

Ontologies et web sémantique: Introduction aux logiques de description

Andreas Herzig

CNRS, IRIT, Univ. Toulouse

<http://www.irit.fr/~Andreas.Herzig>

(en partie repris du cours de Nicola Olivetti,
Univ. Aix-Marseille, <http://www.lsis.org/olivetti>)

Univ. Toulouse, M2 informatique
parcours *Données et connaissances*
2020-2021

Introduction : resources

- livres :

- Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi and Peter F. Patel-Schneider (eds.)
The Description Logic Handbook: Theory, Implementation, and Applications
Cambridge University Press, 2003 (réédité en 2007)
- Franz Baader, Ian Horrocks, Carsten Lutz, and Ulrike Sattler
An Introduction to Description Logic
Cambridge University Press, 2017
<http://dltextbook.org>

- le site web : <http://dl.kr.org>

- ⇒ DL workshop
- ⇒ conférences afférantes
- ⇒ entrepôt d'ontologies

Introduction : mots-clés et sigles

- DL = *description logic* = logique de description

(avant : logiques terminologiques)

‘description’ ⇒ **représenter** les connaissances

but : exprimer plus qu’en logique propositionnelle

‘logique’ ⇒ **raisonner** à partir de ces connaissances

but : avoir de meilleures propriétés calculatoires

que la logique des prédicats

- KB = *knowledge base* = base de connaissances

$$KB = TBox \cup ABox$$

- ABox = *assertion box*

= connaissances contingentes

- TBox = *terminological box*

= ontologie

= connaissances non contingentes

= axiomes

≈ contraintes d’intégrité

Introduction : mots-clés et sigles

- DL = *description logic* = logique de description

(avant : logiques terminologiques)

‘description’ ⇒ **représenter** les connaissances

but : exprimer plus qu’en logique propositionnelle

‘logique’ ⇒ **raisonner** à partir de ces connaissances

but : avoir de meilleures propriétés calculatoires

que la logique des prédicats

- KB = *knowledge base* = base de connaissances

$$KB = TBox \cup ABox$$

- ABox = *assertion box*

= **connaissances contingentes**

- TBox = *terminological box*

= ontologie

= **connaissances non contingentes**

= **axiomes**

≈ contraintes d’intégrité

Introduction : la ABox

ABox = *assertion box*

- vraies à un instant donné dans le temps (qui reste implicite)
 - ① prédicats unaires : propriétés d'individus

Feminine(Anne)

Masculin(Charles)

- ② prédicats binaires : relations entre individus

mereDe(Anne, Charles)

pereDe(Bob, Charles)

pereDe(Charles, Dan)

⇒ la ABox utilise les axiomes de la TBox ...

Introduction : la ABox

ABox = *assertion box*

- vraies à un instant donné dans le temps (qui reste implicite)
 - ① prédicats unaires : propriétés d'individus

Feminine(Anne)

Masculin(Charles)

- ② prédicats binaires : relations entre individus

mereDe(Anne, Charles)

pereDe(Bob, Charles)

pereDe(Charles, Dan)

⇒ la ABox utilise les axiomes de la TBox ...

Introduction : la TBox

TBox = *terminological box*

- axiomatise les propriétés et les relations
 - terminologie DL: concepts et rôles
 - principalement : inclusion de concepts

Mere \sqsubseteq Parent

Mere \sqsubseteq Parent \sqcap Feminine

Mere \sqsubseteq \exists mereDe.T

\exists mereDe.T \sqsubseteq Mere

\exists mereDe.(\exists mereDe.T) \sqsubseteq GrandMere

- aussi: équivalence de concepts

Mere \equiv \exists mereDe.T

- et parfois aussi: inclusion de rôles

mereDe \sqsubseteq parentDe

- vrais pour tout instant temporel (qui reste implicite)

Introduction : la TBox

TBox = *terminological box*

- axiomatise les propriétés et les relations
 - terminologie DL: concepts et rôles
 - principalement : inclusion de concepts

Mere \sqsubseteq Parent

Mere \sqsubseteq Parent \sqcap Feminine

Mere \sqsubseteq \exists mereDe.T

\exists mereDe.T \sqsubseteq Mere

\exists mereDe.(\exists mereDe.T) \sqsubseteq GrandMere

- aussi: équivalence de concepts

Mere \equiv \exists mereDe.T

- et parfois aussi: inclusion de rôles

mereDe \sqsubseteq parentDe

- vrais pour tout instant temporel (qui reste implicite)

Introduction : les concepts

un concept dénote un ensemble d'individus

- concept atomique : Parent, Masculin, Feminine, Mere, ...
 - 1 concept atomique primitif : Parent, Masculin, ...
 - 2 concept atomique non-primitif : Mere, GrandMere, ...
⇒ définis dans la TBox en termes d'autres concepts :

$$\text{Femme} \equiv \text{Personne} \sqcap \text{Feminine}$$

$$\text{Mere} \equiv \exists \text{mereDe}.\top$$

- concept complexe : Personne \sqcap Feminine, $\exists \text{mereDe}.\top$, $\exists \text{mereDe}.\langle \exists \text{mereDe}.\top \rangle$, ...
 - constructeurs de concepts complexes :
 - 1 opérateurs booléens : $\neg C$, $C \sqcap C$, $C \sqcup C$
 - 2 opérateurs combinant concepts et rôles : $\exists R.C$, $\forall R.C$

Introduction : les rôles

un rôle dénote un ensemble de couples d'individus

- rôle atomique :
 - `mereDe`, `partieDe`, `mange`, ...
- rôle complexe : `pereDe` \sqcap `grandPereDe`
- constructeurs de rôles complexes:
 - ① opérateurs booléens : $\neg R$, $R \sqcap R$, $R \sqcup R$
 - ② opérateurs de l'algèbre des relations :
 - $R \circ R$ (composition)
 - R^* (itération, *Kleene star*)
 - R^{-1} (converse)
 - \bar{R} (complément)

Introduction : l'aspect raisonnement

raisonner = 'extraire de la KB plus que ce qu'il y a écrit'

- exemple : $KB = \mathcal{T} \cup \mathcal{A}$, où

$$\mathcal{A} = \{\text{Pere}(\text{Bob})\}$$

$$\mathcal{T} = \{\text{Pere} \sqsubseteq \text{Personne}\}$$

- KB ne contient pas $\text{Personne}(\text{Bob})$
- KB a comme **conséquence logique** $\text{Personne}(\text{Bob})$!
- services de raisonnement :
 - 1 inférence de propriétés d'individus
 - 2 inférence de relations entre individus
 - 3 subsomption de concepts
 - 4 classification: calculer la hierarchie de subsomption entre concepts
 - 5 satisfaisabilité
 - 6 non-vacuité de concepts (pas de A tel que $KB \models A \equiv \perp$)
 - 7 non-redondance de concepts (pas de A, A' t.q. $KB \models A \equiv A'$)

Introduction : l'aspect raisonnement

raisonner = 'extraire de la KB plus que ce qu'il y a écrit'

- exemple : $KB = \mathcal{T} \cup \mathcal{A}$, où

$$\mathcal{A} = \{\text{Pere}(\text{Bob})\}$$

$$\mathcal{T} = \{\text{Pere} \sqsubseteq \text{Personne}\}$$

- KB ne contient pas $\text{Personne}(\text{Bob})$
- KB a comme **conséquence logique** $\text{Personne}(\text{Bob})$!
- services de raisonnement :
 - 1 inférence de propriétés d'individus
 - 2 inférence de relations entre individus
 - 3 subsomption de concepts
 - 4 classification: calculer la hierarchie de subsomption entre concepts
 - 5 satisfaisabilité
 - 6 non-vacuité de concepts (pas de A tel que $KB \models A \equiv \perp$)
 - 7 non-redondance de concepts (pas de A, A' t.q. $KB \models A \equiv A'$)

Introduction : avantages des DL

- ① sémantiques claires et bien définies
 - ≠ langage naturel
- ② tâches de raisonnement **décidables**
 - ≠ logique du premier ordre FOL
 - plus difficiles que pour la logique propositionnelle
 - raison : les DL sont plus expressifs
- ③ applications multiples
 - web sémantique
 - bases de données (modèles entité-relation exprimables en DL)
 - génie du logiciel (diagrammes UML exprimables en DL)
- ④ flexibles
 - hiérarchie des DL
 - chaque DL est déterminée par des restrictions du langage
 - expressivité vs. complexité

Introduction : complexité vs. expressivité

- restrictions de l'expressivité :
 - pas de négation ' \neg ' ; pas de disjonction ' \sqcup ' ; ...
 - pas d'inclusion de rôles ' \sqsubseteq ' ; pas de rôles complexes ; ...
 - dans $\exists R.C$ il faut que $C = \top$; ...
- expressivité du langage \Rightarrow complexité du raisonnement
 - PTIME = temps polynomial
 - NPTIME = temps polynomial non-déterministe
 - PSPACE = espace polynomial (PSPACE = NPSPACE)
 - EXPTIME = temps exponentiel
 - NEXPTIME = temps exponentiel non-déterministe
 - EXPSPACE = espace exponentiel

