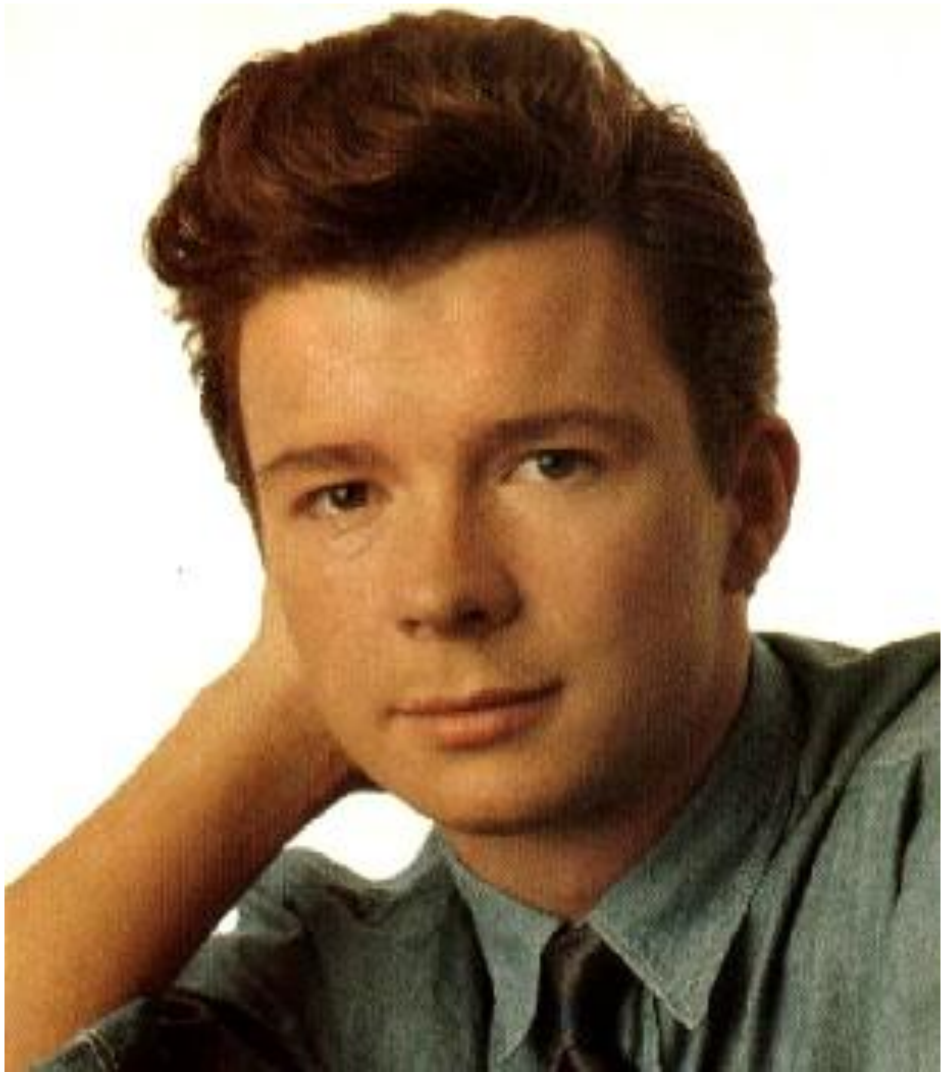
Logique

Sommaire



1-Introduction

Bibliographie

Logique :

- Systèmes formels, Claude Benzarken – Service logique Mathématiques
- Logique pour l'IA, Jean Paul Delahaye

Prolog :

- Arbre de Prolog, Shapiro , Sterling
- Wiki : Portail Logique

Langages

3 langages étudiés :

- Langage des propositions
- Langage des prédicats
- Prolog (seul langage de programmation)

2- Langage de Proposition

Première couche : couche d'accroche,

Buts :

- Etude de la Modélisation du raisonnement
- Analyse d'argumentations simples
- Utilisation de systèmes formels de démonstration

2.1) La Syntaxe

La proposition est la construction de base du langage, commen en francais, il faut que les propositions soient connectées pour que le texte ait du sens.

Il faut des liens entre les proposition élémentaires et les proposition atomiques .

Toute proposition est bâtie à partir de proposition élémentaires par des connecteurs.

2.1.1) Les connecteurs usuels

On utilisera :

- La négation
- La conjonction
- La disjonction
- L'implication
- (l'équivalence)

Idée de	Usage	Notation
Négation	Non	¬
Conjonction	Et	^
Disjonction	Ou	v
Implication	Si… alors	=>

Remarque :

Dans la logique vue içi, le temps n’est pas pris en compte (I .e le pouvoir d’expression des langages des proposition est largement moindre que celui du français)

Le et parfois peut signifier « puis »
La logique temporelle n’est pas vue ici.

Exo 1

- A =>B
- A=> B
- A=>B
- A=>B
- A=>B

2.1.2) Proposition élémentaires (atomiques)

Ce sont des énoncés (des affirmations) sans connecteurs pouvant se revéler être vraies ou faux.

Remarque :

Dans ce cours on ne peut modéliser des affirmartions vraies à 20%, on trabaille dans une logique à 2 valeurs Vrai/Faux, 1/0.

Les logiques floues permettent de considérer 1 infinité de valeurs : [0,1]

2.1.3) Grammaire des propositions

<prop> ::= <prop_atom> | <prop_comp>

<prop_comp> ::=¬<prop>|<prop><connect_bin><prop>

<connect_bin> ::= => | ^ | v

<prop_atom> ::= ident en minuscule

Priorité des Connecteurs :

Priorité décroissante :

$$\neg$$
$$\wedge$$
$$\vee$$
$$=>$$

Remarque :

Toute proposition est construite à partir d’un nombre fini de prop_atom. (Si manque d’imagination on uttilise un suffixe pour des idents)

$$i_n \, , n \in \mathbb{N}$$

Le langage défini par la grammaire est noté $\mathcal{P}$ (p onde).

