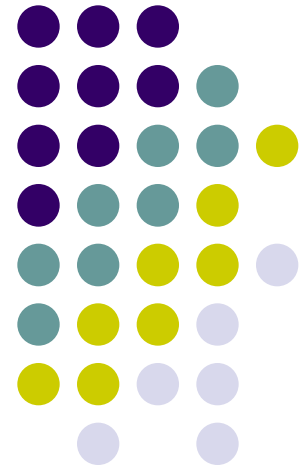
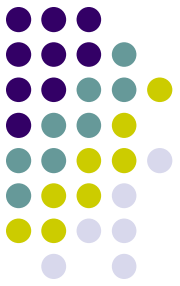


Chapitre IV: Logiques de description

IV.2 familles des logiques de description

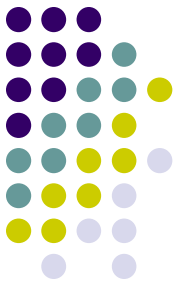




Plan du cours

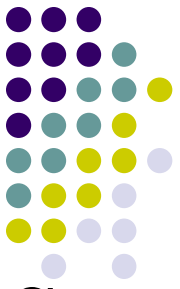
- Introduction
- Le langage basique AL
- Extensions du langage AL
- La famille SH
- Remarques (équivalence entre langages)
- Conclusion

introduction



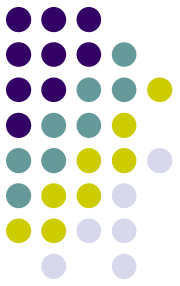
- Il existe plusieurs langage LD :AL,ALC,ALE |....
- Les LD se distinguent par les constructeurs qu'elles proposent :
- plus elles ont de constructeurs, plus elles sont expressives, et ont des chances d'être non décidables ou de complexité très élevée
- les LD trop peu expressives ne permettent pas de représenter des domaines complexes.

Le langage basique AL



- Le langage AL (pour Attributive Language) a été introduit comme un langage minimal ayant un intérêt pratique.
- Les autres langages de la famille AL en sont une extension

Le langage basique AL



- **Conventions**
- A, B :dénotent des concepts atomiques
- C,D : dénotent des concepts définis

Le langage basique AL

Syntaxe

- Les descriptions sont générées selon les règles syntaxiques suivantes

$C, D \rightarrow$	A		(concept atomique)
	$\top$		(concept universel)
	$\perp$		(concept bottom)
	$\neg A$		(negation atomique)
	$C \sqcap D$		(intersection)
	$\forall R.C$		(restriction de valeur)
	$\exists R.\top$		(quantificateur existentiel limité)

Le langage basique AL



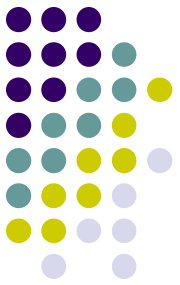
Constructeurs d' $\mathcal{AL}$:

$\neg A$	la négation atomique
$C \sqcap D$	l'intersection de concepts
$\forall R.C$	la restriction de valeur (quantification universelle complète)
$\exists R.T$	la quantification existentielle limitée *

- Remarques

- la négation n'est appliquée qu'aux concepts atomiques
- Seulement le concept universel est permis à la portée du quantificateur existentiel
- ne permet pas la spécification de rôles composés⁷

Le langage basique AL

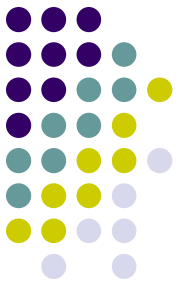


Signification des constructeurs avec quantification

- **$\exists R.T$** : les individus qui ont au moins une relation de type R avec un individu de n'importe quel type.
- **$\forall R.C$** : les individus dont toutes les relations de type R se font avec des individus de type C.
- Exemples: sur les cités .
 - $\text{cit Moderne} = \exists \text{ contientEcole}.T$
 - $\text{cit Compacte} = \forall \text{ contient}. \text{Logement}$
 - $\quad \quad \quad \forall \text{ contient}. \neg \text{Ecole},$
 - $\quad \quad \quad \exists \text{ voisin}.T,$
 - $\quad \quad \quad \exists \text{ voisin} (\exists \text{ contientEcole } T)$

Le langage basique AL

•Sémantique ALC



Un interprétation I est une paire $(\Delta^I, \cdot^I)$ où

- Δ^I est un ensemble (le domaine) et
- $\cdot^I$ est une fonction d'interprétation :

