1

Représentation des connaissances

HMIN231

Notes de cours

Auteur:
SAÏ Ismaël

Notes de cours prises dans le cadre de la première année de Master AIGLE.

Date - 20 juillet 2020

Table des matières

1 Introduction

Enseignant : Jean-François Baget.
Contact : Baget@lirmm.fr

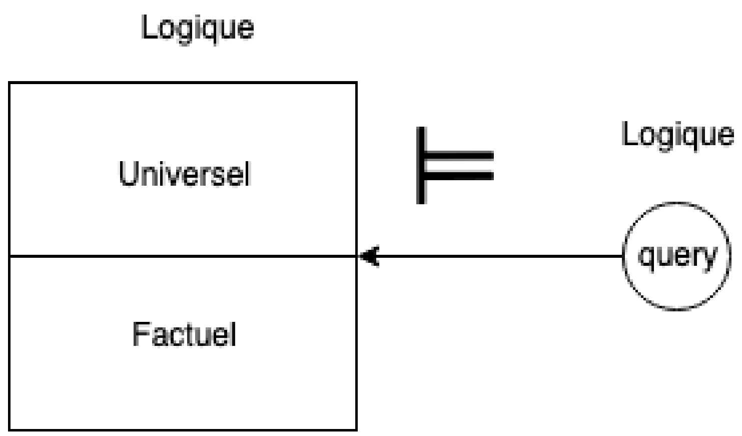

Fig. 1: Schéma Knowledge Base

Une base de connaissance KB (Knowledge base) est séparée en deux parties :

- **Factuelle**, qui sont les assertions ou relations sur des individus.
  Exemple : Socrate est un Homme.
- **Universelle (ou Ontologie)** , qui sont les assertions ou relations sur des ensemble d'individus.
  Exemple : Tout les Hommes sont mortel.

Remarque :
(*) Les composants factuels peuvent être stocké dans une base de donnée relationelle.
(*) Les composants universels font parti de la logique.

La logique des descriptions tend à vouloir répondre aux requêtes effectués sur une KB.
Exemple : Socrate est mortel ?

2 Partie I : Logiques standard

2.1 Syntaxe

Définition :

```
La syntaxe concerne les règles utilisées pour la construction de symboles et des mots d'un
    langage.
```

Pour ce cours, nous poserons **F** l'ensemble des formules ou assertions du langage.

2.2 Sémantique

Définition :

```
la sémantique d'une langue concerne sa signification, son interprétation. Une langue peut
    posséder plusieurs interprétations.
```

Pour ce cours, nous poserons **I** l'ensemble des interprétations.

Nous auront aussi besoin d'une fonction qui calcul si une formule f est vraie dans une interprétation i.

Exemple :

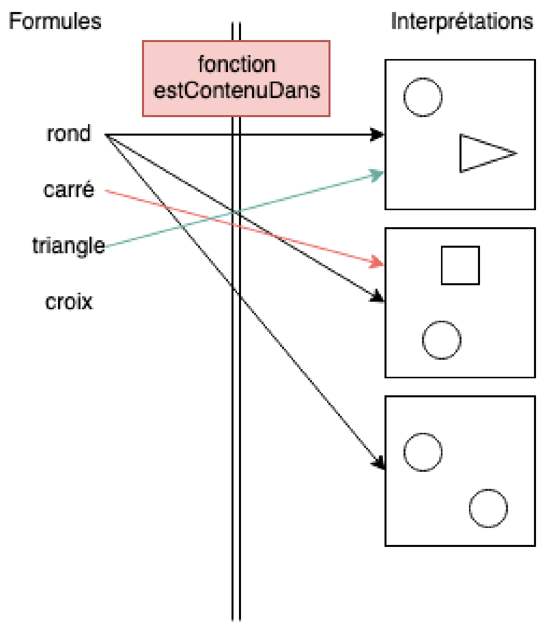

FIG. 2: Exemple de fonction

Définitions :

```
Si f est vraie dans i, alors i est un modèle de f. Sinon, c'est un contre-modèle.
```

```
une formule f est satisfiable s'il existe un modèle de f dans I. Sinon elle est
    insatisfiable.
```

```
une formule f est valide si tout interprétation de I est un modèle. S'il existe un contre-
    modèle, f est invalide.
```

```
une formule f est contingente si f est  la fois satisfiable et invalide, ie f possède  la
    fois un modèle et un contre modèle.
```

```
On dit que f' est conséquence sémantique de f si tout modèle de f est un modèle de f'.
```

Note : f' est conséquence sémantique de f se note aussi $f' \models f$. On dit que f a pour conséquence sémantique f'.
Exemple : Deux personnes entendent la phrase :
- Il pleut.

Les deux personnes s'accorderont à dire que puisqu'il pleut, le sol est mouillé. Même si l'un imagine une pluie torrentielle et l'autre une averse. Dans les deux interprétations possible, le fait qu'il pleut a pour conséquence sémantique que le sol est mouillé.

Propriété : Si $A \models B$ et $B \models C$, alors $A \models C$.

2.3 La logique des prédicats

2.3.1 Vocabulaire

Le vocabulaire de la logique des prédicats est constituée d'une liste d'atomes. Posons V la liste d'atomes de la logique des prédicats.

