Chapitre 6 : LES LOGIQUES DE DESCRIPTION

I- Introduction :

- Les **Logiques de Description (LD)** sont des langages de représentation des connaissances mettant l'accent sur le raisonnement
- **L'objectif majeur** : raisonner efficacement (temps de réponse minimal) pour la prise de décision

=> Importance du rapport expressivité/performance des différentes LD

- **Une approche ontologique** : pour représenter la connaissance d'un domaine les LD demande la définition :
  - de **catégories générales d'individus**
  - de **relations logiques** que les **individus** ou **catégories** peuvent entretenir **entre eux**.

- **Les LD s'appuient sur** : la Logique des Prédicats, les Schémas (Frames) [Minsky, 1981], les Réseaux Sémantiques [Quillian, 66], …
- **Nombreuses correspondances** : catégories générales d'objets et de relations fait partie de l'héritage dans les schémas et réseaux sémantiques.

1° génération de LD (1980 - 1990) : langages de représentation des connaissances mettant l'accent sur le raisonnement :

- **liées aux travaux sur les systèmes à base de connaissances** : KL-ONE [Brachman & Schmolze 85], LOOM [MacGregor & al. 86], …
- **raisonnements et inférences en temps polynomial :**
  - avec des **algorithmes** de vérification de subsomption de type **normalisation/comparaison** (structural subsumption algorithms).
  - réservés à des **LD peu expressives**, sinon **incomplets**, incapables de prouver certaines formules vraies.

2° génération de LD (1990 à aujourd'hui) : Logiques de Description Expressives (LDE) :

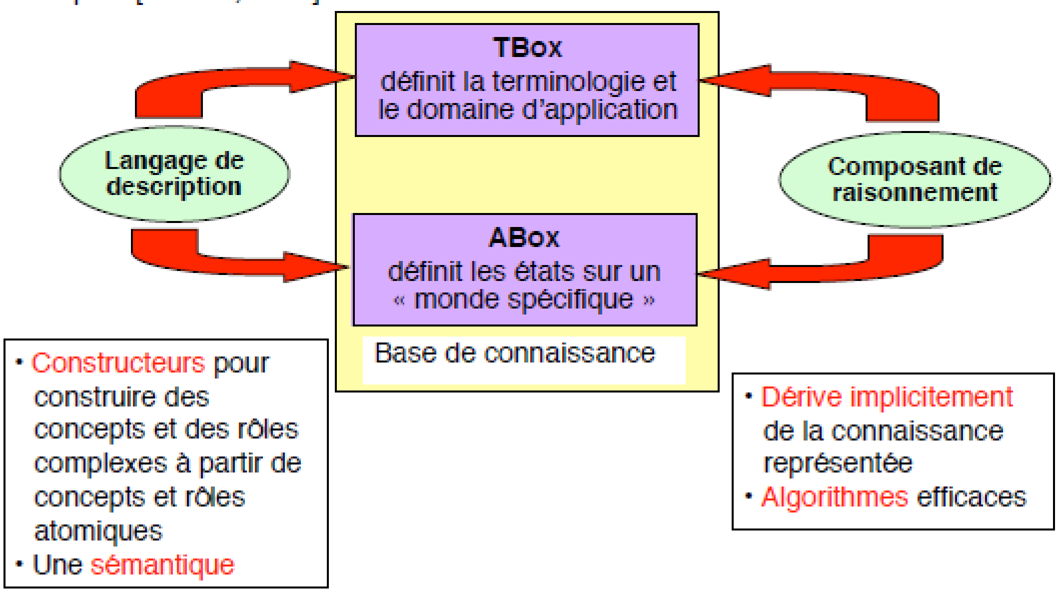
- **années 1990** : **nouveaux algorithmes** de vérification de satisfiabilité à base de tableaux (tableau-based algorithms) :
  - raisonnant sur des **LD dites expressives** ou **très expressives**
  - en **temps exponentiel**
  - mais en pratique, au **comportement acceptable** [Baader & al., 2003]

La grande expressivité des LD traitées par ces algorithmes a ouvert la porte à de **nouvelles applications** comme le **Web Sémantique** [Baader, Zou, Horrocks & al., 2003]

- **Les LD permettent de représenter la connaissance d'un domaine au travers :**
  - de **concepts** (ou classes) du domaine et
  - de **relations** (ou rôles) pouvant être établies **entre** ces **classes** ou entre les **instances** de ces classes appelées **individus**
- **Modélisation des connaissances avec les LD à 2 niveaux :**
  - **le niveau terminologique ou TBox :**
    - décrit les **connaissances générales d'un domaine**
    - définit les **concepts** (classes) et les **rôles** (relations)
  - **le niveau factuel ou ABox :**
    - décrit les **individus** en les **nommant** et en spécifiant leur classes et attributs (en termes de concepts et de rôles)
    - spécifie des **assertions** portant sur ces individus nommés.
- **Remarque : plusieurs ABox** peuvent être **associées à une même TBox** :
    - chacune représente une **configuration** constituée d'individus,
    - utilise les **concepts** et **rôles** de la **TBox** pour l'exprimer.

II-Architecture des systèmes à base des LD :

D'après [Baader, 2000]:



1- Le niveau Terminologique (TBOX) :

1-1 Définition :

La TBox contient :

- **Les Entités Atomiques : concepts atomiques** et **rôles atomiques constituant** les entités élémentaires de la LD
- **4 Concepts et Rôles atomiques prédéfinis minimaux :**
  - le **concept** $\top$ **et le rôle** $\top_R$, les plus généraux de leur catégorie
  - le **concept** $\perp$ **et le rôle** $\perp_R$, les plus spécifiques (ensemble vide)
- **Les Entités Composées :**
  - **concepts** et **rôles atomiques** peuvent être **combinés** au moyen de **constructeurs** pour former des entités **composées**

Conventions :

- ***A** et **B** dénotent des **concepts atomiques***
- ***C** et **D** dénotent des **concepts composés***
- ***R** dénote un rôle*
- *les nom de **concepts** commencent par une **Majuscule** : Ex : Homme, Femme, …*
- *les noms de **rôles** par une **minuscule** : Ex : relationParentEnfant, …*

- **Les constructeurs** : permettent la **combinaison** de **concepts** et **rôles atomiques** pour former des **entités composées** :

  Ex : le concept composé Mâle ⊓ Femelle : résulte de l'application du constructeur ⊓ aux concepts atomiques Mâle et Femelle, et **s'interprète** comme l'ensemble des individus appartenant aux concepts Mâle et Femelle.

  Les LD se **distinguent** par les **constructeurs** qu'elles proposent :

  - plus elles ont de constructeurs, plus elles sont **expressives**, et ont des chances d'être **non décidables** ou de **complexité très élevée**
  - les LD **trop peu expressives** ne permettent pas de représenter des domaines complexes.

- **Les axiomes terminologiques d'une des 2 formes** :
  - **C ≜ D** (ou C ≡ D) : énonçant des **relations d'équivalence** (de définition) entre concepts : *C équivalent par définition à D*
  - **C ⊑ D** : énonçant des **relations d'inclusion : *C est inclus dans D***

1-2 Consistance et Subsomption:

- **Consistance de concepts** :
  - un **concept** (une classe) est **consistant**, s'il existe **au moins un individu membre de cette classe :**

    Remarque : si on définit une classe (concept) comme étant à la fois une sous-classe des classes Homme et Femme, et que la TBox spécifie aussi que ces 2 classes sont disjointes (aucun individu ne peut à la fois être un Homme et une Femme) : ce nouveau concept est alors inconsistant.

- **Subsomption de concepts** :
  - la **subsomption** consiste à **déduire qu'un concept, cad une classe, est une sous-classe d'une autre classe**, même si ce n'est pas déclaré explicitement dans la base de connaissances :

    Ex : si on spécifie que Humain est une sous-classe de Animal, que Mere est une sous classe de Humain, on peut déduire qu'une Mère est une sous-classe de Animal : Mere $\sqsubseteq$ Animal

1-3 Interprétation d'une TBOX :

- **tout concept** est associé à **un ensemble d'individus** dénotés par ce concept
- **Une interprétation $\mathcal{I}$ suppose l'existence :**
  - d'un **domaine d'interprétation** Δ ou $\Delta^{\mathcal{I}}$, ensemble non vide représentant les entités du monde décrit et composé d'individus
  - d'une **fonction d'interprétation** $\mathcal{I}$ ou $\cdot^{\mathcal{I}}$, assignant :
    - à chaque **concept atomique A**, un ensemble $\mathbf{A}^{\mathcal{I}}$, tel que $\mathbf{A}^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
    - à chaque **rôle atomique R**, une relation binaire $\mathbf{R}^{\mathcal{I}}$ telle que $\mathbf{R}^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
- l'interprétation $\mathcal{I}$ **satisfait un axiome d'équivalence C $\equiv$ D ssi $\mathbf{C}^{\mathcal{I}} = \mathbf{D}^{\mathcal{I}}$** *(égalité des ensembles d'individus)*
- l'interprétation $\mathcal{I}$ **satisfait un axiome d'inclusion C $\sqsubseteq$ D ssi $\mathbf{C}^{\mathcal{I}} \subseteq \mathbf{D}^{\mathcal{I}}$** *(inclusion des ensembles d'individus).*
- l'interprétation $\mathcal{I}$ **satisfait une TBox $\mathcal{T}$ ssi $\mathcal{I}$ satisfait tous les axiomes de la TBox $\mathcal{T}$** *(on dit que $\mathcal{I}$ est un modèle de la TBox $\mathcal{T}$).*

2- Le niveau Assertionnel : ABOX :
2-1 Définition

- **Une ABox contient 2 types d'assertions sur des individus :**
  - des **assertions d'appartenance : spécifiant** leur **classe** et leurs **attributs** :

    Ex : Marie est une femme et qu'elle a 2 enfants ; Marie est une Mère (individu instance de la classe mère)

  - des **assertions de rôle** : **spécifiant** les **relations existantes entre individus** :

    Ex : une mère doit avoir au moins un enfant : la ABox devra contenir au moins un autre individu, et une relation entre celui-ci et Marie indiquant qu'il est un de ses enfants.

Convention :

*les **individus nommés** sont représentés par des lettres **a, b***

2-2 Interprétation :

- Une **fonction d'interprétation** $\mathcal{I}$ ou $\cdot^{\mathcal{I}}$, **associe à chaque nom d'individu nommé a**, un **individu** $a^{\mathcal{I}}$ tel que $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ *(Les moteurs d'inférence pour LD font souvent l'hypothèse de noms uniques : pour tout individu nommé a et b, $a^{\mathcal{I}} \neq b^{\mathcal{I}}$)*
- **Interprétation d'une Assertion d'appartenance d'une ABox :**
  - étant donnée une **assertion d'appartenance notée C(a)** déclarant que pour cette ABox, il existe un individu nommé a, membre du concept C de la TBox associée : **une interprétation $\mathcal{I}$ satisfait C(a) ssi $a^{\mathcal{I}} \in C^{\mathcal{I}}$**
- **Interprétation d'une Assertion de rôle d'une ABox :**
  - étant donné une **assertion de rôle R(a, b)** déclarant que pour cette ABox, il existe un individu nommé a, en relation avec un individu nommé b par le rôle R (défini dans la TBox associée), tel que a fait partie du domaine de R et b fait partie de l'image de R : **une interprétation $\mathcal{I}$ satisfait R(a,b) ssi $(a^{\mathcal{I}},b^{\mathcal{I}}) \in R^{\mathcal{I}}$**
- Une **interprétation $\mathcal{I}$ satisfait une ABox $\mathcal{A}$** ($\mathcal{I}$ est un **modèle de la ABox $\mathcal{A}$**) **ssi $\mathcal{I}$ satisfait toutes les assertions de $\mathcal{A}$.**

Exemple :
Considérons la base de connaissances suivante composée d'une TBOX et d'une ABOX :

TBox :

Femme ≡ Personne ⊓ Femelle
Homme ≡ Personne ⊓ ¬ Femelle
Mere ≡ Femme ⊓ ∃aEnfant.Personne
Pere ≡ Homme ⊓ ∃aEnfant.Personne
Parent ≡ Pere ⊔ Mere
GrandMere ≡ Mere ⊓ ∃aEnfant.Parent
MereAvecPlusieursEnfants ≡ Mere ⊓ ≥3 aEnfant
MereSansFille ≡ Mere ⊓ ∀aEnfant.¬Femme
Epouse ≡ Femme ⊓ ∃aCommeMari.Homme

ABox :

MereSansFille(Marie) Femme(Alice)
Pere(Pierre)
aEnfant(Marie,Pierre)
aEnfant(Marie,Paul)
aEnfant(Pierre,Alice)

on peut en déduire GrandMere(Marie)

III- LA LOGIQUE DE DESCRIPTION MINIMALE ALC :

La logique de description ALC (Attributive Langage with Complement) est une logique de description minimale définie à partir de la logique de base AL.