Il arrivera que l’on réduise le nombre de connecteurs. Dans ce cas on mettra en indice les connecteurs retenus.

$$\mathcal{P}(\neg,1)$$

Exemple :

A

P1

(¬ a)

(¬ a)=> (b v (c ^d)) $\in \mathcal{P}$

A => B $\in \mathcal{P}$

Exo 2

H1 : $((\neg a)\hat{}r)v(n\hat{}(\neg c)))$

H1 : $((\neg a) => r)v(n => (\neg c)))$

H2 : $(\neg c) \vee (\neg a))$

H3 : $(l => ((\neg c) => \neg n \wedge a$

Exo 3

1)

$$r \wedge y => \neg q$$
$$\neg b => q \vee g$$
$$c \wedge \neg e => \neg(g \wedge q)$$
$$(g \vee q) \wedge \neg c => r \vee b$$

2)

$$b \vee r \vee e$$

Exo 4

V : a commis le vol	**P1 :** $v => p \vee c$
P : bien préparé	**P2 :** $p => (c => b)$
C : complice	**P3 :** $\neg b$
B : butin important	

On ne sait pas dire pour l'instant si cet argument est convainquant.

Par contre elle sera convaincante ou non indépendamment de la signification des propositions.
Elle est syntaxiquement convaincante ou non.

2.2) Calculer avec des propositions.

2.2.2) Table de vérités

Construction de l'application v

Par récurrence sur la longueur des $P \in \wp$

1) P ∈ A,

   Dans ce cas, v(p) ≡val(p) : c'est le noyau de l'extension.

2)
   a) P≡¬Q
      Q est plus courte que P (1 connecteur en moins)
      On a v(Q) (application de la récurrence)

$$P = a \wedge b$$

V(Q)	V(P)
0	1
1	0

   b) $P \equiv Q < cb > R$ où cb est un connecteur binaire, Q & R sont plus courts que P, On connait v(Q) et v(R)

$$On\ déduit\ v(P) de\ v(Q) et\ v(R)$$

V(Q)	V(R)	V(P=Q∧R)	V(P=QVR)	V(P=Q=>R)
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	1

Exercice 5

a	b	B=a∧b	C=B=>¬b	V(D=b=>C)	A
0	0	0	1	1	1
0	1	0	1	1	1
1	0	0	1	1	1
1	1	1	0	0	0

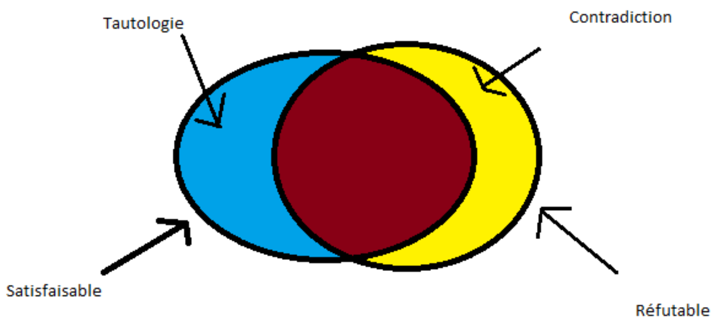
2.2.3) Définitions

Si P est tautologie :

-Toute valuation est réalisation

-∀v, v(P)=1

- La colonne principale de la table de vérité ne comporte que des 1.



Remarque : 2.11

Deux propositions atomiques ne sont jamais sémantiquement équivalentes.

Deux propositions A & B quelconques ne contenant pas exactement les mêmes propositions atomiques peuvent être sémantiquement équivalentes.

a∧(b=>b) et (a∧c) v (a ∧ ¬c) sont sémantiquement équivalentes.

Remarques

1- Deux ensembles contradictoires sont équivalents :

   Ils ne sont pas satisfaisables, donc ils n'ont pas de réalisation.

   ➔ Ils ont bien les mêmes réalisations

2- Deux ensembles satisfaisables ne sont pas nécessairement équivalents sémantiquement.

3- Deux ensembles S et T réduits à un élément.

   ➔ On retrouve bien la définition de propositions équivalentes.

Cas Particulier Important

R={A1,…,An}   et   S= {A1∧…∧An)

R & S Sont sémantiquement équivalents.

- ∀v, v est réalisation de R si v est réalisation de chacunes des propositions de P, ∀i, v(Ai)=1
- ∀v, v est réalisation de A1∧…∧An, si ∀i v(Ai)=1 (d'après la table de vérité de ∧) .
  Donc R et S ont les mêmes réalisations.

Equivalences remarquables

On rajoute les constantes

- vrai telles que ∀v, v(vrai)=1
- faux telles que ∀v, v(faux)=0

Les équivalences ne sont pas vraies dans toutes les logiques

Remarques

- Une tautologie peut être sémantiquement équivalente à la constante vrai
- Une contradiction peut être sémantiquement équivalente à la constante faux.
- Une même réalité peut s'exprimer de nombreuses façons équivalentes.
- On note que l'ensemble des connecteurs choisis pour P est luxueux.

Equivalences :

- Principe de Dualité : Les équivalences remarquables font ressortir des réécritures symétriques entre les connecteurs ∨ et ∧ :
- Soient A et B ∈ P(¬,∨ , ∧)
- Soit A' ∈ P(¬,∨ , ∧) réécrite a partir de A en inversant les ∨ et les ∧.
- Soit B' ∈ P(¬,∨ , ∧) réécrite a partir de B en inversant les ∨ et les ∧.
- Si A ≈ B alors A' ≈ B'.

Ex7

P=r=> (s=> r)

1 Table de vérité

2 Par équivalence remarquable

$$\neg r \vee (\neg s \vee r)$$
$$\neg r \vee (r \vee \neg s)$$
$$(\neg r \vee r) \vee \neg s$$
$$vrai \ \vee \neg s$$
vrai

Exercice 8

P1	P2	P3	P4	A	P3∧ ¬$P4$	(P3∧ ¬$P4$) => A	C
1	1	1	1	1	0	1	1
1	1	1	0	1	1	1	1
1	1	0	1	1	0	1	1
1	1	0	0	1	0	1	1
1	0	1	1	0	0	1	0
1	0	1	0	1	1	1	0
1	0	0	1	0	0	1	0
1	0	0	0	0	0	1	0
0	1	1	1	0	0	1	0
0	1	1	0	1	1	1	0
0	1	0	1	0	0	1	0
0	1	0	0	0	0	1	0
0	0	1	1	0	0	1	0
0	0	1	0	1	1	1	0
0	0	0	1	0	0	1	0
0	0	0	0	0	0	1	0

P1	P2	C
1	1	0
1	0	1
0	1	1
1	1	1

C= ¬(P1∧P2)