2.3.2 Syntaxe

Nous allons définir la syntaxe par induction structurelle.
Base : Si a est un atom, alors a est une formule.
Induction : Posons f une formule.
- Si f s'écrit sous la forme $(A \wedge B)$ alors f est une formule.
- Si f s'écrit sous la forme $(A \vee B)$ alors f est une formule.
- Si f s'écrit sous la forme $(\neg A)$ alors f est une formule.
- Si f s'écrit sous la forme $(A \Rightarrow B)$ alors f est une formule.

Remarque : Les parenthèses sont importantes surtout lors de l'utilisation d'un parser. En effet la formule $A \wedge B \wedge C$ ne sera pas valide pour le parser, il faut l'écrire sous la forum $(A \wedge (B \wedge C))$

2.3.3 Sémantique

ATTENTION : On interprète le VOCABULAIRE pas la FORMULE.

Posons la fonction i tel que :
$i : V \mapsto Bool$
Nous pouvons maintenant définir les propriétés suivantes :

- Si f est un atome, alors i est un modèle de f si et seulement si $f^i = True$
- Si $f = f1 \wedge f2$ alors i est un modèle de f si et seulement si $f1^i = True ET f2^i = True$
- De même pour les autres opérateurs logique.

Remarque : Les règle ci-dessus ne font que traduire les tables de vérités.

f1	f2	$f1 \wedge f2$
True	True	True
True	False	False
False	True	False
False	False	False

2.4 Logique des prédicats

2.4.1 Vocabulaire

Le vocabulaire de la logique des prédicats est composé de :
- nom de prédicats possédant une arité
- des constantes

ATTENTION : Un terme est soit une constante, soit une variable. Les variables ne faisant pas parti du vocabulaire.

2.4.2 Syntaxe

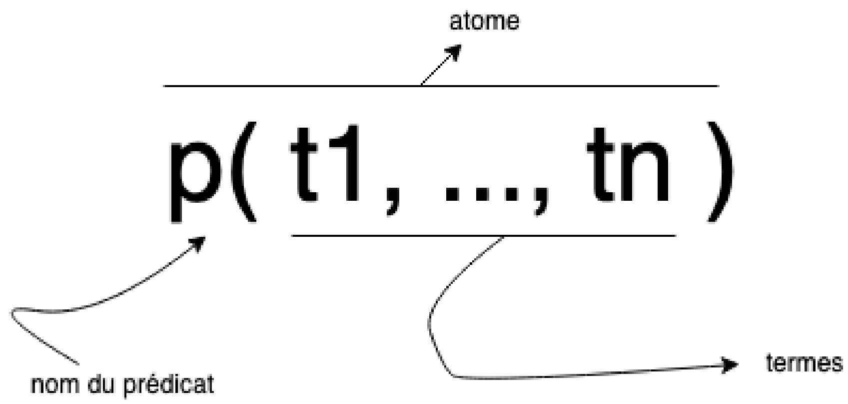

FIG. 3: Syntaxe de la logique des prédicats

Nous avons aussi les règles suivantes :

- $\exists x.F[X]$
- $\forall x.F[X]$

ATTENTION : Il faut aussi rajouter les règle nécessaire pour la gestion des variables libres et liés. N'étant pas l'objectif de ce cours, cette partie ne sera pas défini.

2.4.3 Sémantique

L'interprétation est égale à : $I = (\Delta, .^I)$ avec Δ le domaine d'interprétation (qui doit être différent de $\oslash$) et $.^I$ la fonction d'interprétation. **Remarque :** Le domaine d'interprétation est possiblement infini. Posons$C$l'ensemble de constantes.

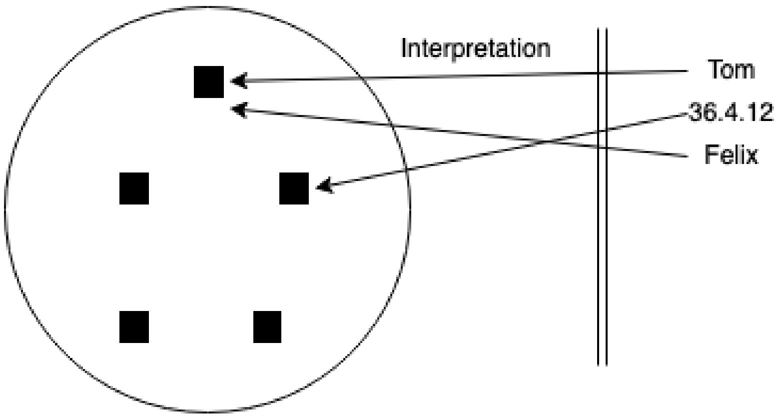

FIG. 4: Interprétation des constantes

ATTENTION : Dans certaines références, l'unique name assemption est en vigueur, c'est à dire que $\forall C, C', C \neq C' \Rightarrow C^I \neq C'^I$. En d'autre mot, deux constantes ne peuvent t'avoir la même interprétation.