\Rightarrow 'navigateur de complexité' sur <http://dl.kr.org>

- 1 Le langage ALC
- 2 Les langages ALCN et ALCQ
- 3 Relation avec la logique du premier ordre
- 4 Structure et propriétés des TBox
- 5 Structure et propriétés des ABox
- 6 Tâches de raisonnement
- 7 Le langage SROIQ
- 8 Mécanismes de raisonnement

Outline

- 1 Le langage ALC
- 2 Les langages ALCN et ALCQ
- 3 Relation avec la logique du premier ordre
- 4 Structure et propriétés des TBox
- 5 Structure et propriétés des ABox
- 6 Tâches de raisonnement
- 7 Le langage SROIQ
- 8 Mécanismes de raisonnement

Syntaxe

- ALC = **A**ttributive **C**oncept **L**anguage with **C**omplements
[Schmidt-Schauß & Smolka, 1991]

N.B. : plutôt que 'langage', devrait s'appeler *logique*
(logique = langage + sémantique)

- convention :
 - A, A_j, \dots : concepts atomiques
 - dans les exemples : commence par majuscule (Parent, ...)
 - C, D, C_j, \dots : concepts arbitraires
 - R, R_j, \dots : rôles atomiques
 - dans les exemples : commence par minuscule (**parentDe**, ...)
 - *pas de rôles complexes*

Syntaxe : concepts complexes

- constructeurs de concepts :

\top	concept universel	“tout”
\perp	concept vide	“rien”
$\neg C$	complément (ou : négation)	“non C”
$C \sqcap D$	intersection (ou : conjonction)	“C et D”
$C \sqcup D$	union (ou : disjonction)	“C ou D”
$\forall R.C$	quantification universelle restreinte	“tous les R-successeurs sont dans C”
$\exists R.C$	quantification existentielle restreinte	“il existe un R-successeur qui est dans C”

- grammaire des concepts ('BNF') :

$$C ::= N_C \mid \top \mid \perp \mid \neg C \mid (C \sqcap C) \mid (C \sqcup C) \mid (\forall N_R.C) \mid (\exists N_R.C)$$

où :

- C : symbole non-terminal de la grammaire
- $N_C \in NomsConcepts$, $N_R \in NomsRoles$ ('symboles terminaux')
- $\top, \perp, \neg, \sqcap, \sqcup$: connecteurs logiques

Syntaxe : concepts complexes

- constructeurs de concepts :

\top	concept universel	“tout”
\perp	concept vide	“rien”
$\neg C$	complément (ou : négation)	“non C”
$C \sqcap D$	intersection (ou : conjonction)	“C et D”
$C \sqcup D$	union (ou : disjonction)	“C ou D”
$\forall R.C$	quantification universelle restreinte	“tous les R-successeurs sont dans C”
$\exists R.C$	quantification existentielle restreinte	“il existe un R-successeur qui est dans C”

- grammaire des concepts ('BNF') :

$$C ::= N_C \mid \top \mid \perp \mid \neg C \mid (C \sqcap C) \mid (C \sqcup C) \mid (\forall N_R.C) \mid (\exists N_R.C)$$

où :

- C: symbole non-terminal de la grammaire
- $N_C \in \text{NomsConcepts}$, $N_R \in \text{NomsRoles}$ ('symboles terminaux')
- $\top, \perp, \neg, \sqcap, \sqcup$: connecteurs logiques

Syntaxe : exemples

- concepts atomiques :
Personne, Masculin, Feminine, Riche, ...
- rôles atomiques :
parentDe, mereDe, pereDe
- concepts complexes :
 - 1 $\text{Personne} \sqcap \text{Feminine}$
 - 2 $\text{Personne} \sqcap \neg \text{Feminine}$
 - 3 $\text{Personne} \sqcap \exists \text{parentDe}.\top$
 - 4 $\text{Personne} \sqcap \forall \text{parentDe}.\perp$
 - 5 $\text{Personne} \sqcap \exists \text{parentDe}.\top \sqcap \forall \text{parentDe}.\text{Feminine}$
 - 6 $\text{Personne} \sqcap (\text{Riche} \sqcup \exists \text{parentDe}.\text{Riche})$
 - 7 $\text{Personne} \sqcap \exists \text{parentDe}.\exists \text{parentDe}.\top$

Syntaxe : exercices

- étant donné les ensembles

$NomsConcepts = \{Personne, Masculin, Feminine, Riche\}$

$NomsRoles = \{parentDe, sœurDe, frereDe\},$

définir les concepts complexes suivants :

- être une femme
- être une mère
- être une mère qui n'a que des garçons
- être un oncle
- être un grand-oncle
- être un quelqu'un qui a un neveu riche
- être quelqu'un qui a un oncle riche ??
⇒ pas de converse dans ALC
- être quelqu'un qui n'a qu'un seul enfant ??
⇒ pas de restriction de cardinalité dans ALC

Sémantique

- **interprétation**: donne du sens aux concepts atomiques et aux rôles atomiques (dans un contexte particulier)
- interprétation $\mathcal{I} = (\Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}})$ telle que :
 - $\Delta^{\mathcal{I}}$ est un ensemble non-vide (domaine d'individus)
 - $(\cdot)^{\mathcal{I}} : \text{NomsConcepts} \rightarrow 2^{\Delta^{\mathcal{I}}}$ (interprétation des concepts)
 - $(\cdot)^{\mathcal{I}} : \text{NomsRoles} \rightarrow 2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}}$ (interprétation des rôles)

logique du premier ordre : il y faut en plus interpréter les prédicats 3-aires, 4-aires,...

- une interprétation \mathcal{I} :
 - donne du sens aux concepts et rôles atomiques

$$(\mathbf{A})^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$$

$$(\mathbf{R})^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$$

- reste à donner du sens aux concepts complexes...

Sémantique (suite)

- interprétation de concepts complexes :

$$\top^I = \Delta^I$$

$$\perp^I = \emptyset$$

$$(\neg C)^I = \Delta^I \setminus C^I$$

$$(C \sqcap D)^I = C^I \cap D^I$$

$$(C \sqcup D)^I = C^I \cup D^I$$

$$(\forall R.C)^I = \{a \in \Delta^I : \text{pour tout } b \in \Delta^I, \text{ si } (a, b) \in R^I \text{ alors } b \in C^I\}$$

$$(\exists R.C)^I = \{a \in \Delta^I : \text{il existe } b \in \Delta^I \text{ tel que } (a, b) \in R^I \text{ et } b \in C^I\}$$

Sémantique : quelques équivalences

équivalences sémantiques :

$$(\neg\neg C)^I = C^I$$

$$(\neg(C \sqcap D))^I = (\neg C \sqcup \neg D)^I$$

$$(\neg(C \sqcup D))^I = (\neg C \sqcap \neg D)^I$$

$$(\neg\forall R.C)^I = (\exists R.\neg C)^I$$

$$(\neg\exists R.C)^I = (\forall R.\neg C)^I$$

Proposition (forme normale négative)

Tout concept C peut être transformé en un concept $nnf(C)$ tel que

- ① dans $nnf(C)$, la négation apparaît seulement devant des concepts atomiques,
- ② pour toute interprétation I , $(nnf(C))^I = C^I$.

● exemple : $\neg\forall\text{parentDe}.\perp$

(... il manque une équivalence)

Sémantique : quelques équivalences

équivalences sémantiques :

$$(\neg\neg C)^I = C^I$$

$$(\neg(C \sqcap D))^I = (\neg C \sqcup \neg D)^I$$

$$(\neg(C \sqcup D))^I = (\neg C \sqcap \neg D)^I$$

$$(\neg\forall R.C)^I = (\exists R.\neg C)^I$$

$$(\neg\exists R.C)^I = (\forall R.\neg C)^I$$

Proposition (forme normale négative)

Tout concept C peut être transformé en un concept $nnf(C)$ tel que

- ① dans $nnf(C)$, la négation apparaît seulement devant des concepts atomiques,
- ② pour toute interprétation \mathcal{I} , $(nnf(C))^I = C^I$.

- exemple : $\neg\forall\text{parentDe}.\perp$

(... il manque une équivalence)

Sémantique : quelques équivalences

équivalences sémantiques :

$$(\neg\neg C)^{\mathcal{I}} = C^{\mathcal{I}}$$

$$(\neg(C \sqcap D))^{\mathcal{I}} = (\neg C \sqcup \neg D)^{\mathcal{I}}$$

$$(\neg(C \sqcup D))^{\mathcal{I}} = (\neg C \sqcap \neg D)^{\mathcal{I}}$$

$$(\neg\forall R.C)^{\mathcal{I}} = (\exists R.\neg C)^{\mathcal{I}}$$

$$(\neg\exists R.C)^{\mathcal{I}} = (\forall R.\neg C)^{\mathcal{I}}$$

Proposition (forme normale négative)

Tout concept C peut être transformé en un concept $\text{nnf}(C)$ tel que

- ① dans $\text{nnf}(C)$, la négation apparaît seulement devant des concepts atomiques,
- ② pour toute interprétation \mathcal{I} , $(\text{nnf}(C))^{\mathcal{I}} = C^{\mathcal{I}}$.