2.2.6) Formes Normales

Idée de construction de A1

X=>Y -> ¬X ∨Y

X ∧ Y → $\neg\neg(X \wedge \neg Y) \rightarrow \neg(\neg X \vee \neg Y)$

Idée de construction de A2

X=> Y → $\neg(X \wedge \neg Y)$

$X \vee Y \rightarrow \neg(\neg X \wedge \neg Y)$

Idée de construction de A3

$X \vee Y \rightarrow \neg X => Y$

$X \wedge Y \rightarrow \neg(X => \neg Y)$

Idée de construction de A4

2 méthodes

- Syntaxique -> identité remarquables (tp 8 prolog)
- Sémantique A -> table de vérité de A -> A4
- Supposons que A comporte n propositions atomiques et on suppose que la table de vérité de A comporte p réfutations (v1,…,vp)

1. Soit vj une des réfutations de A. On construit une clause Cvj de n littéraux
   Cvj = l1 v … v ln
   Question : comment choisir les littéraux li, i ∈ [1,n] de telle sorte que pour cette valuation v est réfutation de Cvj et que pour toute autre valuation soit realisation de Cvj ?
   Exemple : p1-> 1,p2 -> 0, p3 ->1, p4 -> 1
   C1= ¬P1vP2v¬P3v¬P4
   V1(C1)=0
   Vi(Ci)=1 pour i≠ 1
   Réponse :
   Li=pi si v(pi) = 0
   Li = ¬ pi si v(pi) = 1
   2- A4=∧Cj

Idée de preuve de la propriété

1. Soit H={H1,…,Hn} où Hi sont des propositions de P.
2. $\forall Hi \in H, \exists Ai \in P(\neg, \vee, \wedge)$telle que Ai $\approx$ Hi et Ai = Ci1∧…∧Cin
   $On\ a\ bien$
   $Ai \approx \{\mathrm{Ci1} \wedge \ldots \wedge \mathrm{Cin}\}$
   et d'après"cas particuliers importantes"
   $Ai \approx \{\mathrm{Ci1}, \ldots, \mathrm{Cin}\} = \mathrm{Ai}$
3. Finalement
   $$H' = \bigcup_{j=1}^{m} Aj$$
   ➔ Preuve du théorème de Godel

Théorème de Complétude de Godel.(calcul des prédicats du premier ordre)

Le calcul des prédicats est complet au sens où toute proposition qui est vraie dans ce calcul peut être démontrée.

2.3) Sémantique de la déduction

2.3.1) Conséquence Valide

Si H |= A alors si une des propositions de H est a 0, alors on ne regarde même pas A.

2.3.2) Théorème sémantique de la déduction

Les tautologies sont conséquences valides de tout ensemble de proposition.

Cas limite de la définition de conséquence valide

Cas particulier H=∅,

Toute valuation est alors réalisation de H (on peut écrire que toute valuation est réfutation de H) Non contradictoire car H est vide.

Donc ∀A∈H, v(A)=1 (on pourrait aussi écrire v(A)=0.

⚠ Si on prend la négation de cette formule ∃ A∈H, v(A)=0, celle-ci est fausse.

On note alors |=A'. (A' est une tautologie).

Exercice 9(Cf Exo4 Question 2)

$$H | = a?$$

Démos manières de répondre :

1. On construit la table de vérité et on conclut.
2. On est convaincu que la réponse est négative, d'après la déf des conséquences valides, il suffit d'exhiber une valuation v telle que v(H)=1 et v(A)=0

Trouver v tel que :
V(¬ a)=0

V(a => (b ⋁ c))=1 ; v(b=>(c=>d))=1 et v (¬d)=1

Exemple contredisant a-> 1 b-> 1 c-> 0 d-> 0, A=0,H=1

Démonstration du théorème 2.3
1.Sens ->

Supposons β |=A, d'après la définition de conséquence valide .

∀ v, v est réalisation de β, v est alors réalisation de A, on a v(A)=1.

Soit v une réalisation quelconque de β – {B}. On veut montrer que v est réalisation de B=> A.

Deux cas se présentent (v(B)=0 ou v(B)=1)

a- V(B)=0 alors v(B=>A)=1 (Cf table de vérité de =>)
   Dans ce cas on à bien β – {B} |= B=> A
b- V(B)=1 alors v(β)=1 (car v(β – B)=1)
   Donc v(A)=1 (car on a B|= A, cf définition de |=)
   On a bien v(B=>A)=1 (Cf table de vérité de =>)
   Dans ce cas on à bien β – B |= B=>A

2.Sens <-

Supposons que β –{B} |=B=>A, ∀v réalisation de β – B, alors v est réalisation de B=>A, v(B=>A)=1

Soit v une réalisation quelconque de B. On veut montrer que v est réalisation de A, v(A)=1

Donc v(β – {B})=1 car β – B C β

On a v(B=>A)=1 (def de |=)

Et v(B)=1 car B ∈ β

Donc v(A)=1 (déf de =>)

Donc v qui est réalisation de β est aussi réalisation de A

B|= A

Utilité du Théorème

Corollaire 1 :
C est conséquence valide de H={ H_1, … ,H_n} ssi H_n => (H_{n-1}=> … =>(H_1=>C))

Ie H_1, … ,H_n |= C

Application du Théorème
H_1, … ,H_n |=C   ssi H_2, … ,H_n |= H_1 => C

ssi H_n => (H_{n-1}=> … =>(H_1=>C))

Corollaire 2 Réduction à l'absurde
Que dire de H quand H|= faux ?

ie toute réalisation de H est réalisation de faux

Or, la constante faux n'a aucune réalisation donc H n'a aucune réalisation, alors H est contradictoire.

Application du Théorème
H= β U { ¬ A} avec H |= faux (β U { ¬ A} est contradictoire)

β U { ¬ A} ≠ faux   ssi β |=> A => faux ssi β |=> A

Principe de démonstration par l'absurde

Ayant a démontrer A à partir de H, on peut montrer qu'ajouter la négation de A à H aboutit à une contradiction.

Remarque : Principe de base de Prolog.