Voyons maintenant l'intepétation :
Si $c \in C$, alors $C^I \in \Delta$

Interprétation du prédicat unaire Homme :

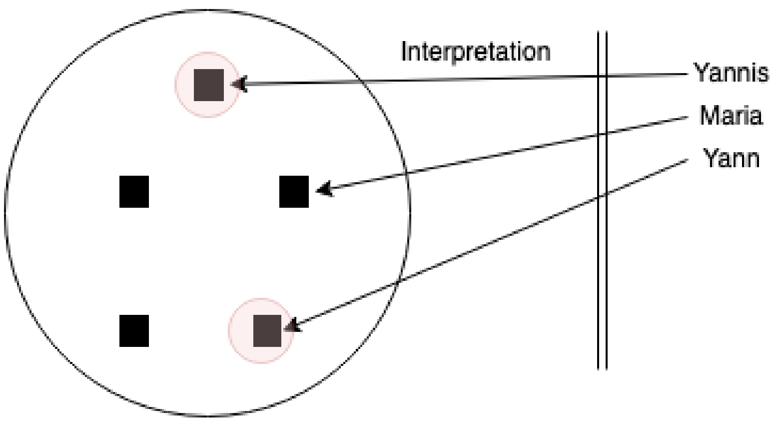

FIG. 5: En rouge, les constantes qui sont des hommes.

Posons 1 = Yannis et 2 = Maria.
Homme(1) renverra vrai.
Homme(2) renverra faux.
Pour généralisé, si $P \in P^{(1)}, alors P^I \subseteq \Delta$, en d'autres terme si P est un prédicat d'arité 1, alors son interprétation est incluse dans le Domaine d'interprétation.
Remarque : $P \in P^{(1)}, alors P^I \subseteq \Delta$ peut aussi s'écrire $P \in P^{(1)}, alors P^I \in P(\Delta)$ ou encore $P \in P^{(1)}, alors P^I \subseteq 2^{\Delta}$

Interprétation du prédicat binaire à.droite.de :
à.droite.de(1,2) renverra vrai. (cf. figure 5)
à.droite.de(2,1) renverra faux. (cf. figure 5)
Pour généralisé, si $P \in P^{(2)}, alors P^I \subseteq \Delta X \Delta$

Nous pouvons donc déduire que pour un prédicat d'arité k, $P \in P^{(k)}, alors P^I \subseteq \Delta^k$

3 La logique de description

La Logique de description, appellé aussi DL (description logic) est un fragment décidable de la FOL (first order logic, logique du premier ordre). Elle permet de représenter des ontologies.

Elle est constitués de deux choses, les concept (qui peuvent être comparé aux classes en Java, ou aux prédicats unaire en logique des prédicats) et les Rôles (qui eux peuvent t'être comparés aux prédicats binaire).

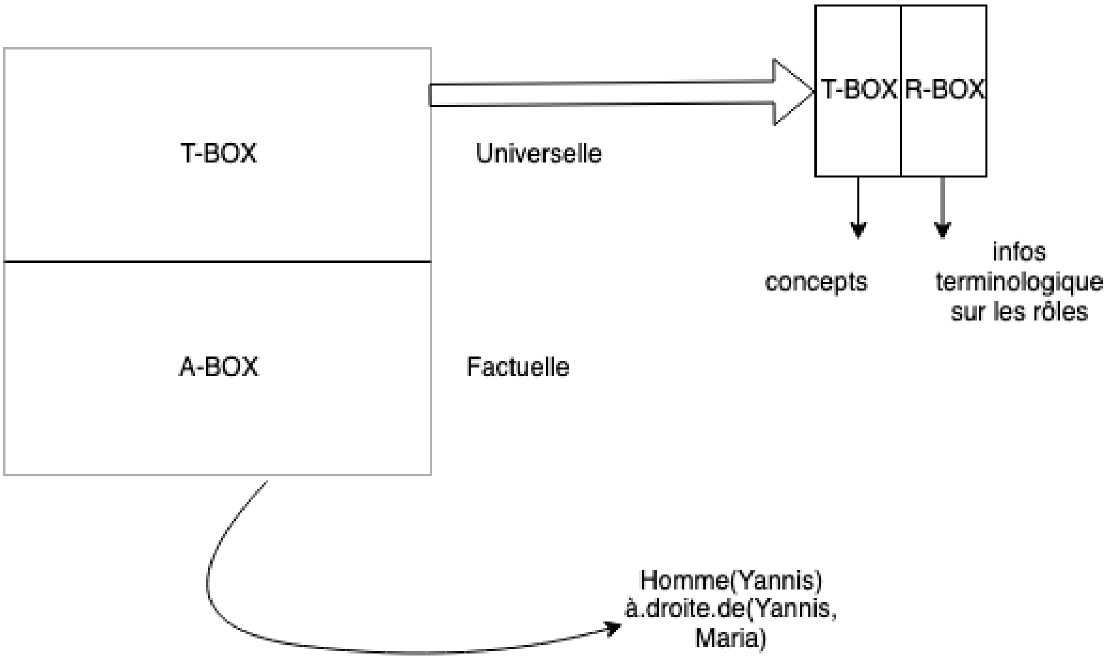

Fig. 6: Logique de description

Les concepts sont stockés dans la T-BOX de la KB. Les rôles dans la R-BOX.

ATTENTION : Posons C et D deux concepts. Alors on peut écrire $C \sqsubseteq D$
Posons R et S deux rôles. Alors on peut écrire $R \subseteq S$
$\subseteq$ est différent de $\sqsubseteq$, le premier concerne des ensembles tandis que le deuxième non.