1- Syntaxe du langage AL :

Soit :

- *A un concept **atomique**, C et D des concepts **atomiques ou complexes**, et R une relation (rôle)*
- ⊤ : le concept universel
- ⊥ : le concept impossible (ou plus spécifique)

Constructeurs d'$\mathcal{AL}$:

¬ A	la négation atomique
C ⊓ D	l'intersection de concepts
∀R.C	la restriction de valeur (quantification universelle complète)
∃R.⊤	la quantification existentielle limitée *

(*) : Ex : Personne ⊓ ∃aEnfant.⊤ : personne ayant au moins un enfant

2- Syntaxe du langage ALC :

- $\mathcal{ALC}$ **(Attributive Langage with Complement)** [Schmidt-Schaub & Smolka 88], est **minimale**, dans le sens où une logique moins expressive représente peu d'intérêt
- $\mathcal{ALC}$ est l'extension de la LD de base $\mathcal{AL}$ à la négation de concept composé ($\mathcal{C}$ - complément), et à la quantification existentielle complète
- $\mathcal{ALC}$ est la logique de description la plus importante, car elle constitue la base de toutes les LD pratiques
- **Signature de la LD $\mathcal{ALC}$:**
  - **La LD $\mathcal{ALC}$ est défini par un tuple ordonné Σ = (C, R, O) de 3 alphabets disjoints :**
    - l'ensemble **C de noms de concepts**
    - l'ensemble **R de noms de rôles**
    - l'ensemble **O de noms d'objets** (ou **noms d'individus**)
  - les noms de concepts et de rôles sont aussi appelés **concepts atomiques et rôles atomiques**

Soit :

- *C et D des concepts* ***atomiques ou complexes****, et R une relation (rôle)*
- $\top$: le concept universel
- $\bot$: le concept impossible (ou plus spécifique)

Constructeurs d'$\mathcal{ALC}$:

$\neg$ C	non C ou Complément de C
C $\sqcup$ D	l'union de concepts (C **ou** D) - ($\mathcal{ALC}$-$\mathcal{U}$)
C $\sqcap$ D	l'intersection de concepts (C **et** D)
$\exists$R.C	la quantification existentielle (existential restriction)*
$\forall$R.C	la quantification universelle (universal restriction)

(*) : Ex :

- $\exists$aEnfant.Personne: personne ayant au moins un enfant
- $\exists$aEnfant.Femme: personne ayant au moins une fille

Signification intuitive des symboles :

Symbole/expression	Signification
concept	ensemble
rôle	relation binaire
¬	ensemble complémentaire
⊔	ensemble union ($\mathcal{ALCU}$)
⊓	ensemble intersection

- **constructeur ¬C** : **négation (complément) d'un concept** désignant (pour une interprétation), l'ensemble des individus n'appartenant pas au concept atomique C :

  Ex : soit le concept Humain représentant l'ensemble des humains, ¬Humain représente l'ensemble des individus qui ne sont pas des humains

- **constructeur C ⊔ D** : disjonction (union) de 2 concepts composés désignant (pour une interprétation), l'ensemble des individus membres soit du concept C ou soit du concept D

  Ex : Etudiants ⊔ Enseignant représentant l'ensemble des individus qui sont étudiants OU enseignants

- **constructeur C ⊓ D** : conjonction (intersection) de 2 concepts composés désignant (pour une interprétation) l'ensemble des individus membres à la fois du concept C et du concept D

  Ex : Etudiants ⊓ Male représentant l'ensemble des individus qui sont étudiants ET males

- **quantificateur existentiel ∃R.C** : désigne (pour une interprétation), l'ensemble des individus, membres du domaine d'un rôle R

  Ex : dans l'interprétation $\mathcal{I}$, modèle de l'ABox et la TBox précédentes (page 20), ∃aEnfant est l'ensemble des individus {Pierre$^{\mathcal{I}}$, Paul$^{\mathcal{I}}$, Alice$^{\mathcal{I}}$}.

- **quantificateur universel ∀R.C :** désigne (pour une interprétation), l'ensemble des individus du domaine d'un rôle R qui sont en relation par R avec un individu du concept C, pour une interprétation donnée.

  Ex : pour la même interprétation $\mathcal{I}$, ∀aEnfant.Femme est l'ensemble des individus {Alice$^{\mathcal{I}}$}.

- **ensemble** ⊤ : désigne (pour une interprétation) l'ensemble de tous les objets/individus

- **ensemble** ⊥ : désigne (pour une interprétation) vide

Remarque : $\mathcal{AL}$ **ne permet pas** la **spécification** de **rôles** à l'aide de **constructeurs** (pas de rôles composés).

Exemple :

- Pour déclarer « **un humain est un animal** », on peut utiliser les concepts atomiques Humain et Animal et déclarer l'axiome :

  $\text{Humain} \sqsubseteq \text{Animal}$ *(Humain est inclus dans Animal)*

- Pour déclarer « **un humain est un animal qui raisonne** », on peut définir le concept Raisonnable et déclarer l'axiome :

  $\text{Humain} \equiv \text{Animal} \sqcap \text{Raisonnable}$

  Il y a l'équivalence entre :

  - le concept Humain représentant l'ensemble des humains,
  - le concept $\text{Animal} \sqcap \text{Raisonnable}$ représentant l'ensemble des individus appartenant à la fois à la classe Animal et à la classe Raisonnable.

- $\mathcal{AL}$ possède la **négation**, seulement appliquée qu'à un **concept atomique** : la classe des non humains est : $\neg\text{Humain}$
- $\mathcal{ALC}$ possède la **négation** appliquée à un **concept atomique** ou **composé** : la classe des individus qui ne sont pas des animaux raisonnables est : $\neg(\text{Animal} \sqcap \text{Raisonnable})$
- $\mathcal{ALC}$ permet de **définir un concept** par **restrictions** sur des **relations** (rôles) :
  - $\forall \text{aEnfant.Femme}$ définit la classe des individus dont tous les enfants sont des femmes
  - $\exists \text{aEnfant.Femme}$ définit la classe des individus dont au moins un enfant est une femme

- **Personne ayant au moins un enfant :**

  $\exists \text{aEnfant.Humain}$

- **Personne qui n'a que des filles** peut être défini ainsi :

  $\forall \text{aEnfant.Femme}$

- **Personne qui n'a pas d'enfant** : on restreint la valeur de la relation aEnfant au concept impossible :

  $\forall \text{aEnfant.}\bot$

  pour appartenir à ce concept, un individu doit avoir tous ses enfants appartenant au concept impossible : il ne peut ainsi avoir d'enfant.

3- Axiomes terminologiques :

- **Les axiomes terminologiques sont de la forme :**
  - **$C \sqsubseteq D$**

  ou

  - **$C \doteq D$**

   avec C et D dénotant des concepts

- Les **définitions de concepts** sont des **axiomes terminologiques** dans lesquels la partie gauche sont des **noms de concepts (ou concepts atomiques)**
- Ils permettent aussi d'**exprimer des propriétés de concepts et de rôles**

Soient A un concept atomique et C un concept composé :

- **Les définitions de concept sont des instructions de la forme :**
  - **$A \doteq C$** (ou **$A \equiv C$**) :

    *lire « **A est par définition égal à C** »*

  ou

  - **$A \sqsubseteq C$** :

    *lire « **A est par définition inclus dans C** » ou « **A est par définition subsumé par C** »*

- **Les définitions de concepts de la forme $A \sqsubseteq C$** sont aussi appelées « définitions primitives de concept » (primitive concept definitions)

Ex :

$$\begin{aligned} \text{Etudiant} &\doteq \text{Personne} \sqcap \exists \text{aNom.string} \\ &\sqcap \exists \text{aAdresse.string} \\ &\sqcap \exists \text{inscritA.ProgrammeFormation} \\ \text{ProgrammeFormation} &\sqsubseteq \exists \text{composeDe.ModuleFormation} \end{aligned}$$

- **Disjonction de concepts** (disjointness) :

  $\text{Homme} \sqsubseteq \neg \text{Femme}$

  l'intersection des individus hommes et des individus femmes est vide
- **Couverture** (coverings) :

  $\top \sqsubseteq \text{Homme} \sqcup \text{Femme}$

  un individu est nécessairement un homme ou une femme
- **Restriction de domaine** (restriction) :

  $\exists \text{aEnfant}.\top \sqsubseteq \text{Parent}$

  un parent a au moins un enfant
- **Plages de restriction ou image** (range restrictions) :

  $\top \sqsubseteq \forall \text{aEnfant.Personne}$

  tout enfant est une personne

  $\top \sqsubseteq \forall \text{aFils.Personne}$

  tout fils est une personne

4- Assertion de concepts et de rôles :

Soit C un concept, R un nom de rôle et a et b des individus.

les **assertions de concepts** sont de la forme :

a : C : ***a appartient à la classe C***

les **assertions de rôles** sont de la forme :

(a, b) : R : ***(a,b) appartient au rôle R***

Ex :

Eric : Etudiant

MasterM6 : Programme

(Eric, MasterM6) : estInscrit

Exemple :

- Une **base de connaissance est une paire** $(\mathcal{T}, \mathcal{A})$ où $\mathcal{T}$ est une TBox et $\mathcal{A}$ une ABox se référant à $\mathcal{T}$
- **Exemple :**

TBox :

Parent $\doteq$ Personne $\sqcap$ $\exists$aEnfant.Personne

Pere $\doteq$ Parent $\sqcap$ Male

GrandParent $\doteq$ Personne $\sqcap$ $\exists$aEnfant.Parent

ABox :

Paul : Personne	Paul : Male
Pierre : Personne	Alice : Personne
(Paul, Pierre) : aEnfant	(Pierre, Alice) : aEnfant

5- Sémantique du langage ALC :

Une interprétation terminologique $\mathcal{I}$ d'une LD **(O, C, R)** consiste en :

- Un **domaine d'interprétation Δ**, ensemble **non vide,** représentant des entités du monde décrit
- Une **fonction d'interprétation** $\mathcal{I}$ associant :
  - à tout individu a ∈ **O**, $\mathcal{I}$ associe un sous-ensemble $\mathcal{I}(a) \in \Delta$
  - à tout concept atomique A ∈ **C**, $\mathcal{I}$ associe un sous-ensemble $\mathcal{I}(A) \subseteq \Delta$
  - à tout rôle atomique R ∈ **R**, $\mathcal{I}$ associe une relation binaire $\mathcal{I}(R) \subseteq \Delta \times \Delta$
- les **autres descriptions possibles de cette fonction** $\mathcal{I}$ sont définies par :

$$\begin{aligned}
\mathcal{I}(\top) &= \Delta \\
\mathcal{I}(\bot) &= \emptyset \\
\mathcal{I}(\neg C) &= \Delta \setminus \mathcal{I}(C) \\
\mathcal{I}(C \sqcap D) &= \mathcal{I}(C) \cap \mathcal{I}(D) \\
\mathcal{I}(C \sqcup D) &= \mathcal{I}(C) \cup \mathcal{I}(D) \\
\mathcal{I}(\forall R.C) &= \{a \in \Delta \mid \forall b.(a, b) \in \mathcal{I}(R) \rightarrow b \in \mathcal{I}(C)\} \\
\mathcal{I}(\exists R.C) &= \{a \in \Delta \mid \exists b.(a, b) \in \mathcal{I}(R) \rightarrow b \in \mathcal{I}(C)\}
\end{aligned}$$

- et **l'hypothèse de nom unique** des individus : ∀ a, b ∈ **O** on a $\mathcal{I}(a) \neq \mathcal{I}(b)$

- On dit que **2 concepts C et D sont équivalents, noté C ≗ D** (C ≡ D), **si on a $\mathcal{I}(C) = \mathcal{I}(D)$,** quelle que soit l'interprétation $\mathcal{I}$

  Ex : l'équivalence :

  ∀aEnfant.Femme⊓∀aEnfant.Médecin ≗ ∀aEnfant.(Femme⊓Médecin)

  l'ensemble des personnes dont tous les enfants sont des femmes, et dont tous les enfants sont des médecins, est exactement le même que (équivalent à) l'ensemble des personnes dont tous les enfants sont à la fois femme et médecin.