Exercice 10

1. Démarche

a	c	n	r	H_1	H_2	H_1,H_2
1	1	1	1			
1	1	1	0		1	
1	1	0	1			
1	1	0	0		1	
1	0	1	1	1	1	1
1	0	1	0	1	1	1
1	0	0	1		1	
1	0	0	0		1	
0	1	1	1	1		
0	1	1	0		1	
0	1	0	1	1		
0	1	0	0		1	
0	0	1	1	1	1	1
0	0	1	0	1	1	1
0	0	0	1	1	1	1
0	0	0	0		1	

H_1,H_2 |= ¬ C

2.4) Systèmes formels

Remarque :

Un théorème es tun objet purement symbolique

Il est très facile de construire un système formel mais beaucoup plus délicat de construire un SF interressant

Remarques :

- Le choix de P{ ¬, =>} n'est pas pénalisant puisque nous avons vu que toute proposition de P possède une proposition sémantiquement équivalent dans P{ ¬, =>}.
- Les propositions A,B,C appartiennent à P{ ¬, =>} Soit 3 infinites
- La règle d'inference indique que si A est un énoncé déjà produit si A=> B est un énoncé produit alors SF_1 produit B.

Preuve : Par récurrence

- Cas de base
  Les axiomes sont ils des tautologies ?
  Facile à Montrer grâce aux tables de vérité
- Cas général
  A partir de deux tautologies, la règle d'inférence ne peut produire qu'une tautologie (A, A=> B)
  Supposons que A et A=> B soient des tautologie c'est-à-dire ∀ v, v(A)=1 et v(A=>B)=1
  Qu'en est il pour B ?

V(A)	V(B)	V(A=>B)
0	0	1
0	1	1
1	0	0
1	1	1

Donc B est une tautologie

Remarque :

- H = Ø on note que ⊢A et on dit que A est un théorème, un théorème est une proposition démontrable en n'utilisant que les axiomes et la règles d'inférence

Exercice 11

Rédiger la démonstration de p=>p dans le système formel SF_1 c'est-à-dire

$\vdash SF_1$ p=> p

P_1 : axiome

P_2 : axiome

P_3 : m p

P_4 : axiome

P_5 : m p

$\vdash SF_1$ p=> p

P_1 : axiome	C=>B :
P_2 : axiome	C
P_3 : m p	C, C=>B, B
P_4 : axiome	B=>A
P_5 : m p	p=>p = A

Avec C= (p=>((p=>p)=>p)

Et B = (p=>(p=>p))=>(p=>p))

P_1 : axiome	(p=>((p=>p)=>p)=> (p=>(p=>p))=>(p=>p)) :
P_2 : axiome	(p=>((p=>p)=>p)
P_3 : m p	(p=>(p=>p))=>(p=>p))
P_4 : axiome	p=> (p =>p)
P_5 : m p	p=>p = A

B ⊢ A ssi β – {B} ⊢ B => A

Preuve

<-

Supposons que β – {B} ⊢ B => A autrement dit

∃ $\mathbb{D}_{B-\{B]}$

P'_1

…

P'_i =B => A

La démonstration cherchée est alors

∃ $\mathbb{D}_{B-\{B]}$

P'_1

…

P'_i =B => A

P'_{i+1} =B car B ∈ β

P'_{i+2} =A mp avec $P'_{i+1 \text{ et}}$ P'_i

→ Sachant B ⊢ A il faut montrer que β – {B} ⊢ B => A

Dit autrement :

On le montre par récurrence sur la longueur de la démonstration D_B

n=1

D_B est réduite à $P_1 = A$

Il y a plusieurs possibilités

(a) A est un axiome alors

$\mathbb{D}_{B-\{B]}$	$P'_1 = A$ axiome $P'_2 = A \Rightarrow (B \Rightarrow A)$ axiome A_1 $P'_3 = B \Rightarrow A$ mp avec P'_1 et P'_2

(b) A ∈ β A ≠ B alors

$\mathbb{D}_{B-\{B]}$	$P'_1 = A$ hypothèse $P'_2 = A \Rightarrow (B \Rightarrow A)$ axiome A_1 $P'_3 = B \Rightarrow A$ mp avec P'_1 et P'_2

Dans ce cas particulier on a montré {A} ⊢ B => A

Donc a fortiori β – {B} ⊢ B => A

(c)

Hypothèse de récurrence

Supposons que pour toute démonstration D_B de A de longueur nLN, il existe une demonstration $\mathbb{D}_{B-\{B]}$ de B=> A

Montrons que si n=N il existe une démonstration de $D_{B-\{B\}}$ de B=> A

$\mathbb{D}_B$	P_1 $P_N = A$

Et on cherche

$\mathbb{D}_{B-\{B\}}$	P'_1
	$P'm = B \Rightarrow A$

Il y a plusieurs possibilité pour A

(a) A est un axiome

(b) A est une hypothèse

A ≠ B

A=B

(c) A est obtenue par mp

Autrement dit, dans D_B, $\exists\, P_j = X$, j<N et $\exists\, P_k = X \Rightarrow A$, k<N

On peut appliquer l'hypothèse de récurrence (car j et k < N)

$\exists\, \mathbb{D}_{B-\{B\}}$	P'_1 $P'_N = B \Rightarrow X$

$\exists\, \mathbb{D}_{B-\{B\}}$	P''_1 $P''_N = B \Rightarrow (X \Rightarrow A)$

Rappel on cherche à montrer B => A dans β – {B}

La démonstration cherchée : concaténation de ces deux démonstrations suivi de

$P'''_{u+v+1} = (B \Rightarrow (X \Rightarrow A))$ axiome A2

$P'''_{u+v+2} = (B \Rightarrow X) \Rightarrow (B \Rightarrow A)$ mp ou celui d'avant et P''_v

$P'''_{u+v+3} = B \Rightarrow A$ mp ou les deux precedents

Remarque ⊢p => p ssi théorème de la déduction {p} ⊢ p

Exercice 12

A=>B, B=> C ⊢ A => C

On peut appliquer le théorème de la déduction, et on se ramène à

A => B, B => C, A ⊢ C

$\mathbb{D}$	P_1 =A (hypothèse) P_2 = A=> B P_3 = B mp P1 et P2 P_4 = B => C P_5 = C mp sur P3, P4

Remarque :

SF_1 n'oublie aucune tautologie !

$$C = \bigvee_{i=1}^{n} C_i$$

Alors v(c) =1 ssi il existe au moins un litteral A_j dans la disjonction tel que v(A_j)=1

Dans la clause vide , il n'existe oas de telle proposition A_j, donc la clause vide ne possède aucune réalisation

Exemple de démonstration a ⋁ b ⋁ c, ¬ a ⋁ b ⋁ c, ¬ b ⋁ c ⊢ c

D P_1= a ⋁ b ⋁ c hypothèse
P_2= ¬ a ⋁ b ⋁ c hypothèse
P_3= b ⋁ c résolution
P_4 = ¬ b ⋁ c hypothèse
P_5= c résolution

Tout théorème de SF_R est une tautologie car il n'y a aucun théorème dans SF_R

Propriété 2

On ne peut pas démontrer que des conséquences valides.