3.1 Définition inductive des concepts et des rôles.

3.1.1 Vocabulaire

La logique de descritpion possède :
- des constantes
- des concepts (primitifs)
- des rôles (primitifs)

Base :
(*) Si C est un concept (primitif) alors C est un concept.
Induction :
(*) $\top$ (top) est le concept universel.
(*) $\bot$ (bottom) est le concept absurde.
(*) Si r est un rôle (primitif) alors r est un rôle.
(*) Supposons C et D des concepts et R et S des rôles, alors $C \sqcap D$ (se lit C et D) est un concept (construit)
Exemple: $Humain \sqcap Cheval$, ici on a défini les centaures.

(*) $C \sqcup D$ (se lit C ou D)
Exemple: $Homme \sqcup Femme$
(*) $\not C$ est un concept (construit)
(*)$\exists r.C$ est un concept (construit)
Exemple :
$\exists parent.Divinite \sqcap parent.Humain$

$\exists$ est un quantificateur existensieelle
q.exists restreint : $\exists r.T$ (*) $\leq$n R
(*) $\geq$n R

Exercices :

Les propositions suivantes sont elles vraies ?

$\forall r.C$

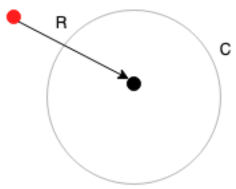

Fig. 7: Exercice 1

Réponse : Oui, car toutes les flèches de R vont dans C.

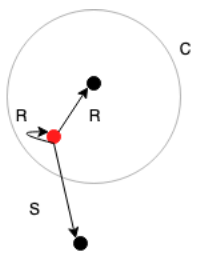

Fig. 8: Exercice 1

Réponse : Oui, car toutes les flèches de R vont dans C.

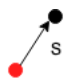

Fig. 9: Exercice 1

Réponse : Oui, par **vacuité**. S'il n'existe pas de flèches de R alors on peut en dire ce qu'on veut.
ATTENTION le piège de la vacuité tombe souvent en examen. La définition de la vacuité est la suivante : $\forall x \in \emptyset, P(x)$

3.1.2 Interprétation

L'interprétation du vocabulaire s'effectue comme en FOL (first order logic) Posons le vocabulaire suivant :
- **constantes :** Tom, Marie.
- **Concepts :** Homme, Femme.
- **Rôles :** marié.à

L'interprétation se note toujours $I(\Delta,.^I)$

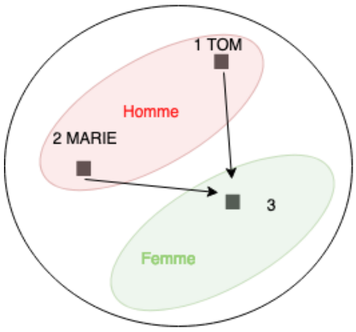

FIG. 10: Exemple Interprétation

$Tom^I = 1$
$Marie^I = 1{,}2$
$Femme^I = 3$ $marié.à^I = (1{,}3),(2{,}3)$

3.1.3 Comment interpréter un concept construit

(Attention la partie suivante tombe toujours à l'examen.)
Posons V un vocabulaire et $I = (\Delta,.^I)$ une interprétation de V. Soit C un concept construit sur V :
(*) Si C primitif, C' est déjà interprété dans I.
(*) Si $C = \top, C^I = \Delta$
(*) Si $C = \perp, C^I = \emptyset$
(*) Si $C = (C' \sqcap D), C^I = C'^I \cap D^I$
(*) Si $C = (C' \sqcup D), C^I = C'^I \cup D^I$
(*) Si $C = \neg C', C^I = \Delta$ $C^I = \overline{C^I}$
(*) Si $C = \exists R.C', C^I = \{x \in \Delta | \exists y(x,y) \in R^I \Rightarrow y \in C'^I\}$
(*) Si $C = (\geq nR)^I, C^I = \Big\{x \in D | card(\{(x,y) \in R^I\}) \geq n\Big\}$
De même pour $\leq$
Cela marche de la même façon pour les rôles, par exemple :
(*) $(R \sqcap S)^I = R^I \cap S^I$
(*) $(R^{-1})^I = \{(x,y) \in \Delta^2 | (y,x) \in R^I\}$

Exercices :

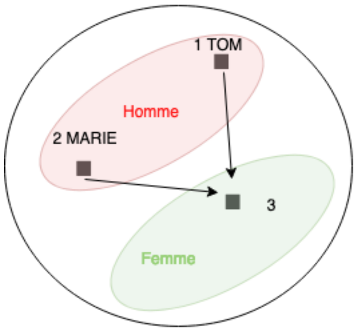

Fig. 11: Exercice 1

Donnez $(\forall R.(\neg(A \sqcap B)))^I$

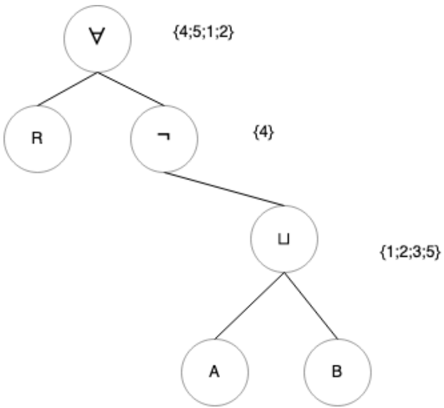

Fig. 12: Exercice 1 - arbre

Réponse : $\{4; 5; 1; 2\}$
ATTENTION À LA VACUITÉ AVEC LE $\forall$

Rappel important :