  Tout énoncé de la forme C ≗ D est appelé dans la TBox **définition**

- **Par définition on a les équivalences suivantes :**

$$\begin{aligned}
\neg\top &\doteq \bot \\
\neg\bot &\doteq \top \\
C \sqcap \neg C &\doteq \bot \\
C \sqcup \neg C &\doteq \top
\end{aligned}$$

- On dit que **le concept C inclus le concept D**, **noté** $C \sqsubseteq D$, ssi on a $\mathcal{I}$(C) ⊆ $\mathcal{I}$(D), **quelle que soit l'interprétation** $\mathcal{I}$

- **Par définition :**

soit le concept universel ⊤ représentant tous les individus du monde représenté, et le **concept impossible** ⊥

  - **pour tout concept C**, on a **l'axiome :**

    $C \sqsubseteq \top$

  - **pour un concept C *impossible***, cad qu'aucun individu ne peut appartenir à ce concept, on a **l'axiome :**

    $C \sqsubseteq \bot$

Exemple :

Soit :

Personne et Male : 2 concepts (2 noms de concepts)

aEnfant : un role (un nom de role)

Soit l'interprétation terminologique $\mathcal{I}$ **sur** Δ **:**

Δ = {Paul, Pierre, Eric, Alice, Lila}

$\mathcal{I}$ (Personne) = {Paul, Pierre, Eric, Alice}

$\mathcal{I}$ (Male) = {Paul, Pierre, Eric }

$\mathcal{I}$ (aEnfant) = {(Paul, Pierre), (Pierre, Alice), (Pierre, Eric)}

On en déduit :

$\mathcal{I}$ (¬Male) = {Alice, Lila }

$\mathcal{I}$ (Personne ⊓ ¬Male) = {Alice}

$\mathcal{I}$ (∃aEnfant.Male) = {Paul, Pierre}

6- Correspondance entre la logique ALC et la logique des prédicats :

- Une **correspondance** existe entre la **LD** $\mathcal{AL}$ et la **logique des prédicats du premier ordre** (First Order Logic - FOL) [Baader et Nutt, 2003] :
  - un **concept atomique A** correspond à un **prédicat unaire** $\phi A(x)$
  - un **rôle R** à un **prédicat binaire** $\phi R(x,y)$
  - un **individu** correspond à une **constante**
  - un **concept composé** à une **formule avec 1 variable libre** $\phi C(x)$.
- **avec les règles de passage suivantes :**

Constructeurs AL		*Logique des prédicats (FOL)*
$\phi_{\neg C}(x)$	$=$	$\neg\phi_C(x)$
$\phi_{C\sqcap D}(x)$	$=$	$\phi_C(x) \wedge \phi_D(x)$
$\phi_{C\sqcup D}(x)$	$=$	$\phi_C(x) \vee \phi_D(x)$
$\phi_{\exists R.C}(y)$	$=$	$\exists x.R(y,x) \wedge \phi_C(x)$
$\phi_{\forall R.C}(y)$	$=$	$\forall x.R(y,x) \rightarrow \phi_C(x)$

Exemple :

- Tout employé travaille pour une compagnie :
  $\forall E.\exists C.(Employe(E) \rightarrow travaillePour(E,C) \wedge Compagnie(C)$
  $Employe \sqsubseteq \exists travaillePour.Compagnie$
- Une compagnie a au moins un employé :
  $\forall C.(Compagnie(C) \rightarrow \exists E.(travaillePour(E, C)))$
  $Compagnie \sqsubseteq \exists travaillePour^{-}.Employe$
- Un manager est un employé :
  $\forall X.(Manager(X) \rightarrow Employe(X))$
  $Manager \sqsubseteq Employe$
- Un manager ne doit pas travailler pour plus que 2 compagnies :
  $\forall M.\forall X.\forall Y.\forall Z.(Manager(M) \wedge travaillePour(M,X)\ travaillePour(M, Z) \rightarrow (X = Y) \vee (X = Z) \vee (Y = Z))$
  $Manager \sqsubseteq \leq 2\ travaillePour.Compagnie$
- Une compagnie ne peut pas être un employé en même temps :
  $\forall X.(Compagnie(X) \rightarrow \neg Employe(X))$
  $\perp \equiv Compagnie \sqcap Employe$
- Pour tout employé travaillant pour une compagnie, on peut automatiquement déduire que la compagnie l'emploie :
  $\forall E.\forall C.(travaillePour(E, C) \rightarrow employer(C, E)$
  $travaillePour^{-} \equiv employer$

IV- Les extensions de la logique AL

Différentes façons d'étendre $\mathcal{AL}$ [Baader, 2003] :

- **Ajouter de constructeurs de concepts et de rôles :**
  - $\mathcal{O}$: permet la description de concepts par l'énumération d'individus nommés,
  - $\mathcal{U}$: permet l'union de concepts arbitraires,
  - ε : permet la quantification existentielle complète,
  - $\mathcal{C}$: permet la négation complète,
  - $\mathcal{I}$: permet les rôles inverses et l'inclusion entre rôles,
  - $\mathcal{F}$, $\mathcal{Q}$, $\mathcal{N}$: 3 variantes de la contrainte de cardinalité sur rôle.
- **Enoncer des contraintes sur l'interprétation des rôles ($\mathcal{NR}+$):**
  - spécification d'un **ensemble de rôles transitifs** $\mathcal{NR}+$, constitue $\mathcal{R}+$, une extension par ajout de contraintes sur l'interprétation des rôles (désignée par la lettre $\mathcal{R}+$) : permet des rôles transitifs tels que ancêtreDe ou frèreDe.
- **Extension de types primitifs ($\mathcal{D}$) et de rôles à valeurs primitives ($\mathcal{U}$) :**
  - $\mathcal{D}$: Ajout à $\mathcal{AL}$ d'un **second domaine d'interprétation** $\Delta^{I}\mathcal{D}$ disjoint avec Δ^{I}, représentant l'ensemble des valeurs de type primitif (entiers, chaînes de caractères, entiers positifs ...), dont les éléments sont des individus primitifs
  - $\mathcal{U}$ **: un nouveau type de rôle** défini comme une relation binaire sur $\Delta^{I}\mathcal{D}\times\Delta^{I}\mathcal{D}$, appelé rôles à valeurs primitives. La lettre $\mathcal{U}$ représente l'ensemble de ces rôles, permettant par la spécification d'assertions de rôle telles que u(a, 205006007) et v(a, "Jean Jacques") où u, v$\in\mathcal{U}$.

- colonne 1 : lettre désignant le constructeur,
- colonne 2 : sa syntaxe d'utilisation
- colonne 3 : sa sémantique.

$[\mathcal{O}]$	$\{a_1, a_2, ..., a_n\}$	$\{a_1^{\mathcal{I}}, a_2^{\mathcal{I}}, ..., a_n^{\mathcal{I}}\}$
$[\mathcal{U}]$	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$[\mathcal{E}]$	$\exists R.C$	$\{a \in \Delta^{\mathcal{I}} \| \{\exists b.(a,b) \in R^{\mathcal{I}}\} \wedge b \in C^{\mathcal{I}}\}$
$[\mathcal{C}]$	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$[\mathcal{I}]$	R_1^{-1}	$\{(y,x) \| (x,y) \in R_1^{\mathcal{I}}\}$
$[\mathcal{H}]$	$R_1 \sqsubseteq R_2$	$R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
$[\mathcal{F}]$	$= 1R$	$\{x \in \Delta^{\mathcal{I}} \| \|\{y \in \Delta^{\mathcal{I}} \| (x,y) \in R^{\mathcal{I}}\}\| = 1\}$
	$\geq 2R$	$\{x \in \Delta^{\mathcal{I}} \| \|\{y \in \Delta^{\mathcal{I}} \| (x,y) \in R^{\mathcal{I}}\}\| \geq 2\}$
$[\mathcal{N}]$	$\geq nR$	$\{a, b \in \Delta^{\mathcal{I}} \| \|(a,b) \in R^{\mathcal{I}}\}\| \geq n\}$
	$\leq nR$	$\{a, b \in \Delta^{\mathcal{I}} \| \|(a,b) \in R^{\mathcal{I}}\}\| \leq n\}$
	$= nR$	$\{a, b \in \Delta^{\mathcal{I}} \| \|(a,b) \in R^{\mathcal{I}}\}\| = n\}$
$[\mathcal{Q}]$	$\geq nR.C$	$\{a, b \in \Delta^{\mathcal{I}} \| \|(a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}\| \geq n\}$
	$\leq nR.C$	$\{a, b \in \Delta^{\mathcal{I}} \| \|(a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}\| \leq n\}$
	$= nR.C$	$\{a, b \in \Delta^{\mathcal{I}} \| \|(a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}\| = n\}$

- $\mathcal{O}$ **: Description de concepts par l'énumération d'individus nommés :**
  - individus nommés $\{a_1, a_2, \ldots a_n\}$, et interprétation $\{a^{\mathcal{I}}_1, a^{\mathcal{I}}_2, \ldots a^{\mathcal{I}}_n\}$,
- $\mathcal{U}$ **: Union de concepts arbitraires :**
  - pour représenter l'ensemble des individus qui appartiennent soit à la classe C, soit à la classe D, on écrit la description **C ⊔ D**, et dont l'interprétation est la suivante : $\mathcal{I}(C \sqcup D) = \mathcal{I}(C) \cup \mathcal{I}(D)$
  - très pratique dans les **restrictions de relations** :

    Ex : représenter un magasin qui ne vend que des chats et des chiens :

    Magasin ⊓ ∀vend.(Chat ⊔ Chien)

    pour appartenir à cette classe, une entité doit appartenir à la fois à l'ensemble des magasins et l'ensemble des entités telles que si elles vendent quelque chose, il s'agit nécessairement d'un chat ou d'un chien.

- $\mathcal{C}$ **: Négation complète :** $\mathcal{ALC}$
  - ¬ C avec C concept atomique ou composé

- ε **: Quantification existentielle (∃) complète :** $\mathcal{ALC}$
  - Limitation dans $\mathcal{AL}$: ∃ permet de spécifier qu'une entité doit avoir au moins une relation avec un autre objet, mais pas possible de spécifier la classe de cet autre objet

    Ex : définir la classe de ceux qui possèdent au moins un chien.

    ⇒ ***Ajout de la quantification existentielle complète*** **∃R.C**, dont l'interprétation est : $\mathcal{I}(\exists R.C) = \{a \in \Delta \mid \exists b.(a, b) \in \mathcal{I}(R) \wedge b \in \mathcal{I}(C)\}$

    Ex : classe des gens possèdant au moins un chien : **∃possède.Chien**

- $\mathcal{I}$ **: Rôles inverses et l'inclusion entre rôles :**
  - permet de définir un rôle qui est l'inverse d'un autre rôle.