● exemple : $\neg\forall\text{parentDe}.\perp$

(... il manque une équivalence)

Outline

- 1 Le langage ALC
- 2 Les langages ALCN et ALCQ**
- 3 Relation avec la logique du premier ordre
- 4 Structure et propriétés des TBox
- 5 Structure et propriétés des ABox
- 6 Tâches de raisonnement
- 7 Le langage SROIQ
- 8 Mécanismes de raisonnement

Le langage ALCN : ALC + restrictions de cardinalité

- ALC plus un nouveau constructeurs de concept : restriction de cardinalité (*Number restriction*)
- grammaire des concepts ('BNF') :

$$C ::= N_C \mid \top \mid \perp \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \forall N_R.C \mid \exists N_R.C \mid \leq n N_R \mid \geq n N_R$$

où :

- $N_C \in \text{NomsConcepts}$
 - $N_R \in \text{NomsRoles}$
 - $n \geq 0$ est un entier naturel
- lecture :
 - $\leq n R$ = 'il y a au plus n R-successeurs'
 - $\geq n R$ = 'il y a au moins n R-successeurs'
 - exercice :
 - définir le concept $=n R$ ('il y a exactement n R-successeurs')

Le langage ALCN : ALC + restrictions de cardinalité

- ALC plus un nouveau constructeurs de concept : restriction de cardinalité (*Number restriction*)
- grammaire des concepts ('BNF') :

$$C ::= N_C \mid \top \mid \perp \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \forall N_R.C \mid \exists N_R.C \mid \leq n N_R \mid \geq n N_R$$

où :

- $N_C \in \text{NomsConcepts}$
 - $N_R \in \text{NomsRoles}$
 - $n \geq 0$ est un entier naturel
- lecture :
 - $\leq n R$ = 'il y a au plus n R-successeurs'
 - $\geq n R$ = 'il y a au moins n R-successeurs'
 - exercice :
 - définir le concept $= n R$ ('il y a exactement n R-successeurs')

ALCN : sémantique

- les mêmes interprétations que dans ALC :

- $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ telle que

- $\Delta^{\mathcal{I}}$ est un ensemble non-vidé (domaine)
- $(\cdot)^{\mathcal{I}} : \text{NomsConcepts} \rightarrow 2^{\Delta^{\mathcal{I}}}$ (interprétation des concepts)
- $(\cdot)^{\mathcal{I}} : \text{NomsRoles} \rightarrow 2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}}$ (interprétation des rôles)

- interprétation des concepts complexes :

$$(\leq n R)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} : \text{Card}(\{b \in \Delta^{\mathcal{I}} : (a, b) \in R^{\mathcal{I}}\}) \leq n\}$$

$$(\geq n R)^{\mathcal{I}} = \dots$$

- strictement** plus expressif que ALC :
les restrictions de cardinalité ne peuvent pas toutes être exprimées dans ALC

preuve non triviale (utilise la *bisimulation* entre interprétations)

ALCN : exercices

- décrire les concepts suivants :
 - ① être une personne avec au moins trois enfants
 - ② être une personne avec exactement trois enfants
 - ③ être un cours avec au plus 15 participants dont tous sont des étudiants de master
 - concepts primitifs : Cours, EtudiantEnMaster
 - rôle primitif : aParticipant
 - ④ être une grand-mère avec une fille qui a deux fils comme seuls enfants
- étendre la définition de la forme normale négative aux restrictions de cardinalité
- est-ce qu'on peut décrire le concept 'être une mère qui a (au moins) deux fils' ?

ALCQ : restrictions de cardinalité *qualifiées*

- grammaire des concepts ('BNF') :

$$C ::= N_C \mid \top \mid \perp \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \leq n N_R.C \mid \geq n N_R.C$$

où $N_C \in \text{NomsConcepts}$, $N_R \in \text{NomsRoles}$, $n \geq 0$

- $\leq n R.C$ = 'au plus n R-successeurs sont dans C '
- $\geq n R.C$ = 'au moins n R-successeurs sont dans C '
- interprétation :

$$(\leq n R.C)^I = \{a \in \Delta^I : \text{Card}(\{b \in \Delta : (a, b) \in R^I \text{ et } b \in C^I\}) \leq n\}$$

$$(\geq n R.C)^I = \dots$$

- ALCQ est au moins aussi expressif que ALCN :

- $\leq n R \stackrel{\text{def}}{=} \dots$
- $\geq n R \stackrel{\text{def}}{=} \dots$
- $\exists R.C \stackrel{\text{def}}{=} \dots$
- $\forall R.C \stackrel{\text{def}}{=} \dots$

- ALCQ est **strictement** plus expressif que ALCN
 - il y a des restrictions de cardinalité qualifiées qui ne peuvent pas être exprimées dans ALCN

Outline

- 1 Le langage ALC
- 2 Les langages ALCN et ALCQ
- 3 Relation avec la logique du premier ordre**
- 4 Structure et propriétés des TBox
- 5 Structure et propriétés des ABox
- 6 Tâches de raisonnement
- 7 Le langage SROIQ
- 8 Mécanismes de raisonnement

Rappel : la logique du premier ordre FOL

- langage :
 - variables objet x, y, \dots
 - prédicats : $P(t_1, \dots, t_n)$
 - t_i : termes, construits à partir de variables et fonctions
 - un prédicat particulier : $egal(t_1, t_2)$, écrit $t_1=t_2$
 - formules complexes : construites avec $\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \exists, \forall$
 $(\exists y \forall x P(x, y)) \rightarrow (\forall x \exists y P(x, y))$
- interprétations $\mathcal{I} = (\Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}})$:
 - domaine $\Delta^{\mathcal{I}}$ (non vide)
 - interprétation d'une variable: $(x)^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
 - interprétation d'un prédicat n -aire : $(P)^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^n$
 - où on a toujours : $egal^{\mathcal{I}} = \{(d, d) : d \in \Delta^{\mathcal{I}}\}$
- conditions de vérité :
 - $(\Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}}) \models_{FOL} P(t_1, \dots, t_n)$ ssi $((t_1)^{\mathcal{I}}, \dots, (t_n)^{\mathcal{I}}) \in (P)^{\mathcal{I}}$
 - $(\Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}}) \models_{FOL} \forall x \varphi$ ssi $(\Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}_x}) \models_{FOL} \varphi$ pour toute variante \mathcal{I}_x de \mathcal{I} en x
 - ...
- formule φ est valide ssi $(\Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}}) \models_{FOL} \varphi$, pour tout $(\Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}})$
- formule φ est satisfaisable ssi ...

Traduction de ALC en FOL

concept $C \mapsto$ formule $\Phi(C, x)$ de FOL

- le résultat de la traduction $\Phi(C, x)$ a une seule variable libre : x ("l'individu actuel")
- définition récursive :

$$\Phi(A, x) = A(x)$$

$$\Phi(\top, x) = \top$$

$$\Phi(\perp, x) = \perp$$

$$\Phi(\neg C, x) = \neg\Phi(C, x)$$

$$\Phi(C \sqcap D, x) = \Phi(C, x) \wedge \Phi(D, x)$$

$$\Phi(C \sqcup D, x) = \Phi(C, x) \vee \Phi(D, x)$$

$$\Phi(\forall R.C, x) = \forall y (R(x, y) \rightarrow \Phi(C, y))$$

$$\Phi(\exists R.C, x) = \dots$$

Traduction de ALC en FOL (suite)

Proposition

Pour tout concept C et pour toute interprétation $(\Delta^I, (.)^I)$:

$$C^I = \{a \in \Delta^I : \mathcal{I} \models_{FOL} \Phi(C, a)\}$$

démonstration par induction sur la forme de C ; on montre :

$$\text{pour tout } a \in \Delta^I : a \in C^I \text{ ssi } \mathcal{I} \models_{FOL} \Phi(C, a)$$

① si C est un concept atomique A alors :

$$\begin{aligned} a \in A^I & \text{ ssi } \mathcal{I} \models_{FOL} A(a) \\ & \text{ ssi } \mathcal{I} \models_{FOL} \Phi(A, a) \end{aligned}$$

② si C est de la forme \top alors ...

③ si C est de la forme \perp alors ...

④ si C est de la forme $\neg D$ alors :

$$\begin{aligned} a \in (\neg D)^I & \text{ ssi } a \notin D^I \\ & \text{ ssi } \mathcal{I} \not\models_{FOL} \Phi(D, a) \quad (\text{par H.I.}) \\ & \text{ ssi } \mathcal{I} \models_{FOL} \Phi(\neg D, a) \end{aligned}$$

⑤ si ...

Traduction de ALC en FOL (suite)

Proposition

Pour tout concept C et pour toute interprétation $(\Delta^I, (.)^I)$:

$$C^I = \{a \in \Delta^I : \mathcal{I} \Vdash_{FOL} \Phi(C, a)\}$$

démonstration par induction sur la forme de C ; on montre :

$$\text{pour tout } a \in \Delta^I : a \in C^I \text{ ssi } \mathcal{I} \Vdash_{FOL} \Phi(C, a)$$

① si C est un concept atomique A alors :

$$\begin{aligned} a \in A^I & \text{ ssi } \mathcal{I} \Vdash_{FOL} A(a) \\ & \text{ ssi } \mathcal{I} \Vdash_{FOL} \Phi(A, a) \end{aligned}$$

② si C est de la forme \top alors ...

③ si C est de la forme \perp alors ...