$R.A$ `sont les flèches qui vont vers A.` $R^{-1}.A$ `sont les flèches qui viennent de A.`

3.2 Résumé

La logique du premier ordre est monotone.
$C^F \models q$
Si on rajoute F' on a toujours q :
$C^F \wedge C^{F'} \models q$

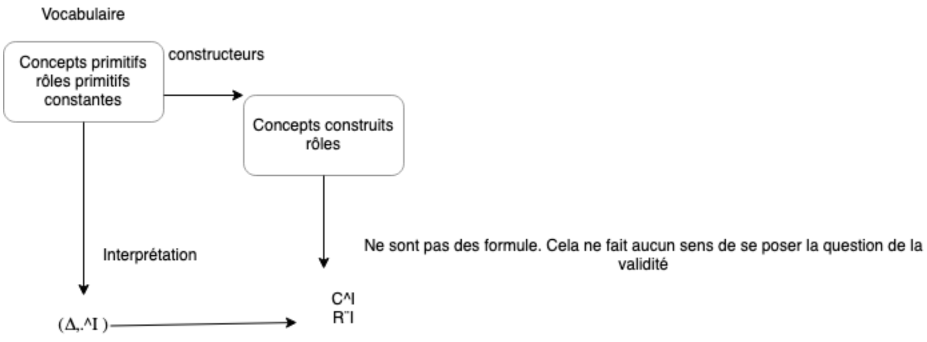

Fig. 13: Résumé

4 De la logique de description à la FOL

(ce chapitre est très important car il tombe toujours en CC/exam) Nous vous traduire la logique de description en FOL afin de pouvoir calculer la validité d'une formule.

4.1 Transformation

Base :
(*) Si C est un concept primitif alors $\Phi_{C)}(x) = C(x)$
(*) Si $C = \top\Phi_{C)}(\top) = \{TRUE, VRAIE, x = x, \square\}$
(*) Si $C = \bot\Phi_{C)}(\bot) = \{FALSE, FAUX, x, \blacksquare\}$
(*) $\Phi_{C\sqcap D)}(x) = \Phi_{(C)}(x) \wedge \Phi_{(D)}(x)$
(*) $\Phi_{C\sqcup D)}(x) = \Phi_{(C)}(x) \wedge \Phi_{(D)}(x)$
(*) $\Phi_{\neg(C)}(x) = \neg\Phi_{(C)}(x)$
(*) $\Phi_{(R.C)}(x) = y.\Phi_{(R)}(x,y) \wedge \Phi_{(C)}(y)$
(*) $\Phi_{\forall(R.C)}(x) = \forall y.(\Phi_{(R)}(x,y) \Rightarrow \Phi_{(C)}(y))$
(*) $\Phi_{(R\sqcup S)}(x,y) = \Phi R(x,y) \wedge \Phi S(x,y)$
(*) $\Phi_{(R\sqcup S)}(x,y) = \Phi R(x,y) \wedge \Phi S(x,y)$
(*) $\Phi_{(\geq nR)} = \exists y1...yn(\bigwedge(1 \leq i \geq n)\Phi_{(R)}(x,yi)) \bigwedge(1 \leq i \leq j \geq n)yi \neq yj$
Rappel :

DL	INT	FOL
$\sqcap$	$\cap$	$\wedge$
$\sqcup$	$\cup$	$\vee$
$\neg$	“	$\neg$
$\sqsubseteq$	$\subseteq$	$\Rightarrow$

Exemple :

Traduire le concept $(C \sqcap \neg(D \sqcup E))$.
$\Phi_{(C\sqcap\neg(D\sqcup E))}(x)$
$= \Phi_{(C)}(x) \wedge \Phi_{(\neg(D\sqcup E))}(x)$
$= C(x) \wedge (\neg\Phi_{(D\sqcup E)}(x))$
$= C(x) \wedge (\neg\Phi_{(D)}(x) \vee \Phi_{(E)}(x))$
$= C(x) \wedge (\neg D(x) \vee E(x))$

deux propriétés importantes
(+) Il n'y a plus de Φ dans la formule à la fin.
(+) x est une variable libre.

Théorème
I est un modèle de $C \sqsubseteq D$ si et seulement si I est un modèle de $\forall x.(\Phi_{(C)}(x) \Rightarrow \Phi_{(D)}(x))$. Toutes instances de C est une instance de D.