    Ex : si Alice regarde Paul, Paul est regardé par Alice. Si on utilise 2 relations regarde et estRegardéPar, on veut que tout fait de la forme regarde(x, y) implique nécessairement le fait estRegardéPar(y, x)

    ⇒ ***Ajout du constructeur d'inversion*** **estRegardéPar ≡ regarde⁻**, et dont l'interprétation est $\mathcal{I}(R^-) = \{(y, x) \mid (x, y) \in \mathcal{I}(R)\}$

- $\mathcal{F}$ **: Constructeur de fonctions**
  - permet de spécifier qu'un rôle (relation) est une fonction : aucune entité ne peut être reliée à plus d'une autre entité par cette relation.

    Ex : la relation mariéAvec, qui nous permettrait de définir le concept HommeMarié : **HommeMarié ≡ Homme ⊓ ∃mariéAvec.⊤**

    ⇒ Pour empêcher une instance de cette classe d'être mariée avec plus d'une personne, il faut ***spécifier que la relation mariéAvec est une fonction***, en écrivant un axiome de la forme ***Fun(R)*** indiquant que le rôle R est une fonction.

- $\mathcal{N}$ **: Restriction de cardinalité**
  - pour représenter des concepts comme l'ensemble de ceux qui ont au moins 2 enfants, ou qui ont au plus 4 enfants

    ⇒ ***Ajout de 2 constructeurs ≤ n R et ≥ n R :***

    Ex : concept de père qui a exactement 2 enfants :

    Homme ⊓ ≥ 2 aEnfant ⊓ ≤ 2 aEnfant

- $\mathcal{Q}$ **: Restriction de cardinalité qualifiée**
  - Les 2 contructeurs **≤ nR** et **≥ nR** précédents ne permettent pas d'imposer le nombre minimal ou maximal d'entités d'une classe spécifique auquel on peut être lié par une relation

    Ex : dans une logique $\mathcal{ALN}$, si on a la relation possède, on ne peut décrire la classe des gens qui possèdent plus de 2 chiens

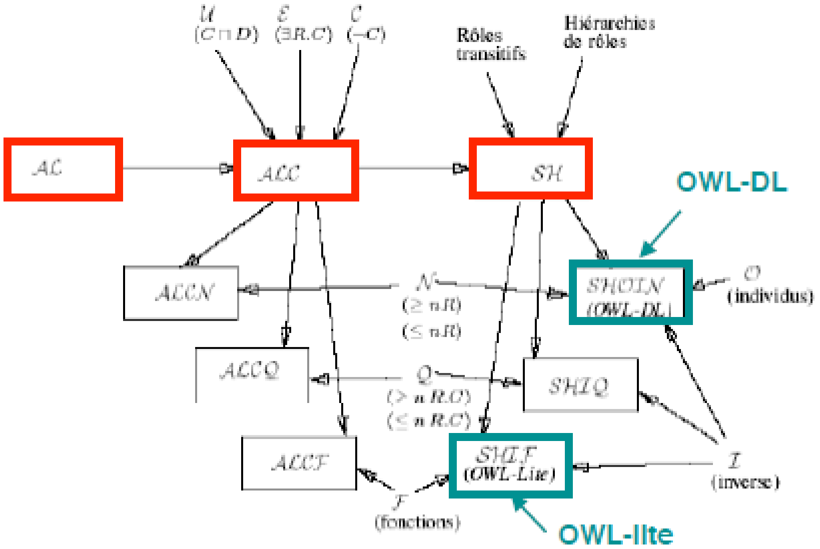
  - Pour cela, il faut les **constructeurs de restriction de cardinalité qualifiée** $\mathcal{Q}$ suivants : **≤ nR.C** et **≥ nR.C**

    Ex : classe des gens qui possèdent 2 chiens ou plus :

    ≥ 2 possède.Chien

- **De façon générale, pour chaque constructeur ajouté**, il faut rajouter la lettre correspondante au nom de la logique originale
- **La LD $\mathcal{ALU}\varepsilon$** est obtenue en rajoutant à la LD $\mathcal{AL}$:
  - l'union ($\mathcal{U}$) et
  - la quantification existentielle complète (ε)
- **La LD $\mathcal{ALC}$ équivaut à la LD $\mathcal{ALU}\varepsilon$:** car l'union et la quantification existentielle complète s'expriment par la négation complète et inversement : $C \sqcup D \equiv \neg(\neg C \sqcap \neg D)$ et $\exists R.C \equiv \neg \forall R.\neg C$ [Baader, 2003]
- **La LD $\mathcal{SH}$:** LD plus expressive obtenue en rajoutant à la LD $\mathcal{ALC}$ les possibilités :
  - d'établir qu'une relation est une sous-propriété d'une autre relation et
  - de définir une relation transitive
  - on écrira ***Tr(R)*** pour signifier qu'une relation R est transitive, et $R_1 \sqsubseteq R_2$ pour signifier que R_1 est une sous-propriété de R_2.

Les principales familles des DL peuvent être illustrées par la figure suivante :



V- Inférences dans les logiques de description :

Comme la base de connaissances en logique de description est composée d'une Abox et d'une Tbox, divers mécanismes de raisonnement ont été développés pour les connaissances terminologiques et les connaissances assertionnelles.

1- Mécanismes pour la Tbox :

4 propriétés intéressantes à prouver pour une TBox [Baader & Nutt, 2003]:

- **Satisfiabilité** : un concept **C** d'une terminologie $\mathcal{T}$ est **satisfiable** ssi il **existe un modèle $\mathcal{I}$ de $\mathcal{T}$ tel que $C^{\mathcal{I}} \neq \emptyset$**
- **Subsomption** : un concept **C** est **subsumé par** un concept **D** pour une terminologie $\mathcal{T}$ **ssi $C^{\mathcal{I}} \sqsubseteq D^{\mathcal{I}}$** pour tout modèle $\mathcal{I}$ de $\mathcal{T}$
- **Équivalence** : un concept **C** est **équivalent** à un concept **D** pour une terminologie $\mathcal{T}$ **ssi $C^{\mathcal{I}} \equiv D^{\mathcal{I}}$**, pour tout modèle $\mathcal{I}$ de $\mathcal{T}$
- **Disjonction** (disjointness) : des concepts **C** et **D** sont **disjoints** par rapport à la terminologie $\mathcal{T}$ **ssi $C^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset$**, pour tout modèle $\mathcal{I}$ de $\mathcal{T}$.

- Résoudre des **problèmes d'inférence** dans la TBox, c'est **prouver une de ces 4 propriétés**
- Résoudre des problèmes d'inférence **se réduit généralement** à prouver soit un **subsomption**, soit une **satisfiabilité** (moteurs d'inférence actuels) :
  - **Réduction des problèmes d'inférence d'une TBox à des problèmes de subsomption :**
    - **C** est **insatisfiable** ⇔ **C** est **subsumé** par $\perp$
    - **C** et **D** sont **équivalents** ⇔ **C** est **subsumé** par **D**, et **D** par **C**
    - **C** et **D** sont **disjoints** ⇔ $C \sqcap D$ est **subsumé** par $\perp$
  - **Réduction des problèmes d'inférence d'une TBox à des problèmes de satisfiabilité :**
    - **C** est **subsumé** par **D** ⇔ $C \sqcap \neg D$ est **insatisfiable**
    - **C** et **D** sont **équivalents** ⇔ $C \sqcap \neg D$ et $\neg C \sqcap D$ sont **insatisfiables**
    - **C** et **D** sont **disjoints** ⇔ $C \sqcap D$ est **insatisfiable**

2-Mécanismes pour la Abox :

Le **niveau factuel** (ABox) comprend **4 principaux problèmes d'inférence** [Baader et Nutt, 2003] :

- **Cohérence** : Une ABox $\mathcal{A}$ est **cohérente** par rapport à une TBox $\mathcal{T}$ **ssi il existe un modèle $\mathcal{I}$ de $\mathcal{A}$ et $\mathcal{T}$**
- **Vérification d'instance** : Vérifier par inférence **si une assertion *C(a)* est vraie pour tout modèle** $\mathcal{I}$ d'une ABox $\mathcal{A}$ et d'une TBox $\mathcal{T}$
- **Vérification de rôle** : Vérifier par inférence **si une assertion *R(a, b)* est vraie pour tout modèle** $\mathcal{I}$ d'une ABox $\mathcal{A}$ et d'une TBox $\mathcal{T}$
- **Problème de récupération** : Pour une ABox $\mathcal{A}$, un concept ***C*** d'une terminologie $\mathcal{T}$, **Inférer les individus** $a^{\mathcal{I}}_1 \ldots a^{\mathcal{I}}_n \in C^{\mathcal{I}}$ pour tout modèle $\mathcal{I}$ d'une ABox $\mathcal{A}$ et d'une TBox $\mathcal{T}$

3- Algorithmes de raisonnement pour les LD :

Il existe deux principales classes d'algorithmes de raisonnement pour la LD

3-1 Algorithme de vérification de subsomption :

- réduisent les problèmes d'inférence à des problèmes de **subsomption,** ceci en **temps polynomial**
- un processus de normalisation produit les **formes normales de concepts** définis qui sont ensuite effectivement **comparées** à l'aide de **règles de comparaisons structurelles**
- **Idée de l'algorithme**: si 2 expressions de concepts sont composées de sous-expressions, comparer séparément une sous-expression d'un concept avec toutes les sous-expressions des autres concepts.
- **Ces algorithmes :**
  - **ont une correction facile à vérifier, mais une complétude difficile à démontrer**
  - ne s'appliquent qu'à des **LD peu expressives**, sans quoi ils sont incomplets, cad qu'ils sont incapables de prouver certaines formules vraies.

3-2 Algorithmes de vérification de satisfiabilité à base de tableaux :

- **années 1990** : nouveaux **algorithmes de vérification de satisfiabilité à base de tableaux** (tableau-based algorithms).
- Ces algorithmes réduisent ainsi les problèmes d'inférence dans les LD à des problèmes de **satisfiabilité**
- **Idée de l'algorithme** de la **méthode des tableaux sémantiques** : dans la LD considérée devant disposer de la négation (¬), la question *le concept D subsume-t-il le concept C ?* est transformée en *l'expression* $C \sqcap \neg D$ *est-elle non satisfiable ?*
- **Ces algorithmes :**
  - raisonnent sur des **LD dites expressives ou très expressives**,
  - **en temps exponentiel**, mais en pratique, le comportement des algorithmes est souvent **acceptable**.
  - **Ils ont une complexité et décidabilité plus facile**
  - La forte expressivité des LD traitées a ouvert la porte à de nouvelles applications telles que le **Web sémantique.**

- Une **procédure de tableau sémantique peut être considérée** comme une **procédure pour construire une interprétation satisfaisante de l'assertion d'un concept donné.**
- Les **dérivations** peuvent être établies sous la forme d'un **arbre**, dont les arêtes représentent la succession des rôles entre les éléments du domaine d'interprétation Δ

Afin d'appliquer la méthode des tableaux sémantiques, il va falloir apporter les transformations à la TBOX :

a- Elimination de la TBOX

Afin d'optimiser le processus d'inférence en utilisant la méthode des tableaux sémantiques et dans le cas où il n'existe aucun cycle dans les définitions des concepts complexes, il faut remplacer tous les concepts complexes d'une formule par leur définition dans la terminologie :