④ si C est de la forme $\neg D$ alors :

$$\begin{aligned} a \in (\neg D)^I & \text{ ssi } a \notin D^I \\ & \text{ ssi } \mathcal{I} \not\Vdash_{FOL} \Phi(D, a) \quad (\text{par H.I.}) \\ & \text{ ssi } \mathcal{I} \Vdash_{FOL} \Phi(\neg D, a) \end{aligned}$$

⑤ si ...

Traduction de ALC en FOL : exercices

$$\Phi(A, x) = A(x)$$

$$\Phi(\top, x) = \top$$

$$\Phi(\perp, x) = \perp$$

$$\Phi(\neg C, x) = \neg\Phi(C, x)$$

$$\Phi(C \sqcap D, x) = \Phi(C, x) \wedge \Phi(D, x)$$

$$\Phi(C \sqcup D, x) = \Phi(C, x) \vee \Phi(D, x)$$

$$\Phi(\forall R.C, x) = \forall y (R(x, y) \rightarrow \Phi(C, y))$$

$$\Phi(\exists R.C, x) = \exists y (R(x, y) \wedge \Phi(C, y))$$

- traduire en FOL :

- 1 $\text{Personne} \sqcap \exists \text{parentDe} . \exists \text{parentDe} . \text{Masculin}$

- 2 $(\forall \text{parentDe} . \text{Masculin}) \sqcap (\forall \text{parentDe} . \neg \text{Masculin})$

- 3 montrer que la traduction est satisfaisable : donner une interprétation de FOL $\mathcal{I} = (\Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}})$ où elle est vraie

Traduction de ALCN en FOL : exercices

- compléter la traduction de ALCN en FOL

$$\Phi(\geq 2 R, x) = \dots$$

$$\Phi(\geq n R, x) = \dots$$

$$\Phi(\leq n R, x) = \dots$$

- traduire en FOL

- Personne $\sqcap \leq 2$ parentDe

Traduction de ALCN en FOL : solution

$$\Phi(\leq 2 R, x) = \forall y_1 \forall y_2 \forall y_3 \left((R(x, y_1) \wedge R(x, y_2) \wedge R(x, y_3) \rightarrow (y_1 = y_2 \vee y_1 = y_3 \vee y_2 = y_3)) \right)$$

$$\Phi(\leq n R, x) = \forall y_1 \dots \forall y_{n+1} \left((R(x, y_1) \wedge \dots \wedge R(x, y_{n+1})) \rightarrow \left(\bigvee_{i < j \leq n+1} y_i = y_j \right) \right)$$

$$\Phi(\geq n R, x) = \exists y_1 \dots \exists y_n \left(R(x, y_1) \wedge \dots \wedge R(x, y_n) \wedge \left(\bigwedge_{i < j \leq n} \neg y_i = y_j \right) \right)$$

Traduction de ALC en logique modale

- logique multimodale K_n :

$$\begin{aligned} \Box_R \varphi &= \text{“}\varphi \text{ est vrai dans } \textit{tous} \text{ les états accessibles via } R\text{”} \\ &= \text{“}\varphi \text{ est nécessaire”} \\ &= \dots \end{aligned}$$

$$\begin{aligned} \Diamond_R \varphi &= \text{“}\varphi \text{ est vrai dans } \textit{quelques} \text{ les états accessibles via } R\text{”} \\ &= \text{“}\varphi \text{ est possible”} \\ &= \dots \end{aligned}$$

- fonction de traduction :

$$\begin{aligned} \mu(A) &= A && \text{si } A \text{ est un concept atomique} \\ \mu(\top) &= \top \\ \mu(\perp) &= \perp \\ \mu(\neg C) &= \neg \mu(C) \\ \mu(C \sqcap D) &= \mu(C) \wedge \mu(D) \\ \mu(C \sqcup D) &= \mu(C) \vee \mu(D) \\ \mu(\forall R.C) &= \Box_R \mu(C) \\ \mu(\exists R.C) &= \Diamond_R \mu(C) \end{aligned}$$

Outline

- 1 Le langage ALC
- 2 Les langages ALCN et ALCQ
- 3 Relation avec la logique du premier ordre
- 4 Structure et propriétés des TBox**
- 5 Structure et propriétés des ABox
- 6 Tâches de raisonnement
- 7 Le langage SROIQ
- 8 Mécanismes de raisonnement

TBox générale

- $C \sqsubseteq D$: 'axiome général d'inclusion de concept'

$$\mathcal{I} \models C \sqsubseteq D \text{ ssi } C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$$

- $C \equiv D$: 'axiome général d'équivalence de concept'

$$\mathcal{I} \models C \equiv D \text{ ssi } C^{\mathcal{I}} = D^{\mathcal{I}}$$

- TBox \mathcal{T} = ensemble fini d'axiomes généraux
- sémantique :

$$\mathcal{I} \models \mathcal{T} \text{ ssi } \mathcal{I} \models t \text{ pour tout } t \in \mathcal{T}$$

- exercices :

- 1 exprimer par un axiome que le 2nd argument de la relation R a la propriété C ('restriction du codomaine')
 - en FOL: $\forall x \forall y (R(x, y) \rightarrow C(y))$
- 2 exprimer par un axiome que le 1er argument de la relation R a la propriété C ('restriction du domaine')
 - en FOL : $\forall x \forall y (R(x, y) \rightarrow C(x))$

TBox générale

- $C \sqsubseteq D$: 'axiome général d'inclusion de concept'

$$\mathcal{I} \models C \sqsubseteq D \text{ ssi } C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$$

- $C \equiv D$: 'axiome général d'équivalence de concept'

$$\mathcal{I} \models C \equiv D \text{ ssi } C^{\mathcal{I}} = D^{\mathcal{I}}$$

- TBox \mathcal{T} = ensemble fini d'axiomes généraux
- sémantique :

$$\mathcal{I} \models \mathcal{T} \text{ ssi } \mathcal{I} \models t \text{ pour tout } t \in \mathcal{T}$$

- exercices :

- 1 exprimer par un axiome que le 2nd argument de la relation R a la propriété C ('restriction du codomaine')
 - en FOL: $\forall x \forall y (R(x, y) \rightarrow C(y))$
- 2 exprimer par un axiome que le 1er argument de la relation R a la propriété C ('restriction du domaine')
 - en FOL : $\forall x \forall y (R(x, y) \rightarrow C(x))$

TBox acyclique

- les axiomes d'inclusion peuvent être éliminés :
 - remplacer $C \sqsubseteq D$ par $C \equiv D \sqcap D'$, où D' est nouveau
 - équivalent à la TBox d'origine \mathcal{T} en ce qui concerne le langage de \mathcal{T}
- $C \equiv D$ est une **définition** ssi C est atomique
 - $\text{etreOncle} \equiv \text{Masculin} \sqcap \exists \text{frereDe} . \exists \text{parentDe} . \top$

⇒ problème : $A \equiv A \sqcup C$ n'est pas vraiment une définition...
- une TBox générale \mathcal{T} est une TBox **acyclique** ssi
 - 1 \mathcal{T} ne contient que des définitions
 - 2 pas de définitions multiples
 - interdit : $A \equiv C, A \equiv C' \in \mathcal{T}$ et $C \neq C'$
 - 3 pas de cycles
 - interdit : $A_1 \equiv \dots A_2 \dots, A_2 \equiv \dots A_3 \dots, \dots, A_n \equiv \dots A_1 \dots$
- si $A \equiv C \in \mathcal{T}$ alors A est un concept non-primitif

⇒ fréquent en pratique (ex. : SNOMED CT)

⇒ intéressant pour le calcul

Outline

- 1 Le langage ALC
- 2 Les langages ALCN et ALCQ
- 3 Relation avec la logique du premier ordre
- 4 Structure et propriétés des TBox
- 5 Structure et propriétés des ABox**
- 6 Tâches de raisonnement
- 7 Le langage SROIQ
- 8 Mécanismes de raisonnement

ABox = *assertion box*

- pour un ensemble donné de noms d'individus
 $NomsIndividus = \{a, b, \dots\}$
 - exemple : $NomsIndividus = \{Alice, Bob, Charles, \dots\}$
- $C(a)$: instance de concept
 - $Masculin(Charles)$
 - $(Masculin \sqcup Feminine)(Dominique)$
 - $(Masculin \sqcap Personne)(Charles)$
- $R(a, b)$: instance de rôle
 - $parentDe(Alice, Charles)$
- ABox \mathcal{A} = ensemble fini de $C(a)$ et $R(a, b)$
- sémantique : extension de la fonction d'interprétation \mathcal{I} aux noms d'individu
 - $(.)^{\mathcal{I}} : NomsIndividus \longrightarrow \Delta^{\mathcal{I}}$
 - $Alice^{\mathcal{I}} \in \Delta^{\mathcal{I}}, Bob^{\mathcal{I}} \in \Delta^{\mathcal{I}}, \dots$
 - hypothèse de nom unique : si $a \neq b$ alors $a^{\mathcal{I}} \neq b^{\mathcal{I}}$
 exemple : $Bob^{\mathcal{I}} \neq Charles^{\mathcal{I}}$
 - $\mathcal{I} \models \mathcal{A}$ ssi
 - 1 pour tout $C(a) \in \mathcal{A} : a^{\mathcal{I}} \in C^{\mathcal{I}}$
 - 2 pour tout $R(a, b) \in \mathcal{A} : (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