Notation : $C \equiv D$ valide si et seulement si
$C \subseteq D$ valide
$D \sqsubseteq C$ valide

(1) Si C concept alors $\neg\neg C \equiv C$ valide .
$(\neg\neg C)^I = (\overline{(\neg C)^I}) = \overline{\overline{C^I}} = C^I$
$\neg\neg C = C$
$\neg\neg C \subseteq C.\forall I.(\neg\neg C)^I \subseteq C^I$
$\neg\neg C \sqsupseteq C.\forall I.(\neg\neg C)^I \supseteq C^I$
$\forall I.(\neg\neg C)^I = C^I$

(2) $\neg(C \sqcap D) \equiv (\neg C) \sqcap (\neg D)$
$\neg(C \sqcap D)^I \equiv (\overline{(C \sqcap D)^I} \equiv (\neg C)^I \sqcap (\neg D)^I$
$\neg(C \sqcap D)^I \equiv \overline{(C \sqcap D)^I} \equiv \overline{C^I} \sqcup \overline{D^I} \equiv \neg C^I \sqcup \neg D^I$

Exemple de démonstration
$\neg(C \sqcup D) \equiv (\neg C) \sqcup (\neg D)$
Je vais prouver que $\forall I, (\neg(C \sqcup D))^I = ((\neg C) \sqcup (\neg D))^I$

Soit I une interprétation quelconque.
$(\neg(C \sqcup D))^I = \overline{(C \sqcup D)^I} = \overline{(C^I) \cap \overline{(D^I)}}$
Par définition du not et définition du $\sqcup$

Prouver que $A \sqsubseteq B$ valide si et seulement si $\neg A \sqcup B \equiv \top$ valide.

Je vais prouver que $\forall I, (A \sqsubseteq B)^I = (\neg A \sqcup B \equiv \top)^I$
Soit I une interprétation quelconque :
$(A \sqsubseteq B)^I = A^I \subseteq B^I$
$(\neg A \sqsubseteq B)^I = (\neg A)^I \cup (B)^I$

Un autre exemple
Montrons que $A \sqsubseteq B$ valide $\Leftrightarrow \neg A \sqcup B \equiv \top$ valide.
Prouvons successivement les deux sens de l'équivalence.

(sens 1, $\Rightarrow$) : $A \sqsubseteq B$ valide $\Rightarrow \neg A \sqcup B \equiv \top$
Posons A et B deux concepts
$\forall I, A^I \subseteq B^I$
$(\neg A \sqcup B)^I = (\overline{A^I} \cup \overline{B^I}) = \Delta$
Méthode logique :
$\neg(\forall R.C) \equiv \exists R.\neg C$ valide
$\Phi_{\forall R.C} = \exists y(R(x,y) \wedge \neg C(y))$

4.2 Règles sur la DL

(*) $\neg(A \sqcup B) \equiv \not A \sqcap \not B$
Exemple de démonstration avec la méthode FOL
(sens $\Rightarrow$)
On veut démontrer que :
$\forall x.\Phi_{(A \sqcup B)}(x) \Rightarrow \Phi_{(\neg A \sqcap B)}(x)$
$\neg(A(x) \vee B(x)) = \neg A(x) \wedge \neg B(x) = \Phi_{(\neg A \sqcap \neg B)}(x)$

(*) $\neg\neg C = C$
(*) $\neg(C \sqcup D) = \neg C \sqcap \neg D$
(*) $\neg(C \sqcap D) = \neg C \sqcup \neg D$
(*) $R.C = \forall R.\neg C$
(*) $\forall R.C = \exists R.\neg C$
(*) $\neg(\geq n.R) = \leq (n-1).R$
(*) $\neg(\leq n.R) = \geq (n+1).R$
(*) $A \sqcap (B \sqcup C) = (A \sqcup B) \sqcap (A \sqcup C)$
(*) $A \sqcup (B \sqcap C) = (A \sqcap B) \sqcup (A \sqcap C)$

$(A \sqcup B) \sqcap \neg C \equiv C \sqcap D$ marche toujours si nous remplaçons C par un concept quelconque (règle de substitution)

4.2.1 Algorithmes

5 Familles de langages

Une famille de langages en DL est determinée par le choix d'un noyau initial de constructeurs.

La famille $\mathcal{AL}$ est constitué des constructeurs $\top$ $\bot$ $\sqcap$ $\not{a}$tomique $\exists R.\top$ et $\forall$

Attention not ne porte que sur concept primitif, on appelle ça la négation restreinte. La quantification est existensielle restreinte. $\sqcap$ ne s'applique pas sur les rôles, seulement sur les concepts.
Nous pouvons construire des langages au-dessus de $\mathcal{AL}$ en ajoutant les constructeurs suivants :
$\sqcup$ Union $\mathcal{U}$
$\neg$ (negation $\mathcal{AL}$) $\mathcal{C}$
$\exists R.C$ (quantificateur existentielle) $\mathcal{E}$
$\leq \geq$ (restriction numérique) $\mathcal{N}$
$\sqcap$ (conjonction de rôle) $\mathcal{R}$

Exemple la grammaire $\mathcal{ALEC}$

5.1 égalités entre langages

Montrons que $\mathcal{ALUE} = \mathcal{ALC}$
Si $\mathcal{C}$ est un concept de $\mathcal{ALUE}$ alors $\exists C'$ dans $\mathcal{ALC}$ tel que $C \equiv C'$ Prenons une formule quelconque de $\mathcal{ALUE}$