Preuve : Par récurrence sur la longueur de la démonstration H ⊢ C

1- Démonstration de longueur lg = 1 ou 2 (pour appliquer la règle d'inférence il faut 2 hypothèses)
   C ∈ H (car pas d'axiome dans ce SF)
   Dans ce cas si ∀ v, on a v (H) = 1, alors v(c)=1 puisque c ∈ H
   Def de consequence valide H |= C
2. Hypothèse de récurrence
   Toute démonstration de longueur lg < N demontrent des conséquences valides
3. Démonstration de lg = N avec C ∉ H (sinon trivial)
   Dans ce cas C ne peut être que la résolvante de 2 clauses situées avant dans la démo

D P_1=
P_i= ¬ a ⋁ b
P_j= a ⋁ A
P_N = C = a ⋁ A

Les démonstrations de P_i et P_j sont de longueur < N et donc pas d'hypothèse de récurrence

---22 septembre : plus de batterie---

Exercice 13

Utilisation de l'algorithme 2.5.5

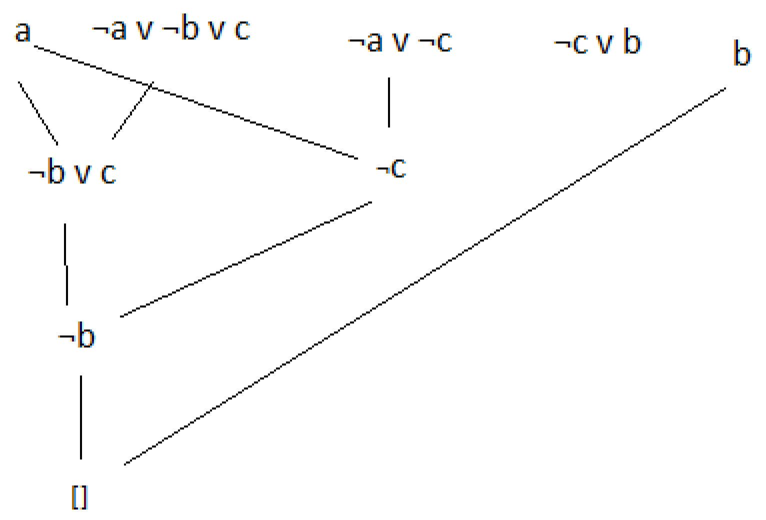
1. a ⋁ ¬ b ⋁ c
2. ¬ a ⋁ c
3. ¬ e

4. A V b V e
5. ¬ c V e
6. ¬ b V c (1 et 2)
7. a V c V e (1,3)
8. a V ¬ b V e (1 et 4)
9. b V c V e (2,3)
10. ¬ a V e (2,4)
11. ¬ c (4,5)
12. a V b (3,5)
13. b V e (9,11)
14. a V c V e (6,4)
15. C V e (14,2)
16. e(15,5)
17. ■(16,3) : clause vide : contradictoire

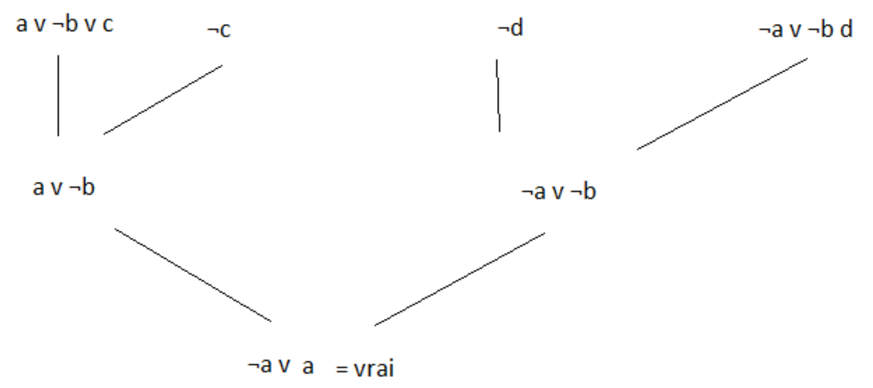
Exercice 14

Impossible ! car aucune clause ne contient ¬ d

Exercice 15



Exercice X



Exercice 16

H1 = (l ∧ t) ⇒ (m V w)

H2 = w ⇒ ((m ∧ ¬ t) V ¬ L)

H3 = m ∧ t ⇒ w

¬ C = ¬ (L ⇒ ¬ t)

H1= (¬ l ∧ t) V (m V w)

C1 = ¬L V ¬ t V m V w

H2= ¬ w V ((m ∧ ¬ t) V ¬ L)

= ¬ w V ¬ L V (m ∧ ¬ t)

= (¬ w V m V ¬ L) ∧ (¬ w V ¬ t V ¬ L)

C2=(¬ w ∨ m ∨ ¬ L)

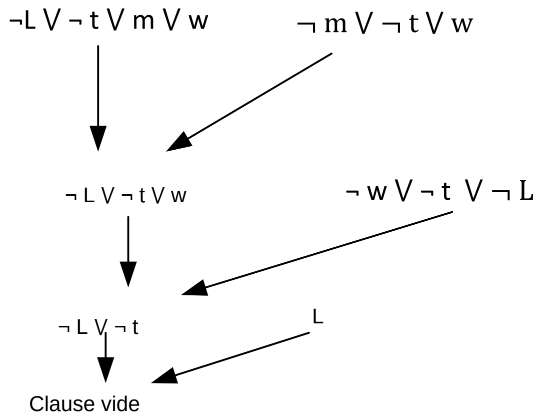
C2=(¬ w ∨ ¬ t ∨ ¬ L)

H3= ¬ (m ∧ t) ∨ w = ¬ m ∨ ¬ t ∨ w

C3 = ¬ m ∨ ¬ t ∨ w

¬ C = ¬ (¬ L ∨ ¬ t)

=l ∧ t



Exemples

$$\forall\, x\ (P(f(x)) \Rightarrow Q(g(x)) \wedge R(x))$$

fbf ?