Exemple d'une KB généalogique

$$\mathcal{T}_{gen} = \{$$

Femme	≡	Personne ⊓ Feminine,
Homme	≡	Personne ⊓ Masculin,
Mere	≡	Femme ⊓ ∃parentDe.Personne,
Pere	≡	Homme ⊓ ∃parentDe.Personne,
Parent	≡	Mere ⊔ Pere,
MereSansFille	≡	Mere ⊓ ∀parentDe.¬Femme

$$\}$$

$$\mathcal{A}_{gen} = \{$$

Femme(Alice),
Homme(Bob),
parentDe(Alice, Charles),
parentDe(Alice, Denis),
parentDe(Bob, Charles)

$$\}$$

Différence avec les bases de données

- DL : hypothèse du monde ouvert (OWA)
 - ⇒ Alice peut avoir d'autres enfants
 - BD : hypothèse du monde clos (CWA)
 - ⇒ Charles et Denis sont les seuls enfants d'Alice
- DL : pas toujours hypothèse du nom unique (UNA)
 - ⇒ Charles et Denis peuvent désigner la même personne
 - BD : toujours hypothèse du nom unique
 - ⇒ Alice a au moins deux enfants
 - ⇒ avec la CWA : Alice a exactement deux enfants

Outline

- 1 Le langage ALC
- 2 Les langages ALCN et ALCQ
- 3 Relation avec la logique du premier ordre
- 4 Structure et propriétés des TBox
- 5 Structure et propriétés des ABox
- 6 Tâches de raisonnement**
- 7 Le langage SROIQ
- 8 Mécanismes de raisonnement

Tâches de raisonnement sur les concepts

C est **satisfaisable** ssi il existe une interprétation \mathcal{I} tel que $C^{\mathcal{I}} \neq \emptyset$

- exemples :
 - $\text{Mere} \sqcap \forall \text{parentDe}.\perp$ est satisfaisable
 - $\forall \text{parentDe}.C \sqcap \exists \text{parentDe}.\neg C$ est insatisfaisable
 - $\forall \text{parentDe}.C \sqcap \forall \text{parentDe}.\neg C$ est satisfaisable (!)
- subsomption de concepts :
 - $C \sqsubseteq D$ ssi $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ pour tout interprétation \mathcal{I}
 - réductible à un test de satisfaisabilité
 - $C \sqsubseteq D$ ssi $C \sqcap \neg D$ est insatisfaisable
- équivalence :
 - C et D sont équivalents ssi $C^{\mathcal{I}} = D^{\mathcal{I}}$ pour tout interprétation \mathcal{I}
 - réductible à deux tests de subsomption
- exclusivité (*disjointness*) :
 - C et D sont disjoints ssi $C^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset$ pour tout interprétation \mathcal{I}
 - réductible à un test de satisfaisabilité
 - C et D sont disjoints ssi $C \sqcap D$ est insatisfaisable

Tâches de raisonnement sur les concepts

C est **satisfaisable** ssi il existe une interprétation \mathcal{I} tel que $C^{\mathcal{I}} \neq \emptyset$

- exemples :

- $\text{Mere} \sqcap \forall \text{parentDe.} \perp$ est satisfaisable
- $\forall \text{parentDe.} C \sqcap \exists \text{parentDe.} \neg C$ est insatisfaisable
- $\forall \text{parentDe.} C \sqcap \forall \text{parentDe.} \neg C$ est satisfaisable (!)

- subsomption de concepts :

- $C \sqsubseteq D$ ssi $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ pour tout interprétation \mathcal{I}
- réductible à un test de satisfaisabilité
 - $C \sqsubseteq D$ ssi $C \sqcap \neg D$ est insatisfaisable

- équivalence :

- C et D sont équivalents ssi $C^{\mathcal{I}} = D^{\mathcal{I}}$ pour tout interprétation \mathcal{I}
- réductible à deux tests de subsomption

- exclusivité (*disjointness*) :

- C et D sont disjoints ssi $C^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset$ pour tout interprétation \mathcal{I}
- réductible à un test de satisfaisabilité
 - C et D sont disjoints ssi $C \sqcap D$ est insatisfaisable

Tâches de raisonnement p.r.à une TBox

\mathcal{C} est satisfaisable p.r.à \mathcal{T} ssi il existe \mathcal{I} tel que $\mathcal{I} \models \mathcal{T}$ et $\mathcal{C}^{\mathcal{I}} \neq \emptyset$

- exemple :

- $\text{Mere} \sqcap \forall \text{parentDe}.\perp$ est insatisfaisable p.r.à \mathcal{T}_{gen}

- subsomption p.r.à \mathcal{T} ...

- équivalence p.r.à \mathcal{T} ...

- exclusivité (*disjointness*) p.r.à \mathcal{T} ...

- classification = calculer toute hiérarchie de subsomption d'une TBox (graphe)

⇒ tout se réduit à des tests de satisfaisabilité p.r.à \mathcal{T}

⇒ et si la TBox \mathcal{T} est acyclique alors on peut faire encore plus simple ...

Tâches de raisonnement p.r.à une TBox

C est satisfaisable p.r.à \mathcal{T} ssi il existe \mathcal{I} tel que $\mathcal{I} \models \mathcal{T}$ et $C^{\mathcal{I}} \neq \emptyset$

- exemple :
 - $\text{Mere} \sqcap \forall \text{parentDe}.\perp$ est insatisfaisable p.r.à \mathcal{T}_{gen}
 - subsomption p.r.à \mathcal{T} ...
 - équivalence p.r.à \mathcal{T} ...
 - exclusivité (*disjointness*) p.r.à \mathcal{T} ...
 - classification = calculer toute hiérarchie de subsomption d'une TBox (graphe)
- ⇒ tout se réduit à des tests de satisfaisabilité p.r.à \mathcal{T}
- ⇒ et si la TBox \mathcal{T} est acyclique alors on peut faire encore plus simple ...

Comment se débarasser d'une TBox acyclique

- $C^{\mathcal{T}}$ = expansion de C par \mathcal{T} :
 - 1 remplacer chaque concept non-primitif A_i dans C par la définition de A_i dans \mathcal{T}
 - 2 itérer jusqu'à ce qu'il n'y a plus de concepts non-primitifs
 \Rightarrow termine car \mathcal{T} est acyclique (*point fixe*)

- $\mathcal{A}^{\mathcal{T}} = \{C^{\mathcal{T}}(a) : C(a) \in \mathcal{A}\} \cup \{R(a, b) : R(a, b) \in \mathcal{A}\}$

- exercice :

$$\mathcal{T}_{gen} = \left\{ \begin{array}{lll} \text{Femme} & \equiv & \text{Personne} \sqcap \text{Feminine}, \\ \text{Homme} & \equiv & \text{Personne} \sqcap \text{Masculin}, \\ \text{Mere} & \equiv & \text{Femme} \sqcap \exists \text{parentDe}. \text{Personne}, \\ \text{Pere} & \equiv & \text{Homme} \sqcap \exists \text{parentDe}. \text{Personne}, \\ \text{Parent} & \equiv & \text{Mere} \sqcup \text{Pere}, \\ \text{MereSansFille} & \equiv & \text{Mere} \sqcap \forall \text{parentDe}. \neg \text{Femme} \end{array} \right\}$$

- 1 identifier les concepts primitifs
- 2 trouver les expansions de tous les concepts non-primitifs
- 3 trouver l'expansion de $\text{Mere} \sqcap \forall \text{parentDe}. \perp$

Comment se débarasser d'une TBox acyclique

- $C^{\mathcal{T}}$ = expansion de C par \mathcal{T} :
 - 1 remplacer chaque concept non-primitif A_i dans C par la définition de A_i dans \mathcal{T}
 - 2 itérer jusqu'à ce qu'il n'y a plus de concepts non-primitifs
 \Rightarrow termine car \mathcal{T} est acyclique (*point fixe*)

- $\mathcal{A}^{\mathcal{T}} = \{C^{\mathcal{T}}(a) : C(a) \in \mathcal{A}\} \cup \{R(a, b) : R(a, b) \in \mathcal{A}\}$

- exercice :

$$\mathcal{T}_{gen} = \left\{ \begin{array}{lll} \text{Femme} & \equiv & \text{Personne} \sqcap \text{Feminine}, \\ \text{Homme} & \equiv & \text{Personne} \sqcap \text{Masculin}, \\ \text{Mere} & \equiv & \text{Femme} \sqcap \exists \text{parentDe. Personne}, \\ \text{Pere} & \equiv & \text{Homme} \sqcap \exists \text{parentDe. Personne}, \\ \text{Parent} & \equiv & \text{Mere} \sqcup \text{Pere}, \\ \text{MereSansFille} & \equiv & \text{Mere} \sqcap \forall \text{parentDe. } \neg \text{Femme} \end{array} \right\}$$

- 1 identifier les concepts primitifs
- 2 trouver les expansions de tous les concepts non-primitifs
- 3 trouver l'expansion de $\text{Mere} \sqcap \forall \text{parentDe. } \perp$

Comment se débarrasser d'une TBox acyclique (suite)

Proposition

Si \mathcal{T} est acyclique alors
 C est satisfaisable p.r. à \mathcal{T} ssi
 $C^{\mathcal{T}}$ est satisfaisable.