Tel que

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

$$\perp^I = \emptyset$$

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

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

$$(\forall R.C)^I = \{a \in \Delta^I \mid \forall b. (a, b) \in R^I \rightarrow b \in C^I\}$$

$$(\exists R.\top)^I = \{a \in \Delta^I \mid \exists b. (a, b) \in R^I\}$$

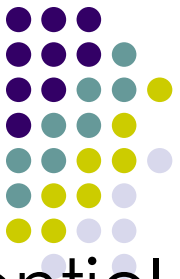
Extensions du langage AL (la famille AL)



$$\mathcal{AL} = \{\top, \perp, \neg A, C \sqcap D, \forall r.C, \exists r\}$$

- $\mathcal{ALC} = \mathcal{AL} \cup \{\neg C\}$ (négation de concepts primitifs ou définis).
- $\mathcal{ALU} = \mathcal{AL} \cup \{C \sqcup D\}$ (disjonction de concepts).
- $\mathcal{ALE} = \mathcal{AL} \cup \{\exists r.C\}$ (quantification existentielle typée).
- $\mathcal{ALN} = \mathcal{AL} \cup \{\geq n r, \leq n r\}$ (cardinalité des rôles, en particulier, $(\exists r) \equiv (\geq 1 r)$).
- $\mathcal{ALR} = \mathcal{AL} \cup \{r_1 \sqcap r_2\}$ (conjonction de rôles).
- Et encore les rôles inverses ($\mathcal{I}$), une hiérarchie de rôles ($\mathcal{H}$), le constructeur range et la restriction sur les co-domaines ($\mathcal{Q}$) ...

Extensions du langage AL

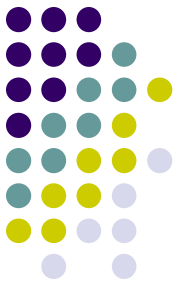


Quantification existentielle complète $\exists R.C$ (ALE)

- En AL, on peut utiliser le quantificateur existentiel pour spécifier qu'une entité doit avoir au moins une relation avec un autre objet mais on ne peut pas spécifier la classe de cet autre objet. Ceci limite fortement l'expressivité du langage
- *Exemple : les gens possédant au moins une voiture*
- *Solution : étendre AL par quantification existentielle complète $\exists R.C$*
- *Sémantique :*

$$I(\exists R.C) = \{a \in \mid \exists b. (a, b) \in I(R) \wedge b \in I(C)\}$$

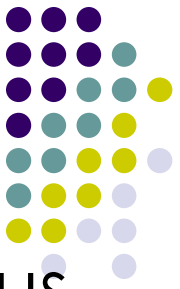
Extensions du langage AL



Négation sans restriction (ALC)

- on peut étendre le langage en ajoutant la possibilité d'utiliser la négation sans aucune restriction
- Cette caractéristique est désignée par le symbole C (pour *complément*)
- Sémantique : $I(\neg C) = \{a \in \Delta \mid I(C)\}$
- Exemple:
- les gens qui n'ont pas d'enfants de sexe masculin:
- $\neg \exists a \text{Enfant.Homme}$

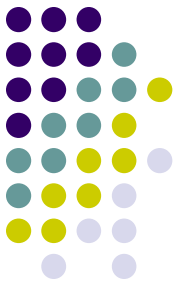
Extensions du langage AL



L'union (ALU)

- On peut représenter l'ensemble des individus qui appartiennent soit à la classe C, soit à la classe D,
- on écrirait la description $C \sqcup D$
- Sémantique : $I(C \sqcup D) = I(C) \cup I(D)$
- *Exemple:*
- Un magasin qui ne vend que des chats et des chiens : $\text{Magasin} \sqcap \forall \text{vend.}(\text{Chat} \sqcup \text{Chien})$

Extensions du langage AL



Constructeur de fonctions (ALF)

- Pour spécifier que la relation R en fait une fonction.

telle que aucune entité ne peut être reliée à plus d'une autre entité

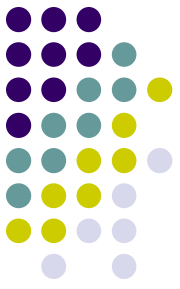
- par cette relation On écrira donc un axiome de la forme **Fun(R)** pour dire qu'un rôle R est une fonction.