Élimination de la TBox

en absence de définitions circulaires

Adaptation de Figure 2.3 de F. Baader, W. Nutt

$$
\begin{array}{rcl}
\texttt{Woman} & \equiv & \texttt{Person} \sqcap \texttt{Female} \\
\texttt{Man} & \equiv & \texttt{Person} \sqcap \neg\texttt{Female} \\
\texttt{Mother} & \equiv & \texttt{Person} \sqcap \texttt{Female} \sqcap \exists\texttt{hasChild.Person} \\
\texttt{Father} & \equiv & \texttt{Person} \sqcap \neg\texttt{Female} \sqcap \exists\texttt{hasChild.Person} \\
\texttt{Parent} & \equiv & \texttt{Person} \sqcap \exists\texttt{hasChild.Person} \\
\texttt{Grandmother} & \equiv & \texttt{Person} \sqcap \texttt{Female} \sqcap \exists\texttt{hasChild.Person} \sqcap \\
 & & \exists\texttt{hasChild.(Person} \sqcap \exists\texttt{hasChild.Person)} \\
\texttt{MotherWithManyChildren} & \equiv & \texttt{Person} \sqcap \texttt{Female} \sqcap \geq 3\ \texttt{hasChild} \\
\texttt{MotherWithoutDaughter} & \equiv & \texttt{Person} \sqcap \texttt{Female} \sqcap \\
 & & \forall\texttt{hasChild.(}\neg\texttt{Person} \sqcup \neg\texttt{Female)}
\end{array}
$$

b- Transformation de la TBOX sous la Forme Normale Négative (NNF)

- Soit C un concept arbitraire dans $\mathcal{ALC}$
- On dit que C est en **forme normale négative (NNF) ssi ¬ apparaît seulement immédiatement devant le nom d'un concept**

La NNF peut être obtenue en appliquant les règles suivantes :

$\neg\neg C \Rightarrow_{NNF} C$	
$\neg\top \Rightarrow_{NNF} \bot$	$\neg\bot \Rightarrow_{NNF} \top$
$\neg(C \sqcup D) \Rightarrow_{NNF} \neg C \sqcap \neg D$	$\neg(C \sqcap D) \Rightarrow_{NNF} \neg C \sqcup \neg D$
$\neg\forall R.C \Rightarrow_{NNF} \exists R.\neg C$	$\neg\exists R.C \Rightarrow_{NNF} \forall R.\neg C$

- Un **calcul de tableau sémantique** est une **procédure de preuve formelle**, existant dans plusieurs DL, avec les mêmes caractéristiques :
  - Un tableau sémantique est un **système de réfutation** : étant donné un système initial de contraintes ou tableau T, il tente soit de montrer que **T est non satisfiable**, soit de **construire une interprétation satisfaisante**
  - Un tableau est une suite de séries d'affirmations construites selon certaines **règles d'inférence**, et est généralement énoncé sous la forme d'un **arbre :**
    - Les règles d'inférence stipulent comment un tableau T génère un nouveau tableau dans lequel une branche est transformée en n nouvelles branches.
    - Cela se fait en élargissant la branche à sa feuille, en créant jusqu'à n nœuds successeurs
    - Chacun des nouveaux nœuds contient de nouvelles affirmations

c- Algorithme de déduction par les tableaux sémantiques :

Étapes

1. Faire la négation d'un énoncé
2. Phase d'expansion du tableau. On applique les règles d'expansion (de "déconstruction")
   - chaque branche est une conjonction
   - certaines règles vont créer de nouvelles branches
   - on arrête de développer une branche quand aucune règle n'est applicable
   - les feuilles sont des axiomes
3. On essaie de "fermer" les branches (trouver des contradictions)

Arrêt

- Si on ferme toutes les branches
  - La négation de l'énoncé est impossible
  - Donc l'énoncé est valide
- Si une branche reste ouverte
  - Elle donne un exemple de la négation de l'énoncé
  - Donc au moins un contre-exemple de l'énoncé existe
  - Donc l'énoncé n'est pas valide

d- Règles d'expansion :

Description formelle des règles

Soit un tableau et $\mathcal{A}$ un chemin complet qui va de la racine de l'arbre à une feuille. Le tableau 1 fournit la liste des règles d'expansion pour la logique $\mathcal{ALCN}$.

Règle	Condition	Action
règle-$\sqcap$	$\mathcal{A}$ contient $(C_1 \sqcap C_2)(x)$ et ne contient pas déjà les deux énoncés $C_1(x)$ et $C_2(x)$.	On ajoute à $\mathcal{A}$ les énoncés $C_1(x)$ et $C_2(x)$
règle-$\sqcup$	$\mathcal{A}$ contient $(C_1 \sqcup C_2)(x)$ et ne contient aucun des deux énoncés $C_1(x)$ et $C_2(x)$.	On ajoute à $\mathcal{A}$ le branchement suivant : $C_1(x)$ $C_2(x)$
règle-$\exists$	$\mathcal{A}$ contient $(\exists R.C)(x)$ et il n'existe aucun individu z tel que $R(x, z)$ et $C(z)$ sont aussi dans $\mathcal{A}$	On ajoute $R(x, y)$ et $C(y)$ à $\mathcal{A}$, où y est un nom d'individu qui n'existe pas déjà dans $\mathcal{A}$
règle-$\forall$	$\mathcal{A}$ contient $(\forall R.C)(x)$ et $R(x, y)$, mais ne contient pas $C(y)$	On ajoute $C(y)$ à $\mathcal{A}$
règle-$\geq$	$\mathcal{A}$ contient $(\geq n\ R)(x)$ et il n'y a pas dans $\mathcal{A}$ des individus $z_1, \ldots, z_n$ qui sont tous distincts (c'est-à-dire qu'on doit avoir explicitement dans $\mathcal{A}$ l'énoncé $z_i \neq z_j$ pour chaque paire possible avec cet ensemble d'individus) et qui sont tels que $\mathcal{A}$ contient la relation $R(x, z_i)$ pour tous ces individus $(1 \leq i \leq n)$.	Soit un ensemble de n individus dénotés par $y_1, \ldots, y_n$, qui sont des noms qui n'existent pas dans $\mathcal{A}$. On ajoute à $\mathcal{A}$ les énoncés $y_i \neq y_j$ pour chaque paire possible avec cet ensemble, ainsi que les énoncés $R(x, y_i)$ pour $(1 \leq i \leq n)$.
règle-$\leq$	$\mathcal{A}$ contient $(\leq n\ R)(x)$ et les énoncés $R(x, y_1), \ldots, R(x, y_{n+1})$. Il n'existe aucune identité $y_i = y_j$ dans $\mathcal{A}$ pour $(1 \leq i \leq n+1)$, $(1 \leq j \leq n+1), i \neq j$	Pour chaque paire possible (y_i, y_j) d'individus parmi y_i, y_{n+1}, on ajoute une nouvelle branche avec $y_i = y_j$.

– Règles de tableau pour la logique $\mathcal{ALCN}$

On ajoute la contradiction $\square$ à la fin d'un chemin $\mathcal{A}$ lorsqu'une des situations suivantes se présente :

— $\mathcal{A}$ contient à la fois un énoncé de la forme $C(x)$ et son complément $\neg C(x)$

— $\mathcal{A}$ contient un énoncé de la forme $C(x)$, un énoncé de la forme $\neg C(y)$ et une des identités $x = y$ ou $y = x$.

— $\mathcal{A}$ contient un énoncé de la forme $\perp(x)$ (qui, rappelons-le, signifie que l'individu x ne peut pas exister)

Ainsi, pour faire une preuve, on crée d'abord l'énoncé dont on veut montrer l'insatisfaisabilité. Ensuite, en appliquant un séquence de règles, on ajoute de nouveaux énoncés. On crée ainsi ce qu'on appelle un *tableau*, qui est en fait un arbre, à cause des branchements qui peuvent y apparaître, comme dans l'exemple précédent.

Lorsqu'on a ajouté la contradiction $\square$ dans une branche, on dit qu'elle est fermée et on cesse d'en faire l'expansion. Lorsque toutes les branches d'un tableau sont fermées, on dit que le tableau est fermé et l'énoncé initial est considéré insatisfaisable. Si un tableau contient au moins une branche ouverte et qu'on ne réussit plus à appliquer des règles pour en faire l'expansion, le processus s'arrête et on conclut que l'énoncé initial n'est pas insatisfaisable.

Exemple :

Algorithme de déduction par tableau

Description de l'algorithme

La meilleure méthode pour faire des inférences avec la logique descriptive est la méthode du tableau. Supposons que nous voulions prouver $C \sqsubseteq D$. Le principe de la démonstration est le suivant. On nie le fait à prouver et on tente de démontrer que ce nouveau fait est insatisfaisable. Dans notre cas, cela revient à démontrer que $C \sqcap \neg D$ est insatisfaisable.

Supposons par exemple que l'on veut démontrer que la classe $\exists\mathsf{possède}.(\mathsf{Livre} \sqcap \mathsf{Antiquité})$ est subsumée par la classe $(\exists\mathsf{possède}.\mathsf{Livre} \sqcap \exists\mathsf{possède}.\mathsf{Antiquité})$, c'est-à-dire :

$$\exists\mathsf{possède}.(\mathsf{Livre} \sqcap \mathsf{Antiquité}) \sqsubseteq (\exists\mathsf{possède}.\mathsf{Livre} \sqcap \exists\mathsf{possède}.\mathsf{Antiquité})$$

En d'autres mots, on veut prouver que si quelqu'un possède un livre ancien, il appartient nécessairement à la classe de ceux qui possèdent au moins un livre et au moins une antiquité. Pour ce faire il faut prouver que la classe suivante est insatisfaisable :

$$\exists\mathsf{possède}.(\mathsf{Livre} \sqcap \mathsf{Antiquité}) \sqcap \neg(\exists\mathsf{possède}.\mathsf{Livre} \sqcap \exists\mathsf{possède}.\mathsf{Antiquité})$$

La première étape avant de commencer la preuve consiste à normaliser la description de manière à que toutes les négations se trouvent immédiatement à côté d'un identificateur de classe. On y arrive en appliquant les règles d'équivalence présentées à la section 3 :

$$\begin{array}{l}
\exists\mathsf{possède}.(\mathsf{Livre} \sqcap \mathsf{Antiquité}) \sqcap \neg(\exists\mathsf{possède}.\mathsf{Livre} \sqcap \exists\mathsf{possède}.\mathsf{Antiquité}) \\
\exists\mathsf{possède}.(\mathsf{Livre} \sqcap \mathsf{Antiquité}) \sqcap (\neg\exists\mathsf{possède}.\mathsf{Livre} \sqcup \neg\exists\mathsf{possède}.\mathsf{Antiquité}) \\
\exists\mathsf{possède}.(\mathsf{Livre} \sqcap \mathsf{Antiquité}) \sqcap (\forall\mathsf{possède}.\neg\mathsf{Livre} \sqcup \forall\mathsf{possède}.\neg\mathsf{Antiquité})
\end{array}$$

Pour faire notre preuve, on suppose un individu arbitraire, que l'on dénomme a, et qui est une instance de cette classe :

$$(1) \quad (\exists\mathsf{possède}.(\mathsf{Livre} \sqcap \mathsf{Antiquité}) \sqcap (\forall\mathsf{possède}.\neg\mathsf{Livre} \sqcup \forall\mathsf{possède}.\neg\mathsf{Antiquité}))(a)$$

(2) $(\exists \text{possède}.(\text{Livre} \sqcap \text{Antiquité}))(a)$
(3) $(\forall \neg \text{possède}.\text{Livre} \sqcup \forall \neg \text{possède}.\text{Antiquité})(a)$

On sait maintenant que notre individu a possède un livre qui est une antiquité. On ne sait pas quel est cet objet, mais on sait qu'il existe. Appelons b ce nouvel individu. On peut alors écrire :

(4) $\text{possède}(a, b)$
(5) $(\text{Livre} \sqcap \text{Antiquité})(b)$

Par un raisonnement similaire à ce que nous avons fait précédemment, on peut ajouter les faits suivants :

(6) $\text{Livre}(b)$
(7) $\text{Antiquité}(b)$

Considérons maintenant le fait (3) que nous avons ajouté plus tôt. Il nous indique que a appartient à la classe de ceux qui ne possèdent aucun livre ou qui ne possèdent aucune antiquité. Autrement dit, au moins un des deux énoncés suivants doit être vrai :