- exemple : $\text{Mere} \sqcap \forall \text{parentDe} . \perp$ satisfaisable p.r. à \mathcal{T}_{gen} ssi
 $(\text{Mere} \sqcap \forall \text{parentDe} . \perp)^{\mathcal{T}_{gen}} =$
 $\text{Personne} \sqcap \text{Feminine} \sqcap \exists \text{parentDe} . \text{Personne} \sqcap \forall \text{parentDe} . \perp$
 est satisfaisable tout court
 ... ce qui n'est pas le cas
- exercice :
 - trouver \mathcal{T} et A t.q. $A^{\mathcal{T}}$ est exponentiellement plus long que \mathcal{T}

Tâches de raisonnement p.r.à une ABox

une ABox \mathcal{A} est **satisfaisable** ssi il existe \mathcal{I} tel que $\mathcal{I} \models \mathcal{A}$

- satisfaisabilité p.r.à une TBox :
 - \mathcal{A} satisfaisable p.r.à \mathcal{T} ssi il existe \mathcal{I} tel que $\mathcal{I} \models \mathcal{T}$ et $\mathcal{I} \models \mathcal{A}$
 - si \mathcal{T} est acyclique : \mathcal{A} satisfaisable p.r.à \mathcal{T} ssi $\mathcal{A}^{\mathcal{T}}$ satisfaisable
 - subsume la satisfaisabilité d'un concept
 - C est satisfaisable ssi $\{C(a)\}$ est satisfaisable, pour un individu a quelconque
- inférence de propriétés :
 - $\mathcal{T} \cup \mathcal{A} \models C(a)$ ssi pour tout \mathcal{I} , si $\mathcal{I} \models \mathcal{T}$ et $\mathcal{I} \models \mathcal{A}$ alors $\mathcal{I} \models C(a)$
 - exemple :
 - $\mathcal{T}_{gen} \cup \{\text{GrandMere}(\text{Alice})\} \models \text{Mere}(\text{Alice})$
 - $\mathcal{T} \cup \mathcal{A} \models C(a)$ ssi $\mathcal{A} \cup \{\neg C(a)\}$ insatisfaisable p.r.à \mathcal{T}
- requête :
 - trouver tous les a tel que $\mathcal{T} \cup \mathcal{A} \models C(a)$
- réalisation :
 - calculer les noms de concept les plus spécifiques de la TBox pour un individu donné (cf. la classification)
 - Bob est instance de Personne, Masculin, Pere

Tâches de raisonnement p.r.à une ABox

une ABox \mathcal{A} est **satisfaisable** ssi il existe \mathcal{I} tel que $\mathcal{I} \models \mathcal{A}$

- satisfaisabilité p.r.à une TBox :
 - \mathcal{A} satisfaisable p.r.à \mathcal{T} ssi il existe \mathcal{I} tel que $\mathcal{I} \models \mathcal{T}$ et $\mathcal{I} \models \mathcal{A}$
 - si \mathcal{T} est acyclique : \mathcal{A} satisfaisable p.r.à \mathcal{T} ssi $\mathcal{A}^{\mathcal{T}}$ satisfaisable
 - **subsume la satisfaisabilité d'un concept**
 - C est satisfaisable ssi $\{C(a)\}$ est satisfaisable, pour un individu a quelconque
- inférence de propriétés :
 - $\mathcal{T} \cup \mathcal{A} \models C(a)$ ssi pour tout \mathcal{I} , si $\mathcal{I} \models \mathcal{T}$ et $\mathcal{I} \models \mathcal{A}$ alors $\mathcal{I} \models C(a)$
 - exemple :
 - $\mathcal{T}_{gen} \cup \{\text{GrandMere}(\text{Alice})\} \models \text{Mere}(\text{Alice})$
 - $\mathcal{T} \cup \mathcal{A} \models C(a)$ ssi $\mathcal{A} \cup \{\neg C(a)\}$ insatisfaisable p.r.à \mathcal{T}
- requête :
 - trouver tous les a tel que $\mathcal{T} \cup \mathcal{A} \models C(a)$
- réalisation :
 - calculer les noms de concept les plus spécifiques de la TBox pour un individu donné (cf. la classification)
 - Bob est instance de Personne, Masculin, Pere

Outline

- 1 Le langage ALC
- 2 Les langages ALCN et ALCQ
- 3 Relation avec la logique du premier ordre
- 4 Structure et propriétés des TBox
- 5 Structure et propriétés des ABox
- 6 Tâches de raisonnement
- 7 Le langage SROIQ**
- 8 Mécanismes de raisonnement

Le langage SROIQ

- essentiellement OWL DL

- ALCQ plus :

- rôle universel U
- hierarchies de rôle

$\text{frereDe} \circ \text{parentDe} \sqsubseteq \text{oncleDe}$

- nominaux = concepts particuliers désignant un seul individu

$\exists \text{numeroVol.}\{\text{AF3021}\}$

$\{\text{Alice}\} \approx \{\text{Anne}\}$

... donc pas de UNA !

$\{\text{Alice}\} \not\approx \{\text{Anne}\}$

équivalent : $\{\text{Alice}\} \sqcap \{\text{Anne}\} \sqsubseteq \perp$

$\{\text{Alice}\} \sqcup \{\text{Anne}\} \sqsubseteq \text{Mere}$

équivalent : $\text{Mere}(\text{Alice}), \text{Mere}(\text{Anne})$

$\text{Mere} \sqsubseteq \{\text{Alice}\} \sqcup \{\text{Anne}\}$... ne peut pas être exprimé en ALC !

- rôles inverses

parentDe^-

- *Self*

$\neg \exists \text{parentDe. Self}$

Le langage SROIQ

- grammaire :

rôles : $R ::= U \mid N_R \mid N_R^-$

concepts : $C ::= N_C \mid \{N_I\} \mid \top \mid \perp \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid$

$\forall R.C \mid \exists R.C \mid \leq n R.C \mid \geq n R.C \mid \exists R.Self$

assertions : $\alpha ::= C(N_I) \mid R(N_I, N_I) \mid N_I \approx N_I \mid N_I \neq N_I$

axiomes : $\tau ::= C \sqsubseteq C \mid C \equiv C \mid$

$R \sqsubseteq R \mid R \equiv R \mid R \circ R \sqsubseteq R \mid Disjoint(R, R)$

- en plus :

- dans $\leq n R.C$, $\geq n R.C$, $\exists R.Self$, $Disjoint(R, R)$ les R doivent être *simples* ...
- les inclusions de rôle $R \sqsubseteq R$ ne doivent pas générer des cycles (ou seulement des formes simples de dépendances cycliques)

Outline

- 1 Le langage ALC
- 2 Les langages ALCN et ALCQ
- 3 Relation avec la logique du premier ordre
- 4 Structure et propriétés des TBox
- 5 Structure et propriétés des ABox
- 6 Tâches de raisonnement
- 7 Le langage SROIQ
- 8 Mécanismes de raisonnement**

Décidabilité et complexité

- le problème de satisfaisabilité d'un concept est décidable :
 - PSPACE complet si la TBox est acyclique
 - EXPTIME complet en général
- ⇒ suite : procédure de décision via tableaux pour la satisfaisabilité d'une ABox
- ⇒ subsume toutes les autres tâches de raisonnement dans le cas d'une TBox acyclique

La méthode des tableaux

- entrée : ABox \mathcal{A}
- sortie : ‘oui’ si \mathcal{A} est satisfaisable, ‘non’ sinon
- hypothèse : \mathcal{A} en **forme normale négative**
 - négations seulement devant les atomes
- notions et notations :
 - configuration du tableau = ensemble fini d’ABox’es S
 - initialisation : $S = \{\mathcal{A}\}$
 - S, \mathcal{A} à la place de $S \cup \{\mathcal{A}\}$
 - $\mathcal{A}[X] = “X \text{ est un sous-ensemble de } \mathcal{A}”$
 - format des règles :

$$\frac{S, \mathcal{A}[X]}{S, \mathcal{A}[Y_1], \dots, \mathcal{A}[Y_n]}$$

- si $X \subseteq \mathcal{A}$ alors ajouter Y_i à \mathcal{A}
- applicable si $Y_1 \not\subseteq \mathcal{A}, \dots$ et $Y_n \not\subseteq \mathcal{A}$

Règles de tableau pour ALC

- règle pour \sqcap :

$$\frac{S, \mathcal{A}[(C \sqcap D)(a)]}{S, \mathcal{A}[C(a), D(a)]}$$

- règle pour \sqcup :

$$\frac{S, \mathcal{A}[(C \sqcup D)(a)]}{S, \mathcal{A}[C(a)], \mathcal{A}[D(a)]}$$

- règle pour \exists :

$$\frac{S, \mathcal{A}[(\exists R.C)(a)]}{S, \mathcal{A}[R(a, b), C(b)]} \quad ((b) \notin \mathcal{A})$$

b = nouveau individu

- règle pour \forall :

$$\frac{S, \mathcal{A}[(\forall R.C)(a), R(a, b)]}{S, \mathcal{A}[C(b)]}$$

Règles de tableau pour ALC

- règle pour \sqcap :

$$\frac{S, \mathcal{A}[(C \sqcap D)(a)]}{S, \mathcal{A}[C(a), D(a)]}$$

- règle pour \sqcup :

$$\frac{S, \mathcal{A}[(C \sqcup D)(a)]}{S, \mathcal{A}[C(a)], \mathcal{A}[D(a)]}$$

- règle pour \exists :

$$\frac{S, \mathcal{A}[(\exists R.C)(a)]}{S, \mathcal{A}[R(a, b), C(b)]} \quad ((b) \notin \mathcal{A})$$

b = nouveau individu

- règle pour \forall :

$$\frac{S, \mathcal{A}[(\forall R.C)(a), R(a, b)]}{S, \mathcal{A}[C(b)]}$$

Tableaux : contradictions manifeste

Une ABox \mathcal{A} contient une contradiction manifeste (*clash*) si il existe un a tel que

- $\perp(a) \in \mathcal{A}$, ou
- $C(a) \in \mathcal{A}$ et $\neg C(a) \in \mathcal{A}$

Sinon \mathcal{A} est *ouvert*.