- *Exemple :*

Fun(aPere)

fils $\equiv$ Homme $\sqcap$ $\exists$ aPere.T

Extensions du langage AL



Restriction de cardinalité (ALN)

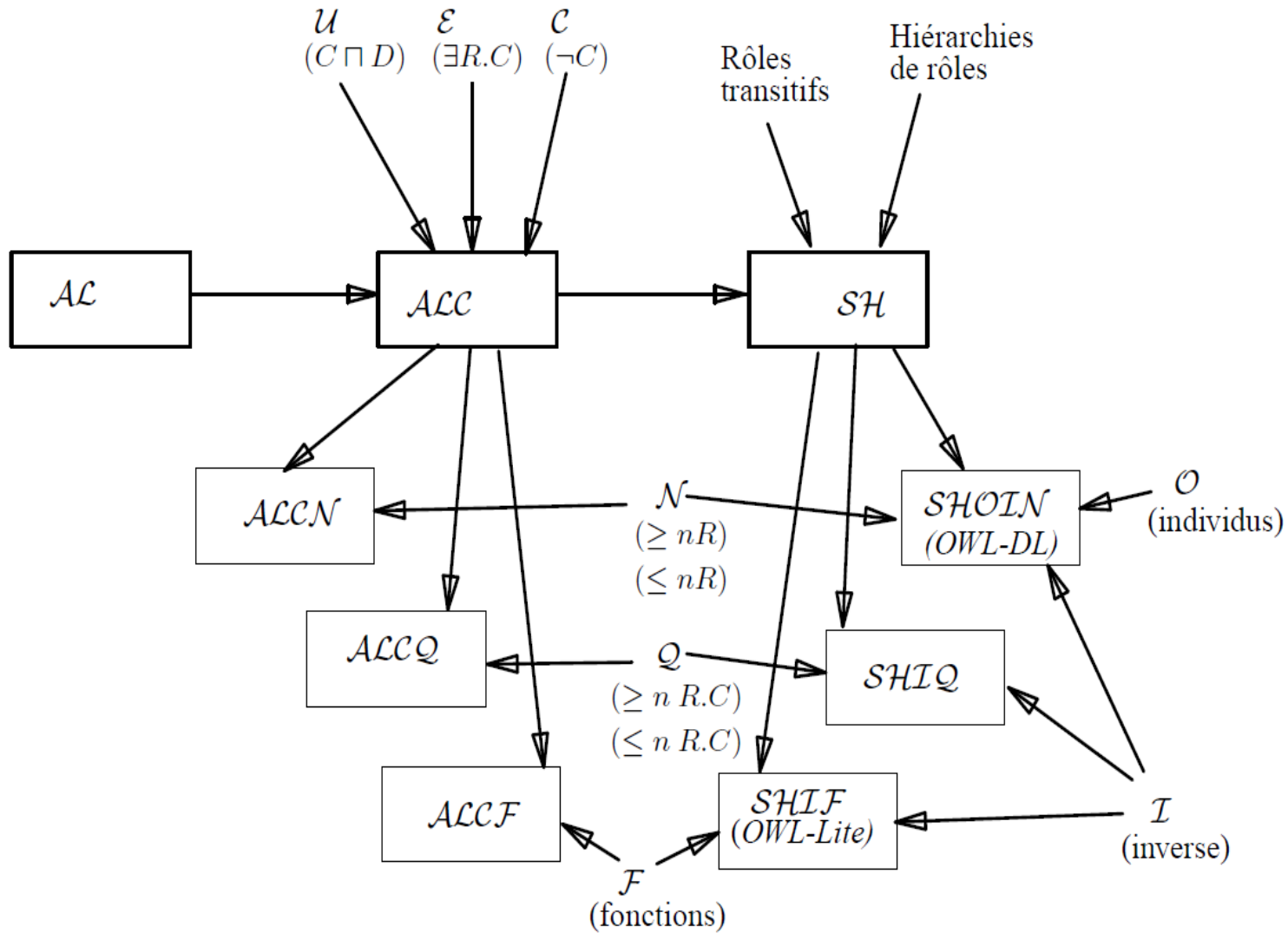
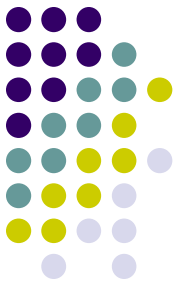
- Deux autres constructeurs, exprimant la restriction de cardinalité (dénommés N), $\leq n R$ et $\geq n R$
- Sémantique :
- $I(\geq n R) = \{a \in \text{dom} \mid |\{b \mid (a, b) \in I(R)\}| \geq n\}$
- $I(\leq n R) = \{a \in \text{dom} \mid |\{b \mid (a, b) \in I(R)\}| \leq n\}$
- Exemple :
- le concept de père qui a exactement deux enfants :
 - $\text{Homme} \sqcap 2 \text{Enf} \sqcap \geq 2 \text{aEnfant} \sqcap \leq 2 \text{aEnfant}$
 - $\text{FemmeMariée} \sqcap \text{Femme} \sqcap 1 \text{mariéAvec}$