(8) $(\forall \text{possède}.\neg \text{Livre})(a)$
(9) $(\forall \text{possède}.\neg \text{Antiquité})(a)$

Supposons que (8) est vrai. On sait déjà que a possède b. L'objet ne peut donc pas être un livre :

(10) $\neg \text{Livre}(b)$

Or, ceci contredit l'énoncé (6). Similairement, conformément à l'énoncé (9), on peut déduire :

(11) $\neg \text{Antiquité}(b)$

Ceci contredit l'énoncé (7). Donc, quelle que soit la possibilité considérée, on arrive toujours à une contradiction, ce qui signifie que notre énoncé initial (1) est insatisfaisable. La subsomption est donc prouvée. Si l'on désigne par $\square$ la contradiction, on peut représenter la preuve par un arbre, où chaque branchement représente des alternatives :

$$
\begin{array}{ll}
(1) & (\exists \text{possède}.(\text{Livre} \sqcap \text{Antiquité}) \sqcap (\forall \text{possède}.\neg \text{Livre} \sqcup \forall \text{possède}.\neg \text{Antiquité}))(a) \\
(2) & (\exists \text{possède}.(\text{Livre} \sqcap \text{Antiquité}))(a) \\
(3) & (\forall \neg \text{possède}.\text{Livre} \sqcup \forall \neg \text{possède}.\text{Antiquité})(a) \\
(4) & \text{possède}(a, b) \\
(5) & (\text{Livre} \sqcap \text{Antiquité})(b) \\
(6) & \text{Livre}(b) \\
(7) & \text{Antiquité}(b)
\end{array}
$$

$$
\begin{array}{llcll}
(8) & (\forall \text{possède}.\neg \text{Livre})(a) & \quad & (9) & (\forall \text{possède}.\neg \text{Antiquité})(a) \\
(10) & \neg \text{Livre}(b) & & (11) & \neg \text{Antiquité}(b) \\
 & \square & & & \square
\end{array}
$$

Les quatre premières règles d'expansion de tableau ont déjà été vues dans l'exemple de preuve ci-haut. Il nous reste maintenant à voir des exemples d'application des deux dernières. Supposons d'abord une terminologie qui contient la définition suivante :

$$\text{Parent} \equiv \exists \text{aEnfant}.\top$$

Nous tenterons maintenant de démontrer que quelqu'un qui a plus de deux enfants est un parent, c'est-à-dire la subsomption suivante :

$$\geq 2 \ \text{aEnfant} \sqsubseteq \text{Parent}$$

Pour ce faire, il faut d'abord écrire le fait dont on veut démontrer l'insatisfaisabilité :

$\geq 2\ \mathsf{aEnfant} \sqcap \neg \mathsf{Parent}$

Puis il faut éliminer la TBox :

$\geq 2\ \mathsf{aEnfant} \sqcap \neg(\exists \mathsf{aEnfant}.\top)$

Finalement, il faut normaliser l'énoncé :

$\geq 2\ \mathsf{aEnfant} \sqcap \forall \mathsf{aEnfant}.\bot$

On peut maintenant commencer la preuve par tableau qui, au moment de l'application de la règle-$\geq$, sera dans l'état suivant :

$$\begin{array}{ll}
(1) & (\geq 2\ \mathsf{aEnfant} \sqcap \forall \mathsf{aEnfant}.\bot)(a) \\
(2) & \quad (\geq 2\ \mathsf{aEnfant})(a) \\
(3) & \quad (\forall \mathsf{aEnfant}.\bot)(a)
\end{array}$$

Ici, d'après l'énoncé (2), on sait que l'individu a doit avoir au moins deux enfants. Or, dans le tableau, ceux-ci ne sont pas explicités. C'est exactement ce que la règle-$\geq$ nous permet de faire :

$$\begin{array}{ll}
(1) & (\geq 2\ \mathsf{aEnfant} \sqcap \forall \mathsf{aEnfant}.\bot)(a) \\
(2) & \quad (\geq 2\ \mathsf{aEnfant})(a) \\
(3) & \quad (\forall \mathsf{aEnfant}.\bot)(a) \\
(4) & \quad\ \mathsf{aEnfant}(a, b) \\
(5) & \quad\ \mathsf{aEnfant}(a, c) \\
(6) & \qquad b \neq c
\end{array}$$

La prochaine étape d'expansion applique la règle-$\forall$ à l'énoncé (3), ce qui entraîne l'ajout de deux énoncés spécifiant que les individus b et c ne peuvent pas exister, d'où l'introduction de la contradiction $\Box$ qui ferme le tableau :

$$\begin{array}{ll}
(1) & (\geq 2\ \mathsf{aEnfant} \sqcap \forall \mathsf{aEnfant}.\bot)(a) \\
(2) & \quad (\geq 2\ \mathsf{aEnfant})(a) \\
(3) & \quad (\forall \mathsf{aEnfant}.\bot)(a) \\
(4) & \quad\ \mathsf{aEnfant}(a, b) \\
(5) & \quad\ \mathsf{aEnfant}(a, c) \\
(6) & \qquad b \neq c \\
(7) & \qquad \bot(b) \\
(8) & \qquad \bot(c) \\
(9) & \qquad \Box
\end{array}$$

Supposons maintenant la terminologie suivante :

$\mathsf{Parent} \equiv \exists \mathsf{aEnfant}.\top$
$\mathsf{ParentDeFamilleNombreuse} \equiv \mathsf{Parent} \sqcap \geq 3\ \mathsf{aEnfant}$

Nous voulons maintenant démontrer qu'un parent de quatre enfants ou plus est un parent de famille nombreuse, soit :

$$\geq 4\ \mathsf{aEnfant} \sqsubseteq \mathsf{ParentDeFamilleNombreuse}$$

Pour ce faire, on démarre le tableau avec le fait $(\geq 4\ \mathsf{aEnfant} \sqcap \neg \mathsf{ParentDeFamilleNombreuse})(a)$ que l'on normalise et réécrit de la manière suivante après élimination de la TBox :

$$(\geq 4\ \mathsf{aEnfant} \sqcap (\forall \mathsf{aEnfant}.\bot \sqcup \leq 2\ \mathsf{aEnfant}))(a)$$

Remarquez qu'ici nous avons appliqué la règle d'équivalence $\neg(\geq n\ R) \equiv\ \leq (n-1)\ R$. Voici l'état du tableau au moment d'appliquer la règle-$\leq$:

$$\begin{array}{lc}
(1) & (\geq 4\ \mathsf{aEnfant} \sqcap (\forall \mathsf{aEnfant}.\bot \sqcup \leq 2\ \mathsf{aEnfant}))(a) \\
(2) & (\geq 4\ \mathsf{aEnfant})(a) \\
(3) & (\forall \mathsf{aEnfant}.\bot \sqcup \leq 2\ \mathsf{aEnfant})(a) \\
(4) & \mathsf{aEnfant}(a,b) \\
(5) & \mathsf{aEnfant}(a,c) \\
(6) & \mathsf{aEnfant}(a,d) \\
(7) & \mathsf{aEnfant}(a,e) \\
(8) & b \neq c \\
(9) & b \neq d \\
(10) & b \neq e \\
(11) & c \neq d \\
(12) & c \neq e \\
(13) & d \neq e
\end{array}$$

$$\begin{array}{lc}
(14) & (\forall \mathsf{aEnfant}.\bot)(a) \\
(15) & \bot(b) \\
(16) & \square
\end{array}
\qquad
\begin{array}{lc}
(17) & (\leq 2\ \mathsf{aEnfant})(a)
\end{array}$$

Considérons maintenant les individus b, c et d. La règle-$\leq$ nous dit qu'il faut créer trois nouvelles branches, qui seront toutes rapidement fermées :

VI- Relations des LD avec d'autres logiques :

1- Les logiques de description et la logique modale

Il existe une relation étroite entre la logique de description ALC et la logique modale. En effet, les concepts de la logique ALC peuvent être considérés comme des variantes syntaxiques des formules modales du système K et inversement : les structures KRIPKE sont interprétées comme des interprétation en logique de description.

Ainsi, les noms des concepts sont considérées comme des variables propositionnelles et le noms de rôle sont interprétés comme des paramètres modaux. Cette correspondance peut être illustrée en utilisant les règles de réecritures suivantes :

$\mathcal{ALC}$-concept		Modal **K** formula
A	$\leftrightsquigarrow$	a, for concept name A and propositional variable a,
$C \sqcap D$	$\leftrightsquigarrow$	$C \wedge D$,
$C \sqcup D$	$\leftrightsquigarrow$	$C \vee D$,
$\neg C$	$\leftrightsquigarrow$	$\neg C$,
$\forall r.C$	$\leftrightsquigarrow$	$[r]C$,
$\exists r.C$	$\leftrightsquigarrow$	$\langle r \rangle C$.

2- Les logiques de description et la logique des défauts

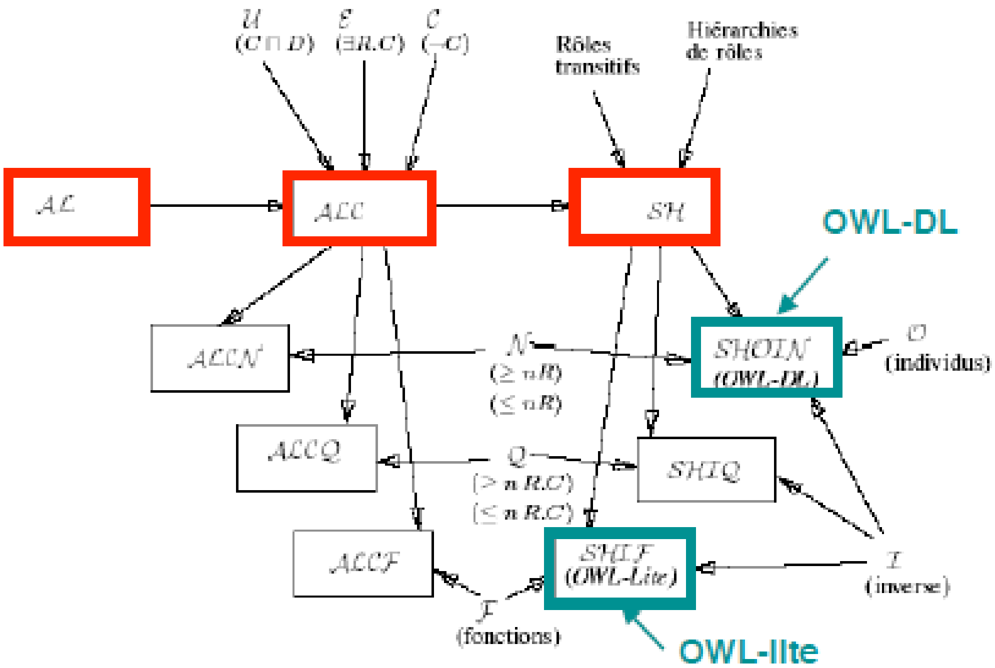
1. [J. Quantz &V. Royer 92] : étendre les DL en y incluant une forme limitée de raisonnement par défaut. Les défauts ne sont pas autorisés dans la définition même des concepts mais en tant que règles incidentes des concepts. Le but principal est la spécification d'une sémantique basée sur les sémantiques de préférence définies dans [Shoham 88]. Les défauts sont exprimés sous la forme de règles du type : c1~ c2. Cette règle signifie : si un objet est une instance du concept C1 alors c'est aussi une instance du concept C2 à moins d'un conflit avec une autre connaissance. (un conflit est caractérisé par l'appartenance à deux concept disjoints, i.e., intersection de leurs extensions vide). Si le conflit porte sur des propriétés strictes, l'information posant pb est rejetée. Si un objet est une instance par défaut de deux concepts disjoints, les auteurs supposent qu'il y a exception sur un des défauts et lancent un processus de résolution de conflits. Ce processus est basé sur la théorie des modèles préférentiels de Shoham.
2. [L. Padgham & al.93] s'inspirent des travaux réalisés dans le cadre de l'*héritage non monotone*. Ils définissent la notion de *subsomption défaut* en se basant sur *l'héritage sceptique*. Dans leur approche un concept possède une définition scindée en deux parties: une partie (appelée *core*) qui comprend les propriétés strictes suffisantes et nécessaires du concept et une partie (appelée *default*) qui comprend des propriétés strictes mais aussi des propriétés défaut.
3. 3. P. Coupey et C. Fouqueré qui ont permis l'introduction de deux nouveaux connecteurs (δ, ε) au sein même de la définition des concepts (dans la T-Box). De manière intuitive, le connecteur δ représente la notion de *défaut* comme par exemple la définition suivante : *Mammifère ≡ Animal ∩ δ Vivipare ∩ Vertébré* définit le concept *Mammifère* comme un Animal Vertébré généralement Vivipare.
   Malheureusement, en utilisant cette définition de Mammifère on peut par exemple inférer qu'un *Canard* (*Canard ≡ Animal ∩ Ovipare ∩ Vertébré ∩ Avec-bec ∩ Pied-Palmé ∩ Vol)* est un mammifère puisqu'il possède les propriétés Vertébré et Animal. En présence de connaissances par défaut la classification automatique semble impossible (def non nécessaire).
   Pour résoudre ce pb, les auteurs introduisent le connecteur ε qui représente une exception à un concept. Ils définissent un point de vue définitionnel pour les défauts et expriment la propriété suivante :
   Un objet est une instance d'un concept C ssi il satisfait les propriétés strictes de C ou est explicitement ''exceptionnel '' par rapport aux propriétés défauts de C.
   Avec cette propriété on ne peut plus inférer qu'un canard est un Mammifère puisqu'il n'est ni vivipare ni exceptionnel par rapport au fait d'être vivipare par contre le concept *Ornithorynque* (*Ornithorynque ≡ Animal ∩ Vertébré ∩ Ovipare ∩ Avec-bec ∩ Vivipare*$^{\varepsilon}$) sera classé sous le concept *Mammifère* vérifie les propriété strictes de *Mammifère* et est exceptionnel / au fait d'être *Vivipare.*

