :



Remarque : La méthode de **résolution linéaire à entrées directes** (en abrégé RLD) n'est pas **complète**.

Clauses de HORN

Cependant il existe des ensembles de formules tels que la résolution RLD permette de caractériser les tautologies qui y sont incluses (complétude relative). C'est le cas pour l'ensemble des **Clauses** de HORN.

- Une **Clause de HORN** est une clause dont au plus un littéral est une variable propositionnelle, les autres littéraux sont tous des négations de variables propositionnelles.

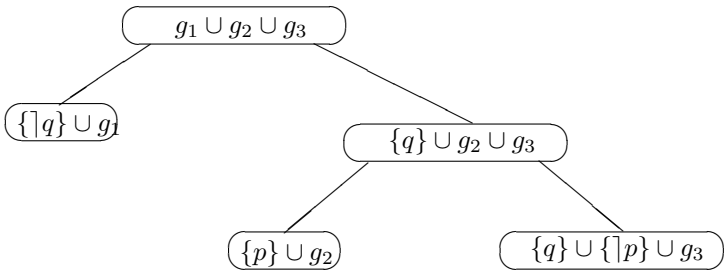
- Une **Clause de HORN** est définie dans le cas où exactement un littéral de cette clause est une variable propositionnelle.

- Si une **Clause de HORN** ne contient pas de variable propositionnelle, on dit qu'elle est **NÉGATIVE** ou qu'elle est un **BUT**.

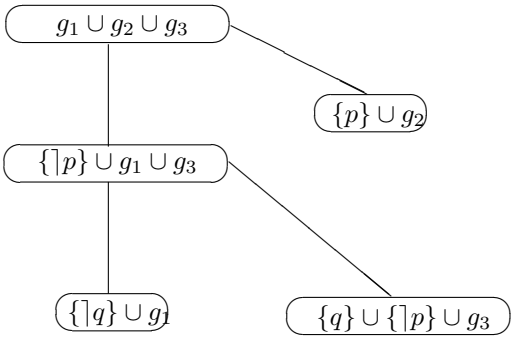
Théorème 3.7 *Soit* C *un ensemble de clauses de* HORN *et soit* C_0 *le sous ensemble formé des clauses définies, alors :*
c_0 *est non contradictoire,*
si C *est contradictoire, il existe une clause négative* $g \in C$ *telle que* $c_0 \cup \{g\}$ *soit contradictoire.*

Théorème 3.8 final *Soit* C *un ensemble de clauses de* HORN *définies, soit* g *une clause négative telle que* $C \cup \{g\}$ *soit contradictoire ; alors il existe une réfutation de* $c \cup \{g\}$ *par* RLD*-résolution.*

Exemple de linéarisation d'une réfutation :



résolution (non RLD)



Linéarisation (en RLD) de la résolution précédente