La famille SH



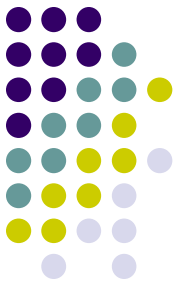
- Les langages de la familles AL nous permettent de faire des descriptions complexes de concepts,
- mais sont plutôt limités en ce qui concerne les rôles
- SH est un une logique expressive :
 - (S)la possibilité d'établir qu'une relation est une sous-propriété d'une autre relation(Exemple aDescendant ,parentDe ,mereDe ,pereDe...)
 - la possibilité de définir une relation transitive (exemple frèreDe, aDescendant ..)
 - **S=ALC+TRANSITIVITE ,H :HIERARCHIE DE ROLE**

La famille SH



La famille SH

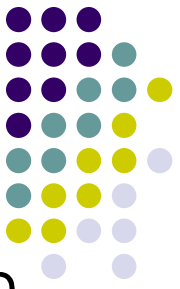
Restriction de cardinalité qualifiée (SHQ)



- Les deux constructeurs pour la restriction de cardinalité vus précédemment ne permettent pas de spécifier la classe concernée
- Pour cela, il faut plutôt avoir accès à un constructeur de restriction de cardinalité *qualifiée* (dénommée Q) :
- $I(\geq n \text{ R.C}) = \{a \in \quad | \{b | (a, b) \in I(R) \wedge b \in I(C)\} | \geq n\}$
- $I(\leq n \text{ R.C}) = \{a \in \quad | \{b | (a, b) \in I(R) \wedge b \in I(C)\} | \leq n\}$

La famille SH

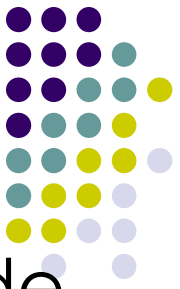
Constructeur d'inversion (SHI)



- Pour spécifier qu'un rôle R1 est l'inverse d'un deuxième R2
- $R1(x,y) \rightarrow R2(y,x)$
- Exemple :
- *ParentDe* , *enfantDE*
- *Regarder* , *regardéPar*
- sémantique du constructeur d'inversion est définie de la manière suivante :
- $I(R-) = \{(y, x) \mid (x, y) \in I(R)\}$
- La déclaration se fait : *estRegardéPar* $\equiv$ *regarde-*

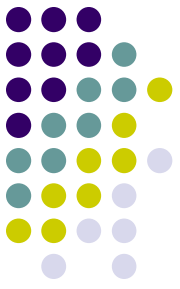
La famille SH

l'énumération (SHO)



- Le constructeur d'énumération(O) permet de désigner des individus sans classe dans la terminologie en énumérant explicitement la liste des individu de cette classe.
- $\{a_1, \dots, a_n\}$ où $a_1, \dots, a_n$ sont des noms d'individus
- Selon la sémantique suivante:
- $I(\{a_1, \dots, a_n\}) = \{I(a_1), \dots, I(a_n)\}$
- *Exemple :*
- `MembrePermanentConseilSécurité` $\equiv$ `{USA,FRANCE,RUSSIE,UK,CHINE}`

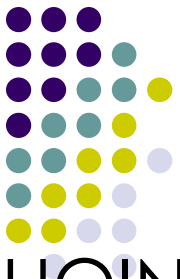
Remarques



- **équivalences logiques dans les langages :**

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

Conclusion



- Différents types :AL ,ALC ,ALCNSH ,SHIF,SHOIN
- *Permet de représenter des ontologies de différentes complexités selon le langage utilisé*
- OWL :SHIF
- OWL DL:SHOIN (D) (D pour les propriétés datatype)