$$P(a,f(a)) \vee (\ \forall\, y\ (R(y,b) \Rightarrow S(f(b)))$$

fbf ?

rouge : terme

vert : formes atomiques

bleu : fbf

3.1.3 Restictions

Exemples

P(x) ∨ R(x)	Interdit car x n'est pas quantifié
P(x) ∨ ∀ x(R(x))	Interdit : 1er x pas quantifié
P(y) ∨ ∀ x(R(x))	Interdit car y n'est pas quantifié

∀ x(P(x) ⇒ ∀ x R(x)) : interdit

∀ x(P(x) ⇒ ∀ y R(y)) : permis

Propriété

Ces restrictions simplifient l'étude de langage $\mathbb{L}$ sans pour autant la dénaturer

3.2.1 Interprétation

G= ∀ x(R(x) ⇒ T(f(x),x)

Fbf (formule bien formée)

Une interprétation

1. D= {4,5}
2. A → 4
3.
4. F → $\begin{cases} 4 \to 5 \\ 5 \to 4 \end{cases}$
5. R $\begin{cases} 4 \to 1 \\ 5 \to 0 \end{cases}$
6. F → $\begin{cases} 4,4 \to 1 \\ 4,5 \to 0 \\ 5,4 \to 1 \\ 5,5 \to 0 \end{cases}$

Une autre interprétation.

1. D={Jeanne, Marc, Valentine, Lucy}
2. a → Valentine
   f → frere
   R → est une femme
   T → plus agée

G = ∀ x (A(x) ⋁ B(x))

I1	I2
D = {true}	D= ℕ
A → {true → 1	A → pair
B → { true → 1	B → impair
I_1 est il modèle de G?	I_2 est il modèle de G?
oui	oui

G= ∀ x (A(x) ⋁ ¬ A(x)) est une tautologie, Sinon, on aurait un a pour lequel on a A(a) ∧ ¬ A(a)

(∃ x (¬ A(x) ∧ ¬ ¬ A(x)))

EX 17

G= ∃ x (P(a,x) ∧ ∀ y ∃ z(Q(x,f(y,z))⇒(Q(y,a) ⋁ Q(y,x)))

1. Oui, il est possible de trouve un arbre syntaxique de racine fbf et dont les feuilles sont des terminaux de G.

2.

   (a) D= ℕ
   (b) Q → la relation = : ℕX ℕ → {0,1}
   (c) P → la relation < : ℕX ℕ → {0,1}
   (d) a → 1
   (e) f → la fonction x : ℕ X ℕ → ℕ

   Il existe x > 1 tel que pour tout y, il existe un z de telle sorte que si x s'écrite comme étant le produit de y et de z, cela impose que y soit égal à x ou à 1.

   Autrement dit

   Il existe au moins un entier naturel x qui n'a comme diviseur que 1 et lui-même.

Autrement dit

X est un nombre premier

Il existe un nb premier > 1 donc I est modèle de G

Exercice 18

D={1,2} P : égale A → 1 B → 2 F : fonction identité	D = $\mathbb{N}$ P → $\leq$ A → 0 B → 0 F → la fonction identité	D = $\mathbb{N}$ P → $\leq$ A → 9 B → 17 F → la fonction successeur

3°)

$\{A_1, A_2, A_3\} \vDash \exists\, x\, P(x,a)$

I_1 ? $\exists\, x\ x=1$

I_2 ? $\exists\, x\ x \leq 0$

I_3 ? $\exists\, x\ x \leq 9$

Il faut un modèle de $\{A_1, A_2, A_3\}$ qui ne soit pas modèle de $\exists\, xP(x,a)$

I_4 :

D= $\mathbb{N}$
P → <
A → 0
B → 1
F → fonction successeur
I_4 est modèle de $\exists\, x <0$? oui

3.3.2 Mise sous forme prénexe

Démonstration
Par construction

Soit $F \in \mathbb{L}$

$1^{ère}$ étape : réécrire Fen remplacant la sous formule de la forme $A \Rightarrow B$ par $\neg A \lor B$, d'où $F_1 \approx F$.

$2^{ème}$ étape : réécrire F_1 en accolant le ¬ aux formules atomiques, on utilise :

$\neg\neg G \rightarrow G$
$\neg (A \lor B) \rightarrow \neg A \land \neg B$
$\neg (A \land B) \rightarrow \neg A \lor \neg B$
$\neg (\exists\, xA) \rightarrow \forall\, x \neg A$
$\neg (\forall\, x\, A) \rightarrow \exists\, x \neg A$
D'où $F_2 \approx F_1$

$3^{ème}$ étape : réécrire F_1 en amenant les quantificateurs en tête et en respectant leur ordre

D'où $F_3 \approx F_2$

<u>Au final F_3 est sous forme prénexe.</u>

Exercice 19
$F= \forall\, x\, (P(x) \land Q(x,a) \Rightarrow (R(x,b) \land \exists\, y\, \forall\, z(R(y,z) \Rightarrow T(x,z))))$

$F_1= \forall\, x\, (\neg (P(x) \land Q(x,a)) \lor (R(x,b) \land \exists\, y\, \forall\, z(\neg R(y,z) \lor T(x,z))))$

$F_2= \forall\, x\, ((\neg P(x) \lor \neg Q(x,a)) \lor (R(x,b) \land \exists\, y\, \forall\, z(\neg R(y,z) \lor T(x,z))))$

$F_3= \forall\, x\, \exists\, y\, \forall\, z\, (\dots)$

Exemples
$\forall\, x\, \exists\, y\, P(x,y)$

$\hookrightarrow \forall\, x\, P(x,f(x))$ f : nouveau nom de fonction

$\exists\, x\, \forall\, y\, Q(x,y,a)$

$\hookrightarrow \forall\, y\, P(b,y,a)$ b : nouveau nom de constante

$\forall\, x\, \forall\, y\, \exists\, z\, (P(x,y) \lor Q(f(y),g(z)))$

$\hookrightarrow \forall\, x\, \forall\, y\, (P(x,h(x,y)) \lor Q(f(y),g(h(x,y))))$ h : nouveau nom/symbole de fonction

$F = \forall x \exists y\, P(x,y)$ $F_1 = \forall x\, P(x,f(x))$

I : D= $\mathbb{Z}$ I est modèle de F mais pas de F_1

P→ la relation <

F → la fonction predeccesseur F : $\forall x \exists y\; x<y$ | $F_1 : \forall x\, (x<x-1)$