VII- L'hypothèse du monde ouvert dans les logiques de description :

Dans les logiques de description, les connaissances sont interprétées selon l'hypothèse du monde ouvert ce qui signifie que l'absence d'information représente l'ignorance plutôt qu'une information négative. De ce fait, il n'y a pas de limitation aux connaissances énoncées. Ainsi, le processus de raisonnement sera complexe car l'hypothèse du monde ouvert donne lieu à plusieurs interprétations possibles.
Contrairement à l'hypothèse du monde ouvert, l'hypothèse du monde fermé ou mode clos induit la prise en compte uniquement des connaissances énoncées explicitement :
Exemple :
Soit l'assertion de rôle suivante d'une ABOX :
a-enfant(Aicha, Mohamed)
Selon, l'hypothèse du monde fermé, cette assertion signifie que Aicha a un seul enfant (Mohamed).
Par contre, dans le monde ouvert, cette assertion signifie uniquement que Mohamed est le fils de Aicha et que rien n'exclut que cette dernière possède d'autres enfants. Pour spécifier cela, il faudra rajouter l'assertion de concept suivante : (≤1 a-enfant)(Aicha).

VIII- Complexité et Expressivité :

La familles des logiques de description englobe une série de langages basés sur des constructeurs offrant ainsi une expressivité variante. Chaque ensemble de constructeur induit une logique précise.
Le tableau suivant donne un aperçu sur quelques logiques de description :



L'enrichissement de la logique de base AL par des constructeurs induit bien évidemment des algorithmes de raisonnement de complexité variante.

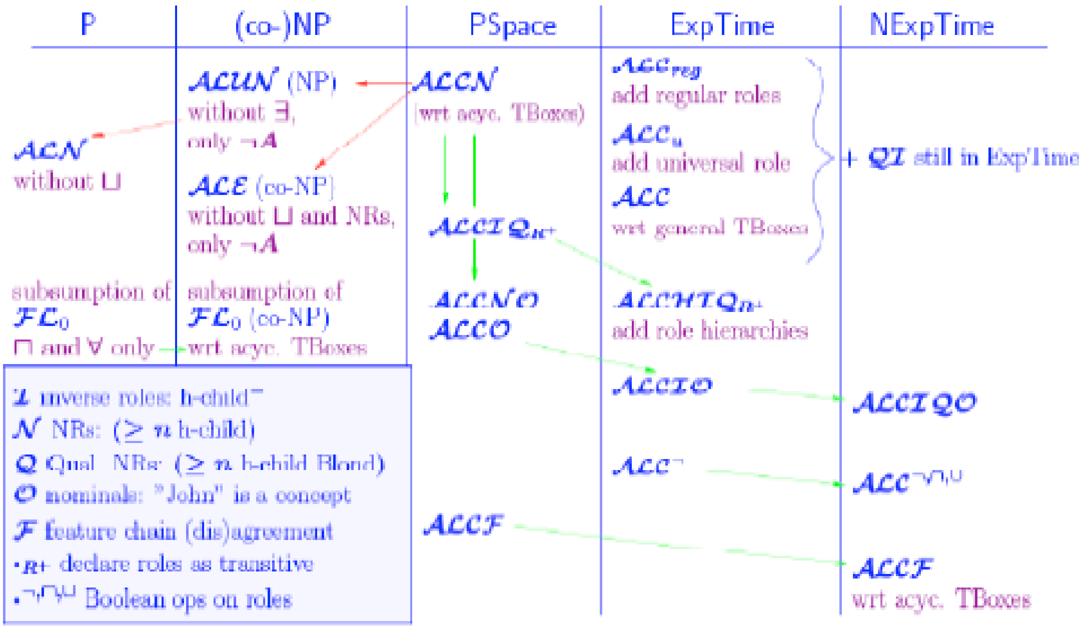
Le tableau suivant dresse une comparaison des complexités des mécanismes de raisonnement selon l'expressivité des logiques de description :

Complexité	LD
P	$\mathcal{AL}$, $\mathcal{ALN}$
NP	$\mathcal{AL}_\mathcal{E}$
PSpace	$\mathcal{ALC}$, $\mathcal{AL}_\mathcal{E}\mathcal{N}$
ExpTime	$\mathcal{SHIQ}$, $\mathcal{SHOQ}$,
NExpTime	...

- **P**: classe des problèmes de décision (en entrée un énoncé de problème et en sortie une réponse oui ou non) demandant un **temps polynomial** par rapport à la taille du problème (n), pour obtenir une solution avec une machine de Turing **déterministe**
- **NP**: classe des problèmes nécessitant un **temps polynomial** pour trouver une solution avec une machine de Turing **non déterministe**
- **PSpace** : classe des problèmes demandant une quantité **de mémoire polynomiale** pour une résolution avec une machine de Turing **déterministe** ou **non déterministe**
- **ExpTime** : classe des problèmes solvables par une machine de Turing **déterministe** en un temps $\Theta(2^{p(n)})$ où $p(n)$ est une fonction polynomiale de n
- **NExpTime** : la classe des problèmes solvables par une machine de Turing **non-déterministe** en un temps $\Theta(2^{p(n)})$ où $p(n)$ est une fonction polynomiale de n

On a P ⊆ NP ⊆ PSpace ⊆ ExpTime ⊆ NExpTime [Padadimitriou, 1994]

Le tableau suivant illustre avec plus de détails la complexité des algorithmes de raisonnement en fonction de l'expressivité des principales logiques de description (http://www.cs.man.ac.uk/~ezolin/dl/).



Le choix d'une logique de description est guidé par le domaine d'application où la prise en compte d'un compromis entre l'expressivité et la compléxité de raisonneur s'impose.

IX- Implémentation des systèmes :

Plusieurs raisonneurs ont été développés pour les familles des logiques de description offrant les mécanismes de raisonnement sur des bases de connaissances exprimées en logique de description (TBOX et ABOX).

Une liste assez exhaustive des raisonneurs est donnée sur le site web officiel des logiques de description : http://dl.kr.org/.

Les raisonneurs libres les plus utilisés sont :

- RACER (Renamed Abox and Concept Expression Reasoner):
  https://github.com/ha-mo-we/Racer
  http://www.racer-systems.com/

- FaCT (Fast Classification of Terminologies) :
  http://www.cs.man.ac.uk/~horrocks/FaCT

- FACT++ : https://code.google.com/archive/p/factplusplus/

- Hermit : http://hermit-reasoner.com/

- Pellet : http://clarkparsia.com/pellet/

Le tableau suivant dresse les caractéristiques des raisonneurs les plus connus :

Moteur	*Racer*	*FaCT*	*Pellet*	*FaCT++*
LD	$\mathcal{SHIQ}(\mathcal{D})^-$	$\mathcal{SHIQ}$, $\mathcal{SHF}$	$\mathcal{SHIN}(\mathcal{D})$, $\mathcal{SHON}(\mathcal{D})$	$\mathcal{SHIF}(\mathcal{D})$
Implantation	*C++*	*Common Lisp*	*Java*	*C++*
Inférence	*TBox/ABox*	*TBox*	*TBox/ABox*	*TBox*
API Java	oui	oui	natif	oui
Mise-à-échelle	bonne	bonne	bonne	bonne
OWL	*OWL-DL*$\sim^{\dagger}$	*OWL-DL*$\sim^{\dagger}$	*OWL-DL*$\sim^{\dagger}$	*OWL-LITE*
Décidabilité	oui (*OWL-LITE*)	oui	oui (*OWL-LITE*)	oui
DIG	oui	oui	non	?

Moteur	*Surnia*	*Hoolet*	*F-OWL*	
LD	logique prédicats	logique prédicats	$\mathcal{SHIQ}(\mathcal{D})$ et *RDF*	
Implantation	*Python*	*Java*	*Java*	
Inférence	*TBox/ABox*	*TBox/ABox*	*TBox/ABox*	
API Java	?	oui	oui	
Mise-à-échelle	médiocre	médiocre	médiocre	
OWL	*OWL-FULL*$\sim^{\dagger}$	*OWL-DL*$\sim^{\dagger}$	*OWL-FULL*$\sim^{\dagger}$	
Décidabilité	non	non	non	
DIG	non	non	non	

X- DOMAINES D'APPLICATION des LD :

1. Web-based information systems. Malgré le succès du www, il reste néanmoins limité à cause de ses liens avec des langages tel que HTML qui se focalisent sur la présentation (formater des textes) plutôt que sur le contenu. Des langages tels que XML ont tenté de capturer un peu plus le sens mais ils sont très insuffisants et ne peuvent supporter des applications intelligentes. Or il est urgent que ces applications se développent. En effet, même si nous sommes très contents de taper quelques mots clés sur Google pour obtenir des informations sur un sujet qui nous intéressent , très souvent nous sommes frustrés par le manque d'informations (**le silence**) ou agacés par le flot d'information inutiles (**les bruits**). L'objectif est d'introduire plus de sémantique pour capturer les informations du web. Pour cela, des langages basés sur des DL ont été développés, OWL (OWL DL) par exemple (web ontology language 2004) est le world wide web consortium (W3C) langage recommandé pour le web semantic. Il exploite la force des DL en particulier sa sémantique et ses techniques de raisonnement.
2. Ingénierie des logiciels : l'idée est d'utiliser une DL pour implémenter un système qui supporte le développeur de logiciels en l'aidant à retrouver des informations sur un gros système de logiciel (LaSSIE system Devanbu et al. 1991.
3. Langage Naturel. Le traitement du langage naturel est l'application à l'origine de cette logique : le sens des mot ou lexique, représentation du sens d'une phrase, le contexte, la désambigüisation des différentes lectures de phrases etc…
4. Gestion des bases de données : le liens avec les BD est très fort. On a souvent besoin de faire coexister des SGBD et des KBS. Le premier pour la persistance des données et la gestion d'un grand nombre de ses données le second pour la gestion intentionnelle des connaissances.