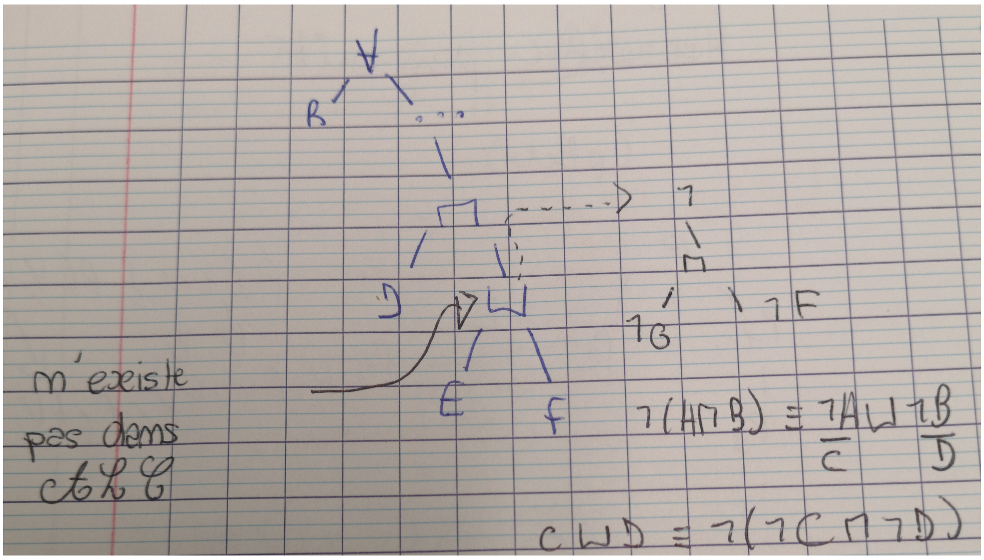

FIG. 14: Exercice 1 - arbre

On peut transformer $\neg(A \sqcap B)$ car tout les constructeurs ne sont pas présent dans le langage $\mathcal{ALC}$ (sens $\Rightarrow$)
$\neg(A \sqcap B) = \neg A \sqcup \neg B$
$C \sqcup D = \neg(\neg C \sqcap \neg D$

de même pour $\exists R.C$
$\neg(\forall R. \not{C})$. Et de même pour les autres occurences.
On aura $F \in \mathcal{ALUE} \Rightarrow F \in \mathcal{ALC}$
Donc $\mathcal{ALUE} \subseteq \mathcal{ALC}$
ATTENTION tout les langages ne sont pas égaux, il faut consulter le treilli de langages sur moodle. Un exercice qui tombe souvent à l'examen est de montrer que deux langages sont égaux/ne sont pas égaux.

(sens $\Leftarrow$)

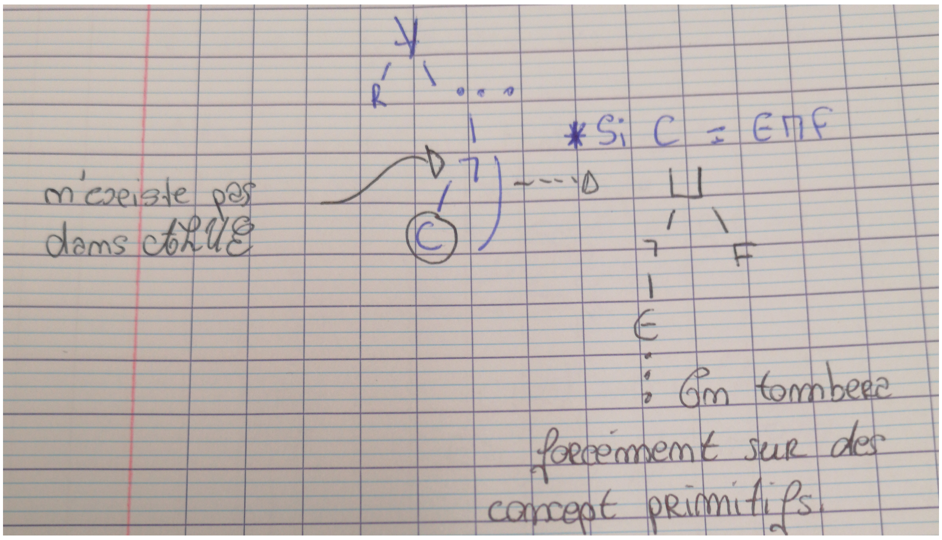

Fig. 15: Exercice 1 - arbre 2

Donc $\mathcal{ALC} \subseteq \mathcal{ALUE}$
Donc $\mathcal{ALUE} \equiv \mathcal{ALC}$

5.1.1 Algorithmes

SAT(«$\mathcal{F}$», x)
T-BOX vrai $\models$ requête $C \sqsubseteq D$
Pour pouvoir traiter ce genre de requêtes il faut que :
(*)La T-BOX possède des inclusions de type $CP \in CC$ avec CP un concept primitif et CC un concept construit.
(*) pas de ciruit dans ces définitions.
par exemple $X \sqsubseteq B \sqcap CB \sqsubseteq A \sqcap D$ est interdit. (*) $c1 \sqsubseteq d1 \sqcap c2 \sqsubseteq d2 \sqcap ... \sqcap ck \sqsubseteq dk \models C \sqsubseteq D$

Nous pouvons effectuer une opération de unraveling.
$A \sqsubseteq B \sqcup \neg C$
$D \sqsubseteq \exists R.E$
$A \sqcap D \sqsubseteq C$
$(B \sqcup \neg C) \sqcap D \sqsubseteq C$
$(B \sqcup \neg C) \sqcap (\exists R.C) \sqsubseteq C$

$\oslash \models C \sqsubseteq D$ `si et seulement si unraveling` $(C \in D) valide$