F et F_1 ne sont pas logiquement équivalentes VRAI Faux

3.3.3) Mise sous forme clausale

Exemple : {P(x), Q(x)}

↪ {P(x), Q(y)}

$F_3 = \{ P(x,y) \lor \neg Q(x) , \neg P(x,z) \lor R(x) , Q(x,y)\}$

↪ $F_4 = \{ P(x,y) \lor \neg Q(x), \neg P(u,z) \lor R(u), Q(v,w)\}$

Exo

$\exists y \forall z\, (P(z,y) \Leftrightarrow \neg \exists x(P(z,x) \land P(x,z))$

$\exists y \forall z(\neg P(z,y) \lor \forall x (\neg P(z,x) \lor \neg P(x,z))) \land (P(z,y) \lor \exists u(P(z,u) \land P(u,z))))$

$\exists y \forall z \exists u \forall x((\neg P(z,y) \lor \neg P(z,x) \lor \neg P(x,z)) \land (P(z,y) \lor(P(z,u) \land P(u,z))))$

3.3.5 Condition d'application de la règle

Remarques :

- R_1 et R_2 sont des formules atomiques ayant le même identificateur de prédicat
- La clause $D_1' \lor D_2'$ s'appelle la clause résolvante
- l_1 et $\neg l_2$ ne figurent pas forcément en tête de leur clauses respectives.

Exemple :

- $C=P(f(x,y),g(b,z))$ clause
- $S = \{<x,h(a)>,<u,b>\}$ substitution
- $C_3 = P(f(h(a),y),g(b,z))$

Exemple

Ces formules sont elles unifiables ? Si elles le sont, c'est qu'on a réussi a les égaliser via une substitution

$F=P(f(x,y),g(b,z))$

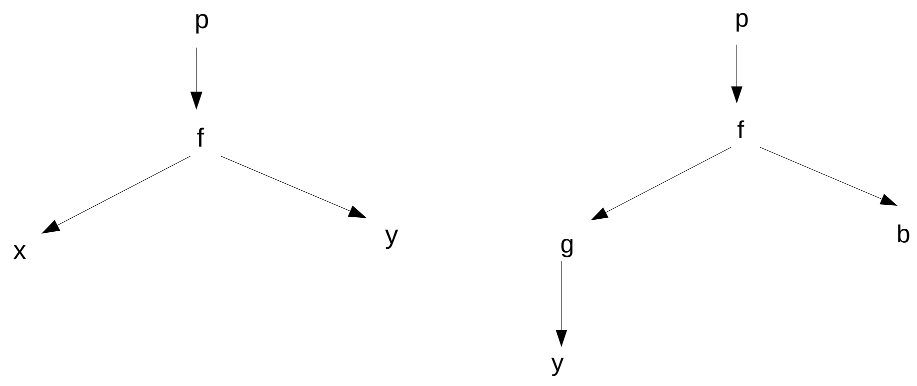
$G=P(f(h(a),y), g(b,z))$

Substitution : $S = \{ <x, h(a)> $

Ex 20

$F = P(f(x,y))$

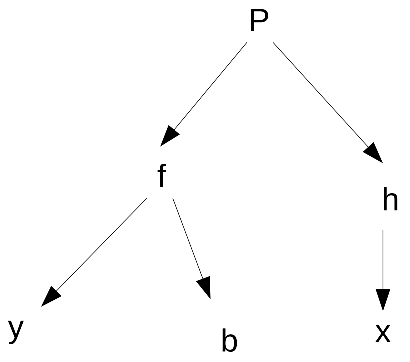
$G= P (f(g,y),b)$

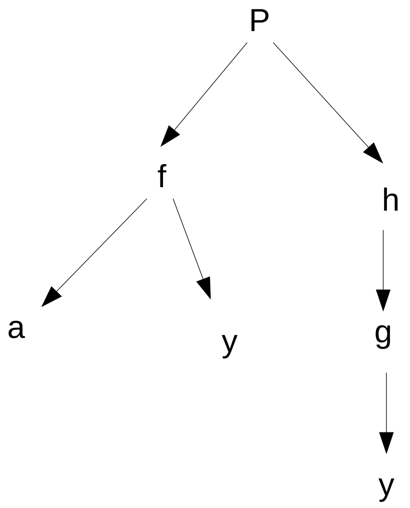


$S=\{<x , g(y)>,<y,b>\}$ n'est pas une substitution

$U = \{<x,g(b)>,<y,b>\}$ Ceci est une bonne substitution et un unificateur ✔

Ex 21





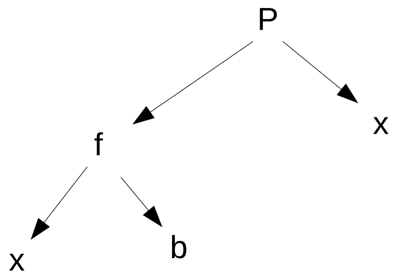
Impossible car x ne peut être a la fois a et g(y)

Ex

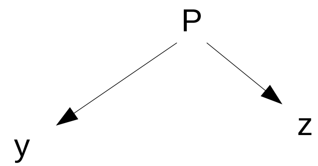
F = P(f(x,b),x)

G=P(y,z)

F)



G)



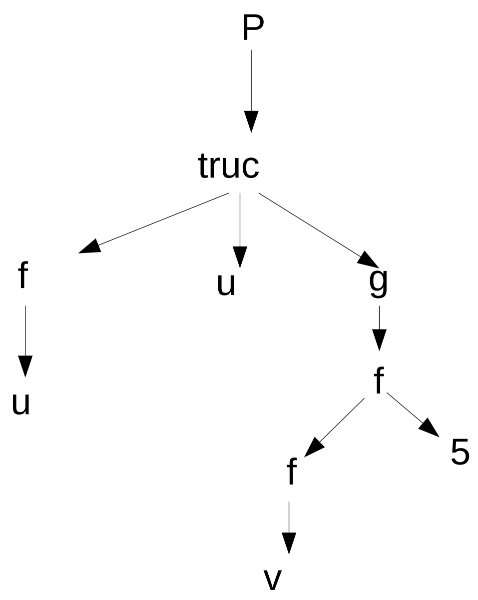
S={<y,f(x,b)>,<z,x>}

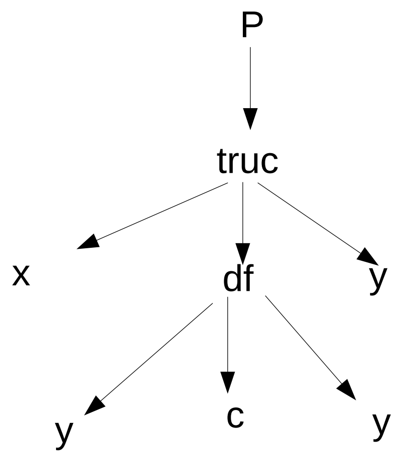
Ou

S={<y,f(z,b)>,<x,z>}

Exercice 22

1)





$$\begin{cases} X \rightarrow f(u) \\ u \rightarrow df(y,c,y) \\ y \rightarrow g(f(f(u),5)) \end{cases}$$