- exemple :

$\mathcal{A} = \{ \text{Personne} \sqcap (\exists \text{parentDe. Personne}) \sqcap \forall \text{parentDe.} \perp(a) \}$

① déjà en forme normale négative

② $\{ \{ \text{Personne}(a), \exists \text{parentDe. Personne}(a), \forall \text{parentDe.} \perp(a) \} \}$

(règle pour \sqcap)

③ $\{ \{ \text{Personne}(a), \exists \text{parentDe. Personne}(a), \forall \text{parentDe.} \perp(a), \text{Personne}(b), \text{parentDe}(a, b) \} \}$

(règle pour \exists)

④ $\{ \{ \text{Personne}(a), \exists \text{parentDe. Personne}(a), \forall \text{parentDe.} \perp(a), \text{Personne}(b), \text{parentDe}(a, b), \perp(b) \} \}$

(règle pour \forall)

⑤ contradiction manifeste : $\perp(b)$

Tableaux : contradictions manifeste

Une ABox \mathcal{A} contient une contradiction manifeste (*clash*) si il existe un a tel que

- $\perp(a) \in \mathcal{A}$, ou
- $C(a) \in \mathcal{A}$ et $\neg C(a) \in \mathcal{A}$

Sinon \mathcal{A} est *ouvert*.

- exemple :

$\mathcal{A} = \{\text{Personne} \sqcap (\exists \text{parentDe. Personne}) \sqcap \forall \text{parentDe.} \perp(a)\}$

① déjà en forme normale négative

② $\{\{\text{Personne}(a), \exists \text{parentDe. Personne}(a), \forall \text{parentDe.} \perp(a)\}\}$

(règle pour \sqcap)

③ $\{\{\text{Personne}(a), \exists \text{parentDe. Personne}(a), \forall \text{parentDe.} \perp(a),$
 $\text{Personne}(b), \text{parentDe}(a, b)\}\}$

(règle pour \exists)

④ $\{\{\text{Personne}(a), \exists \text{parentDe. Personne}(a), \forall \text{parentDe.} \perp(a),$
 $\text{Personne}(b), \text{parentDe}(a, b), \perp(b)\}\}$

(règle pour \forall)

⑤ contradiction manifeste : $\perp(b)$

Raisonnement : exercice complet (1)

$$\{A \sqsubseteq \exists R.B\} \cup \{A(a)\} \models? (\exists R.B)(a)$$

(tâche = inférence de propriétés)

- 1 transformation en le problème de satisfaisabilité d'une ABox :

$$\{A(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \sqsubseteq \exists R.B\} ?$$

- 2 éliminer \sqsubseteq :

$$\{A(a), (\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \equiv A' \sqcap \exists R.B\} ?$$

\Rightarrow TBox acyclique !

- 3 expansion de la ABox par la TBox : $A^{\{A \equiv A' \sqcap \exists R.B\}} = A' \sqcap \exists R.B$

$$\{(A' \sqcap \exists R.B)(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable ?}$$

- 4 mise en forme normale négative :

$$\{(A' \sqcap \exists R.B)(a), (\forall R. \neg B)(a)\} \text{ satisfaisable ?}$$

- 5 construction d'un tableau ...

Raisonnement : exercice complet (1)

$$\{A \sqsubseteq \exists R.B\} \cup \{A(a)\} \models? (\exists R.B)(a)$$

(tâche = inférence de propriétés)

- 1 transformation en le problème de satisfaisabilité d'une ABox :

$$\{A(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \sqsubseteq \exists R.B\} ?$$

- 2 éliminer \sqsubseteq :

$$\{A(a), (\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \equiv A' \sqcap \exists R.B\} ?$$

\Rightarrow TBox acyclique !

- 3 expansion de la ABox par la TBox : $A^{\{A \equiv A' \sqcap \exists R.B\}} = A' \sqcap \exists R.B$

$$\{(A' \sqcap \exists R.B)(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable ?}$$

- 4 mise en forme normale négative :

$$\{(A' \sqcap \exists R.B)(a), (\forall R. \neg B)(a)\} \text{ satisfaisable ?}$$

- 5 construction d'un tableau ...

Raisonnement : exercice complet (1)

$$\{A \sqsubseteq \exists R.B\} \cup \{A(a)\} \models? (\exists R.B)(a)$$

(tâche = inférence de propriétés)

- ① transformation en le problème de satisfaisabilité d'une ABox :

$$\{A(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \sqsubseteq \exists R.B\} ?$$

- ② éliminer \sqsubseteq :

$$\{A(a), (\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \equiv A' \sqcap \exists R.B\} ?$$

\Rightarrow TBox acyclique !

- ③ expansion de la ABox par la TBox : $A^{\{A \equiv A' \sqcap \exists R.B\}} = A' \sqcap \exists R.B$

$$\{(A' \sqcap \exists R.B)(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable ?}$$

- ④ mise en forme normale négative :

$$\{(A' \sqcap \exists R.B)(a), (\forall R. \neg B)(a)\} \text{ satisfaisable ?}$$

- ⑤ construction d'un tableau ...

Raisonnement : exercice complet (1)

$$\{A \sqsubseteq \exists R.B\} \cup \{A(a)\} \models? (\exists R.B)(a)$$

(tâche = inférence de propriétés)

- 1 transformation en le problème de satisfaisabilité d'une ABox :

$$\{A(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \sqsubseteq \exists R.B\} ?$$

- 2 éliminer \sqsubseteq :

$$\{A(a), (\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \equiv A' \sqcap \exists R.B\} ?$$

\Rightarrow TBox acyclique !

- 3 expansion de la ABox par la TBox : $A^{\{A \equiv A' \sqcap \exists R.B\}} = A' \sqcap \exists R.B$

$$\{(A' \sqcap \exists R.B)(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable ?}$$

- 4 mise en forme normale négative :

$$\{(A' \sqcap \exists R.B)(a), (\forall R. \neg B)(a)\} \text{ satisfaisable ?}$$

- 5 construction d'un tableau ...

Raisonnement : exercice complet (1)

$$\{A \sqsubseteq \exists R.B\} \cup \{A(a)\} \models? (\exists R.B)(a)$$

(tâche = inférence de propriétés)

- 1 transformation en le problème de satisfaisabilité d'une ABox :

$$\{A(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \sqsubseteq \exists R.B\} ?$$

- 2 éliminer \sqsubseteq :

$$\{A(a), (\neg \exists R.B)(a)\} \text{ satisfaisable p.r.à } \{A \equiv A' \sqcap \exists R.B\} ?$$

\Rightarrow TBox acyclique !

- 3 expansion de la ABox par la TBox : $A^{\{A \equiv A' \sqcap \exists R.B\}} = A' \sqcap \exists R.B$

$$\{(A' \sqcap \exists R.B)(a)\} \cup \{(\neg \exists R.B)(a)\} \text{ satisfaisable ?}$$

- 4 mise en forme normale négative :

$$\{(A' \sqcap \exists R.B)(a), (\forall R. \neg B)(a)\} \text{ satisfaisable ?}$$

- 5 construction d'un tableau ...