Exemple :

$A \sqsubseteq C \sqcup D$
$B \sqsubseteq A \sqcap \not C$
$F \sqsubseteq A \sqcup B$
$\models A \sqcap \neg B \sqsubseteq F$

(1)
(2) Unraveling :

$(C \sqcup D) \sqcap \neg B \sqsubseteq F$
$(C \sqcup D) \sqcap \neg(A \sqcap \neg C) \sqsubseteq F$
$(C \sqcup D) \sqcap \neg(A \sqcap \neg C) \sqsubseteq (A \sqcap B)$
$(C \sqcup D) \sqcup \neg((C \sqcup D) \sqcap \neg C) \sqsubseteq ((C \sqcup D) \sqcap B))$
$(C \sqcup D) \sqcup \neg((C \sqcup D) \sqcap \neg C) \sqsubseteq ((C \sqcup D) \sqcap \neg C))$

(3) $C \sqsubseteq D$ valide ? $\equiv C \sqcup \neg D \sqsubseteq \perp$ valide.
Ce qui veut dire $x \in (C \sqcap \neg D)^I$ faux.
$\forall I \Rightarrow C \sqcap \neg D : x$ Instatisfiable.
On a un algorithme pour cela.
(def sans circuit)$\oslash \models C \sqsubseteq D \frown$(unraveling) $C' \sqsubseteq D'$ valide. $\frown C' \sqcap D' : x$ insatisfiable (sous forme normal négative (c'est à dire que toute neg porte sur concept primitif $\frown C'' : X$ instatisfiable.

5.1.2 Les trois types de clashs.

Règle 1 :

$C : x$ $\neg C : x$

Règle 2 :

$x = y$ $x \neq y$

Règle 3 :

$\perp : x$

(*) Parcours en largeur des sucesseurs. D'abord le $C(x)$ puis $C(y)$, puis $C(z)$, etc ...
$\sqcup \rightarrow \sqcap \rightarrow \exists \rightarrow \forall \rightarrow \geq \rightarrow \leq$

Exemple :

$Woman \sqsubseteq Female \sqcap Person$ $Parent \sqsubseteq Person \sqcap \exists\ has.child.Person \sqcup \forall has.child.person$ $Mother \sqsubseteq Female \sqcup Parent$ $Mother \sqsubseteq Woman?$

(1) $F \sqcup Pa \sqsubseteq F \sqcup Pe$(unraveling)
$F \sqcup Pe \sqcup \exists hc.Pe \sqsubseteq F \sqcup Pe$
(2) valide ?
$F \sqcup Pe \sqcup .Pe \sqcup \forall hc.Pe \sqcap \neg(F \sqcap Pe) : x$ insatisifiable ?
$F \sqcap Pe \sqcap \exists hc.Pe \sqcap \neg(F \sqcup Pe) : x$ instatisfiable ?

6 $FOL(,\exists)$

6.1 Vocabulaire :

– constantes $\mathcal{C}$

– prédicat, arité $\mathcal{P}$

x: variables. **Attetnion** ne sont pas dans le vocabulaire.
les constantes commence par minuscule les variables par majuscule.
Induction :
γ (termes) $= \mathcal{C} \cup \mathcal{X}$
Atomes $= p(t1,...,tr) p \in \mathcal{P}$ d'arité k et $t1,...,tk$ termes.
Conjononctions : $a1...ap$ où les ai sont des atomes.
fait : fermeture existensielle d'une conjonction.

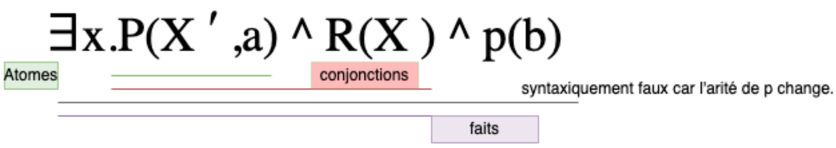

FIG. 16: Exercice 1

Notation :
drawio ici
Nous sommes pas loin d'avoir un ensemble, on va donc dire qu'un fait est un ensemble d'atomes : $\{p(X,Y),p(X,b)\langle\}$

6.2 Sémantique

Qu'est-ce qu'une interprétation ?
RAPPEL : On interprete TOUJOURS un vocabulaire et non pas les variables.

C'est comme en FOL :
$I = (\Delta,.^I)$ avec Δ le domaine d'interprétation et $.^I$ la fonction d'interprétation.
(*) $\Delta \neq \oslash$
(*) si $c \in \mathcal{C}$ alors $c^I \in \Delta$
(*) si $p \in \mathcal{P}$ (d'arité k) alors $P^I \in \delta^k$

$C^I = 2$
$D^I = 5$
$P^I = \{(1,2),(2,4)\}$
$Q = \{(3,4,5),(3,1,1)\}$
$R = \{2,4\}$

I est un modèle d'un fait ?
F = q(X,Y,Z), p(Z,C)
Une valutaion de F dans I c'est une application des terms(F) dans Δ tel que si $C \in \mathcal{C}, val(C) = C^I$

Définition :

`I est un modèle de F si et seulement si il existe une valutaion val de terms(F) dans` Δ `tel que` $\forall p(t1,...,Tk) \in I, val(t1),...,val(tk) p^I$