N’est pas une substitution valide car on retrouve des variables des deux cotés.

D’où

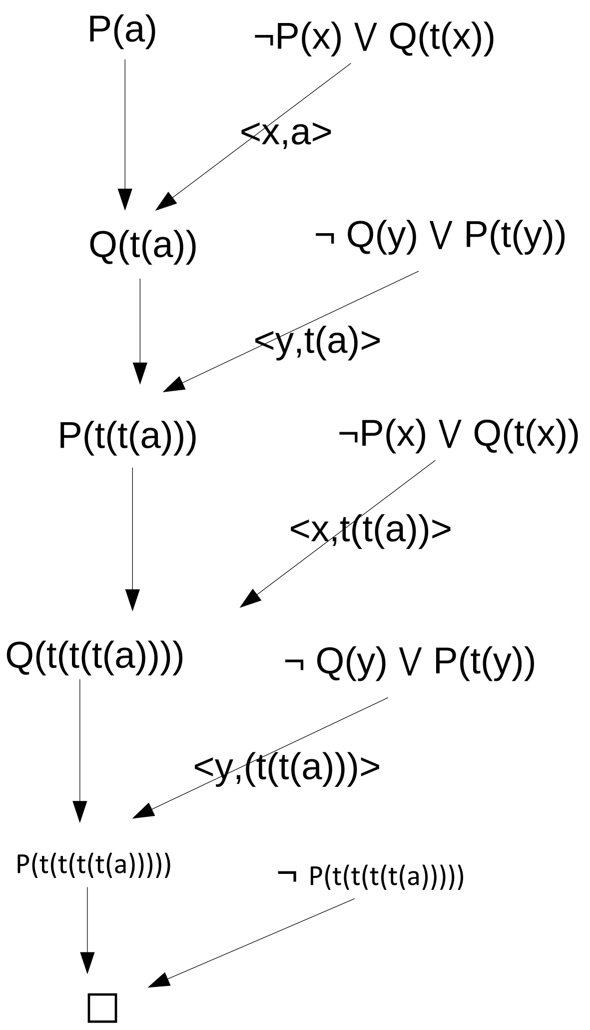
$$\begin{cases} X \rightarrow f(df(g\big(f(f(v),5)\big),c,g\big(f(f(v),5)\big)) \\ u \rightarrow df(g\big(f(f(v),5)\big) \\ y \rightarrow g(f(v),5)) \end{cases}$$

Exercice 23

1) Démarche syntaxique (ne peut être que syntaxique car il existe une infinité d’interprétations)
   a) Appliquer a chaque fbf de H les transformations mise sous forme prénexe et mise sous forme causale
   b) Ajouter a H la négation de la conclusion
   c) Essayer de montrer que l’ensemble obtenu est contradictoire . Pour cela, on utilise SF_r pour inférer la clause vide

	Clauses
$\forall$ x (P(x) $\Rightarrow$ Q(t(x)))	¬ P(x) $\vee$ Q(t(x))
$\forall$ x (Q(x) $\Rightarrow$ P(t(x))	¬ Q(x) $\vee$ P(t(x))
	¬ P(t(t(t(t(a)))))

Soit H’ ={ P(a), ¬ P(x) $\vee$ Q(t(x)), ¬ Q(y) $\vee$ P(t(y)), ¬ P (t(t(t(t(a)))))}



Remarque : Dans une démonstration une même hypothèse (clause) peut être utilisée plusieurs fois et quand chaque clause est utilisée plusieurs fois , elle peut être unifiée différemment à chaque fois.

2°) Il suffit de trouver un contre exemple qui soit modèle de H et non modèle de ∀ x P(x)

1

D = ℕ
P → Être pair
Q → Être impair
T → fonction successeur
A → 0

∀ x P(x)
∃ x ¬P(x) est vrai
X=1 et 1 n'est pas pair

Exercice 24

∀ x (P(x) ⇒ ∃ y(R(x) ∧ Q(x,y)))

∃ xP(x)

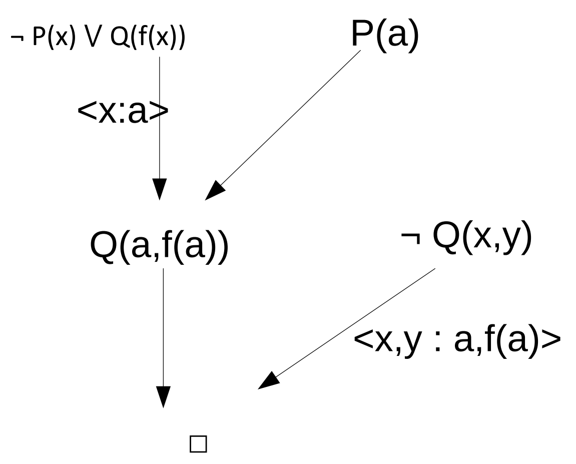
∃ x ∃ y Q(x,y)

Clauses

¬ P(x) V (R(x) ∧ Q(x,f(x)))

¬ P(x) V R(x) ∧ ¬ P(x) V Q(x,f(x)))
Soit :
¬ P(x) V R(x)
¬ P(x) V Q(x,f(x))
P(b)

¬ Q(x,y)



Remarque : Dans une démonstration toutes les hypothèses ne sont pas forcément utile.

Ex 25

H' = { ¬ P(x) V R(f(x)), ¬ R(y) V P(f(y)),P(a), ¬ P(f(f(a)))}.

3 substitution

x→ a

y → f(a)