XI- Exemple d'interprétation des images IRM du cerveau en exploitant les logiques de description [Bloch 2017]:

La base de connaissances est composée d'une Rbox, d'une TBox et d'une Abox. La Tbox décrit les connaissances anatomiques et les pathologies. La Abox représente les résultats sur les structures extraites d'une image IRM par des méthodes d'analyse d'images.

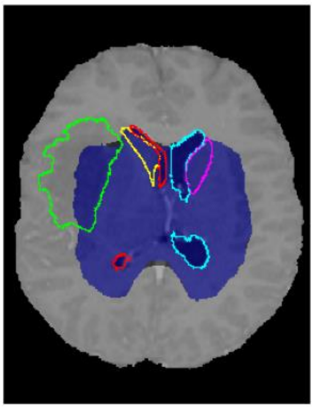
$$
\begin{aligned}
RBox = \{rightOf &\equiv leftOf^{-} \\
above &\equiv below^{-} \\
closeTo &\equiv closeTo^{-} \\
farFrom &\equiv farFrom^{-} \\
isPartOf \circ isPartOf &\sqsubseteq isPartOf \\
hasPart \circ hasPart &\sqsubseteq hasPart \\
isPartOf &\equiv hasPart^{-}\}
\end{aligned}
$$

$$
\begin{aligned}
TBox = \{Hemisphere &\sqsubseteq \exists isPartOf.Brain \\
BrainStructure &\sqsubseteq \exists isPartOf.Brain \\
BrainDisease &\sqsubseteq \exists isPartOf.Brain \sqcap \\
&\quad \neg BrainStructure \\
BrainTumor &\sqsubseteq BrainDisease \\
LVl &\sqsubseteq BrainStructure \\
LVr &\sqsubseteq BrainStructure \\
CNl &\sqsubseteq BrainStructure \\
CNr &\sqsubseteq BrainStructure \\
PeripheralRegion &\sqsubseteq \exists isPartOf.Brain \\
SubCorticalRegion &\sqsubseteq \exists isPartOf.Brain
\end{aligned}
$$

$$
\begin{aligned}
SmallDeformingTumor \sqsubseteq & BrainTumor \sqcap \\
& \exists hasEnhancement.NonEnhanced \\
\boldsymbol{PeripheralSmallDeformingTumor} \sqsubseteq & \boldsymbol{SmallDeformingTumor} \sqcap \\
& \boldsymbol{\exists isPartOf.PeripheralRegion} \sqcap \\
& \boldsymbol{\exists farFrom.(LVl \sqcup LVr)} \\
\boldsymbol{SubCorticalSmallDeformingTumor} \sqsubseteq & \boldsymbol{SmallDeformingTumor} \sqcap \\
& \boldsymbol{\exists isPartOf.SubCorticalRegion} \sqcap \\
& \boldsymbol{\exists rightOf.CNr} \}
\end{aligned}
$$

$$
\begin{aligned}
ABox = \{ a &: BrainTumor \\
b &: NonEnhanced \\
\langle a, b \rangle &: hasEnhancement \\
c &: CNr \\
d &: CNl \\
k &: LVr \\
m &: LVl \\
p &: PeripheralRegion \\
s &: SubCorticalRegion \}
\end{aligned}
$$

L'objectif est d'exprimer une description de haut niveau des tumeurs dans le langage. La figure suivante illustre un résultat de segmentation sur une coupe. Les instances de la Abox a, b, c, d, k, m, p et s désignent des partie de l'image.



. Segmentation de l'image IRM (sur une coupe). Les régions et contours en couleurs montrent les structures segmentées. La région bleue représente la région sous-corticale et la région grise la région périphérique. Les contours jaunes et violets sont ceux des noyaux caudés, les contours rouges et cyan ceux des ventricules latéraux, et les contours en vert sont ceux de la tumeur

La méthode des tableaux est appliquée afin d'exhiber les hypothèses cohérentes. La figure suivante illustre l'application de l'algorithme.

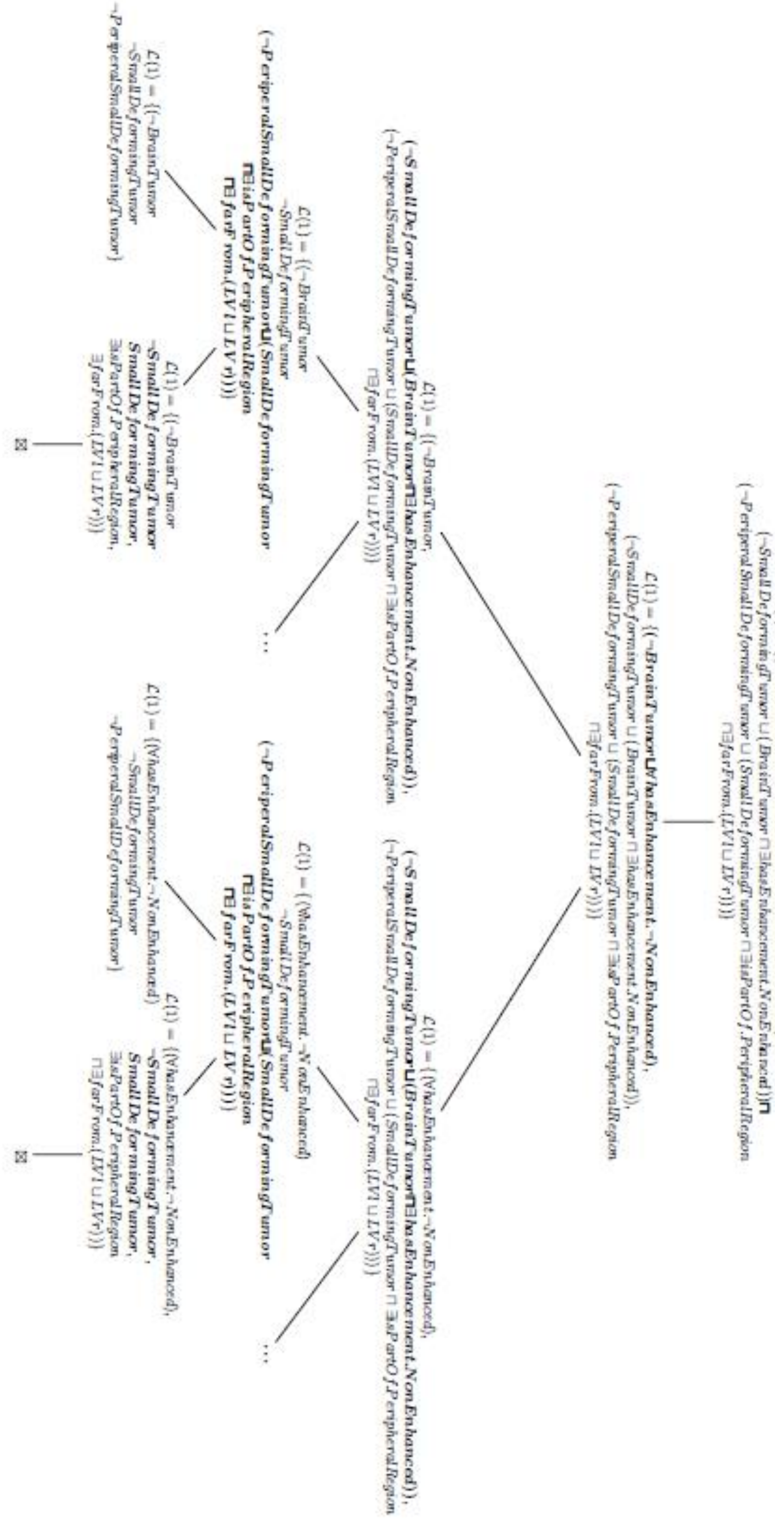
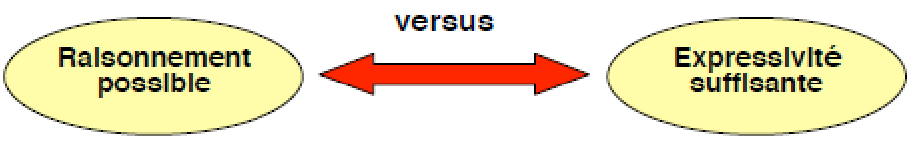


Figure 12. Une partie du tableau sémantique construit pour expliquer l'image IRM du cerveau pathologique

XII- Conclusion :

Le choix d'une DL repose sur un compromis entre l'expressivité des DL et la complexité des mécanismes de raisonnement comme c'est souligné par le tableau suivant [Baader 2000]:

Décidabilité / complexité du raisonnement	Application de concepts pertinents doit être définissable
Nécessite un langage de description restreint	Des domaines d'application nécesite des DL très expressives
Les systèmes et les résultats de complexité disponibles pour différentes combinaisons de constructeurs	Dispose-t-on en pratique d'algorithmes efficaces pour des DL très expressives ?



XIII- Bibliographie :

Remarque : Ce document est inspiré principalement du cours de Bernard Espinasse – Université 2009.

The Description Logic Handbook : Theory, Implementation and Applications, edited by F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, P.F. Patel-Schneider,Cambridge University Press (ISBN-13: 9780521781763 I ISBN-10: 0521781760)

H. J. Levesque and R. J. Brachman. **Expressiveness and tractability in knowledge representation and reasoning**. Computational Intelligence journal 3, 78-93 (1987).

Donini, F., Lenzerini, M., Nardi, D., Schaerf, A., **Reasoning in Description Logics**, in: Principles of Knowledge Representation and Reasoning, edited by G. Brewka; Studies in Logic, Language and Information, CLSI Publications, pp 193-238, 1996.

Baader and U. Sattler. **An Overview of Tableau Algorithms for Description Logics**. Studia Logica, 69:5-40, 2001

D. Calvanese, G. De Giacomo, M. Lenzerini, and D. Nardi**, Reasoning in expressive description logics**, in: A. Robinson and A. Voronkov, editors, Handbook of Automated Reasoning. Elsevier Science Publishers (North-Holland), Amsterdam, 2001, pages 1581-1634.

Hollunder, B., Nutt, W., **Subsumption Algorithms for Concept Languages**, DFKI report, RR-90-04, Saarbruecken Germany, 1990; extended version of a previously published paper in proc. ECAI'90, pp 348-353.

Yifan Yang, Jamal Atif and Isabaelle Bloch : Raisonnement abductif en logique de description exploitant les domaines concrets spatiaux pour l'interprétation d'images, 2017.

CONCLUSION GENERALE

Les différents modèles étudiés permettent de :

- Représenter les connaissances du monde
- Raisonner sur des connaissances

Ces modèles sont basés sur deux approches :

a- Une approche logique : Plusieurs modèles logiques ont vu le jour. Ces modèles sont considérés comme des extensions des logiques classiques (la logique propositionnelle et la logique des prédicats) :

- Les logiques modales :
    - Connaissances Modales
        - Connaissances épistémiques
        - Connaissances déontiques
        - Connaissances doxastiques
    - Connaissances temporelles
    - Connaissances spatiales
- La logique des défauts
- Les logiques de description

b- Une approche graphique ad hoc basée sur des structures de graphes

- Les réseaux sémantiques
- Les frames

MODELES LOGIQUES	MODELES GRAPHIQUES
Expressivité variante	La structure du réseau (nœuds et arcs) permet une bonne représentation des relations entre concepts
Sémantique puissante	
Expressivité variante	
Efficacité du processus de raisonnement (règles d'inférence et axiomes)	Processus de raisonnement ad hoc
Difficulté pour l'interprétation des connaissances par des experts humains	Complexité liée à la structure du graphe
Eloignement du domaine d'application	Manque de sémantique précise