Raisonnement : exercice complet (2)

$\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ satisfaisable ?

construction du tableau :

- ① $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$
- ② $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a)\}$ (règle pour \sqcap)
- ③ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b)\}$
(règle pour \exists ; b nouveau)
- ④ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b), (\neg B)(b)\}$
(règle pour \forall)

\Rightarrow contradiction manifeste

\Rightarrow la ABox $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ est insatisfaisable

\Rightarrow de la KB $\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\}$ on peut inférer $(\exists R.B)(a)$:

$$\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\} \models (\exists R.B)(a)$$

Raisonnement : exercice complet (2)

$\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ satisfaisable ?

construction du tableau :

- ① $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$
- ② $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a)\}$ (règle pour \sqcap)
- ③ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b)\}$
(règle pour \exists ; b nouveau)
- ④ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b), (\neg B)(b)\}$
(règle pour \forall)

\Rightarrow contradiction manifeste

\Rightarrow la ABox $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ est insatisfaisable

\Rightarrow de la KB $\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\}$ on peut inférer $(\exists R.B)(a)$:

$$\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\} \models (\exists R.B)(a)$$

Raisonnement : exercice complet (2)

$\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ satisfaisable ?

construction du tableau :

- ① $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$
- ② $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a)\}$ (règle pour \sqcap)
- ③ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b)\}$
(règle pour \exists ; b nouveau)
- ④ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b), (\neg B)(b)\}$
(règle pour \forall)

\Rightarrow contradiction manifeste

\Rightarrow la ABox $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ est insatisfaisable

\Rightarrow de la KB $\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\}$ on peut inférer $(\exists R.B)(a)$:

$$\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\} \models (\exists R.B)(a)$$

Raisonnement : exercice complet (2)

$\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ satisfaisable ?

construction du tableau :

- ① $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$
- ② $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a)\}$ (règle pour \sqcap)
- ③ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b)\}$
(règle pour \exists ; b nouveau)
- ④ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b), \neg B(b)\}$
(règle pour \forall)

\Rightarrow contradiction manifeste

\Rightarrow la ABox $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ est insatisfaisable

\Rightarrow de la KB $\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\}$ on peut inférer $(\exists R.B)(a)$:

$$\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\} \models (\exists R.B)(a)$$

Raisonnement : exercice complet (2)

$\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ satisfaisable ?

construction du tableau :

- ① $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$
- ② $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a)\}$ (règle pour \sqcap)
- ③ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b)\}$
(règle pour \exists ; b nouveau)
- ④ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b), \neg B(b)\}$
(règle pour \forall)

\Rightarrow contradiction manifeste

\Rightarrow la ABox $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ est insatisfaisable

\Rightarrow de la KB $\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\}$ on peut inférer $(\exists R.B)(a)$:

$$\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\} \models (\exists R.B)(a)$$

Raisonnement : exercice complet (2)

$\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ satisfaisable ?

construction du tableau :

- ① $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$
- ② $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a)\}$ (règle pour \sqcap)
- ③ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b)\}$
(règle pour \exists ; b nouveau)
- ④ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b), \neg B(b)\}$
(règle pour \forall)

⇒ contradiction manifeste

⇒ la ABox $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ est insatisfaisable

⇒ de la KB $\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\}$ on peut inférer $(\exists R.B)(a)$:

$\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\} \models (\exists R.B)(a)$

Raisonnement : exercice complet (2)

$\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ satisfaisable ?

construction du tableau :

- ① $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$
- ② $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a)\}$ (règle pour \sqcap)
- ③ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b)\}$
(règle pour \exists ; b nouveau)
- ④ $\{A'(a), (\exists R.B)(a), (\forall R.\neg B)(a), R(a, b), B(b), \neg B(b)\}$
(règle pour \forall)

⇒ contradiction manifeste

⇒ la ABox $\{(A' \sqcap \exists R.B)(a), (\forall R.\neg B)(a)\}$ est insatisfaisable

⇒ de la KB $\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\}$ on peut inférer $(\exists R.B)(a)$:

$$\{A \equiv A' \sqcap \exists R.B\} \cup \{A(a)\} \models (\exists R.B)(a)$$

Raisonnement : un autre exercice complet (1)

$$\{A \equiv (\exists R.B_1) \sqcap (\exists R.B_2)\} \models? A \sqsubseteq \exists R.(B_1 \sqcap B_2)$$

(tâche = subsomption de concepts p.r.à une TBox)

- 1 transformation en problème de satisfaisabilité de concept p.r.à une TBox :

$$A \sqcap \neg \exists R.(B_1 \sqcap B_2) \text{ satisfaisable p.r.à } \{A \equiv (\exists R.B_1) \sqcap (\exists R.B_2)\} ?$$

- 2 expansion du concept par la TBox (qui est acyclique) :

$$(\exists R.B_1) \sqcap (\exists R.B_2) \sqcap \neg \exists R.(B_1 \sqcap B_2) \text{ satisfaisable ?}$$

- 3 transformation en problème de satisfaisabilité d'une ABox :

$$\{((\exists R.B_1) \sqcap (\exists R.B_2) \sqcap \neg \exists R.(B_1 \sqcap B_2))(a)\} \text{ satisfaisable ?}$$

- 4 mise en forme normale négative :

$$\{((\exists R.B_1) \sqcap (\exists R.B_2) \sqcap \forall R.(\neg B_1 \sqcup \neg B_2))(a)\} \text{ satisfaisable ?}$$

- 5 construction d'un tableau ...

Raisonnement : un autre exercice complet (2)

$\{((\exists R.B_1) \sqcap (\exists R.B_2) \sqcap \forall R.(\neg B_1 \sqcup \neg B_2))(a)\}$ satisfaisable ?

construction du tableau :

- 1 $\{ \{ (\exists R.B_1) \sqcap (\exists R.B_2) \sqcap \forall R.(\neg B_1 \sqcup \neg B_2)(a) \} \}$
- 2 $\{ \{ ((\exists R.B_1)(a), (\exists R.B_2)(a), (\forall R.(\neg B_1 \sqcup \neg B_2))(a)) \} \}$
(règle pour \sqcap , deux fois)
- 3 $\{ \{ ((\exists R.B_1)(a), (\exists R.B_2)(a), (\forall R.(\neg B_1 \sqcup \neg B_2))(a),$
 $R(a, b_1), B_1(b_1), R(a, b_2), B_2(b_2)) \} \}$ (règle pour \exists , deux fois)
- 4 $\{ \{ ((\exists R.B_1)(a), (\exists R.B_2)(a), (\forall R.(\neg B_1 \sqcup \neg B_2))(a),$
 $R(a, b_1), B_1(b_1), R(a, b_2), B_2(b_2),$
 $(\neg B_1 \sqcup \neg B_2)(b_1), (\neg B_1 \sqcup \neg B_2)(b_2)) \} \} = \{ \mathcal{A}_0 \}$
(règle pour \forall , deux fois)
- 5 $\{ \mathcal{A}_0 \cup \{ (\neg B_1)(b_1) \}, \mathcal{A}_0 \cup \{ (\neg B_2)(b_1) \} \}$ (règle pour \sqcup)
- 6 $\{ \mathcal{A}_0 \cup \{ (\neg B_1)(b_1) \} \cup \{ (\neg B_1)(b_2) \}, \mathcal{A}_0 \cup \{ (\neg B_1)(b_1) \} \cup \{ (\neg B_2)(b_2) \},$
 $\mathcal{A}_0 \cup \{ (\neg B_2)(b_1) \} \cup \{ (\neg B_1)(b_2) \}, \mathcal{A}_0 \cup \{ (\neg B_2)(b_1) \} \cup \{ (\neg B_2)(b_2) \} \}$
(règle pour \sqcup , deux fois)

\Rightarrow 3ème ABox ouverte \Rightarrow satisfaisable $\Rightarrow \mathcal{T} \not\models A \sqsubseteq \exists R.(B_1 \sqcap B_2)$

Tableaux pour ALC : la grande finale

- La configuration S est **satisfaisable** ssi il existe une ABox $\mathcal{A}_i \in S$ et une interprétation \mathcal{I} t.q. $\mathcal{I} \models \mathcal{A}_i$.
- La configuration S est **saturée** ssi plus aucune règle ne peut être appliquée

Proposition (terminaison)

Pour toute entrée \mathcal{A} , la procédure de construction de tableau termine par une configuration saturée.

Proposition (adéquation)

Si la ABox \mathcal{A} est satisfaisable alors toutes les configurations obtenues à partir de $\{\mathcal{A}\}$ sont ouvertes.

Proposition (complétude)

Si S est une configuration saturée et ouverte alors il existe une ABox $\mathcal{A} \in S$ tel que \mathcal{A} est satisfaisable.

Tableaux pour des extensions de ALC

- ALCN
- inclusion de concepts généraux (cycliques)
 - requiert test de boucle
 - EXPTIME complet
- inclusion de rôles
- rôles transitifs
- restrictions de cardinalité qualifiées
 - indécidable si combiné avec intersection de rôles !
 - indécidable si combiné avec rôles transitifs !
- constructeur 'un-parmi a_1, \dots, a_n ' : $\{a_1, \dots, a_n\}$
 - $(\{a_1, \dots, a_n\})^I = \{a_1^I, \dots, a_n^I\}$
 - exemple : $\text{Personne} \equiv \{\text{Alice}, \text{Bob}, \text{Charles}\}$
- domaines concrètes
 - nombres entiers : $\text{Adulte} \equiv \text{Personne} \sqcap \exists \text{AgeDe} . \geq 18$
 - ...

Conclusion sur la partie DL

- DL = représenter + raisonner
- DL = plusieurs logiques
 - expressivité (quelles restrictions du langage permis par le domaine ?)
 - complexité (P, NP, PSPACE, EXPTIME)
- DL base de OWL (OWL DL)
- recherches actuelles :
 - trouver des langages d'interrogation (*query languages*) à complexité basse
 - restrictions drastiques du langage : "DL-lite"
 - comment réviser une TBox ?
 - comment mettre à jour une ABox ?
 - comment réparer une ABox qui est devenue insatisfaisable p.r.à une TBox ?
 - $\{Mere(Alice), \forall parentDe. \perp(Alice)\}$ insatisfaisable p.r.à \mathcal{T}_{gen}