Représentation de connaissances

II – Règles ontologiques

et interrogation de bases de connaissances

Marie-Laure Mugnier

INTERPRÉTATIONS

- **Vocabulaire:** $\mathcal{V} = (\mathcal{P}, C)$, où $\mathcal{P}$ = ensemble fini de prédicats
  C = ensemble de constantes (peut être infini)

  ⇒ les formules sont construites sur ce vocabulaire

  ⇒ un **atome** est de la forme $p(e_1, ..., e_k)$
  où $p \in \mathcal{P}$ et chaque e_i (terme) est une constante de C ou une variable

- **Interprétation de $\mathcal{V}$:** $\mathcal{I} = (D_\mathcal{I}, .^\mathcal{I})$, où

  $D_\mathcal{I} \neq \emptyset$ (le domaine de l'interprétation)
  pour tout $c \in C$, $c^\mathcal{I} \in D_\mathcal{I}$
  pour tout $p \in \mathcal{P}$ d'arité k, $p^\mathcal{I} \subseteq D_\mathcal{I}^k$

$\mathcal{V} = (\{p_{/2}, r_{/3}\}, \{a, b\})$

$\mathcal{I}$: $D_\mathcal{I} = \{d_1, d_2, d_3\}$

$a^\mathcal{I} = d_1$, $b^\mathcal{I} = d_2$
$p^\mathcal{I} = \{(d_2, d_1), (d_2, d_3), (d_3, d_2)\}$
$r^\mathcal{I} = \{(d_3, d_3, d_1)\}$

HYPOTHÈSE SIMPLIFICATRICE SUR LES INTERPRÉTATIONS

On va adopter une hypothèse couramment faite qui simplifiera nos notations :

- **hypothèse du nom unique** (Unique Name Assumption) :
  deux constantes différentes désignent forcément des objets différents

⇒ dans toute interprétation, deux constantes différentes s'interprètent par deux éléments de domaine différents

⇒ on peut donc simplier les notations en appelant par le même nom une constante et l'élément du domaine qui l'interprète
(« toute constante s'interprète par elle-même »)

$\mathcal{V}$ = ({$p_{/2}$, $r_{/3}$ }, {a, b})

$\mathcal{I}$: $D_I = \{d_1, d_2, d_3\}$
$a^I = d_1$, $b^I = d_2$
$p^I = \{ (d_2, d_1), (d_2, d_3), (d_3, d_2) \}$
$r^I = \{ (d_3, d_3, d_1) \}$

$D_I = \{a, b, d_3\}$
$p^I = \{ (b, a), (b, d_3), (d_3, b) \}$
$r^I = \{ (d_3, d_3, a) \}$

INTERPRETATIONS (AVEC HYPOTHÈSE UNA)

- **Vocabulaire:** $\mathcal{V} = (\mathcal{P}, C)$, où

  $\mathcal{P}$ = ensemble fini de prédicats

  C = ensemble de constantes (peut être infini)

- **Interprétation de $\mathcal{V}$:** $\mathcal{I} = (D_\mathcal{I}, .^\mathcal{I})$, où

  $D_\mathcal{I} \neq \emptyset$ (le domaine de l'interprétation)

  $C \subseteq D_\mathcal{I}$ (et pour tout $c \in C$, $c^\mathcal{I} = c$)

  pour tout $p \in \mathcal{P}$ d'arité k, $p^\mathcal{I} \subseteq D_\mathcal{I}^k$

- $\mathcal{I}$ est un **modèle** d'une formule close f (construite sur $\mathcal{V}$) si f est vraie pour $\mathcal{I}$

$\mathcal{V} = (\{p_{/2}, r_{/3}\}, \{a, b\})$

$\mathcal{I}$: $D_\mathcal{I} = \{a, b, d_3\}$

$p^\mathcal{I} = \{(b, a), (b, d_3), (d_3, b)\}$

$r^\mathcal{I} = \{(d_3, d_3, a)\}$

$f_1 = \exists x \exists y\ (p(b,x) \wedge r(x,x,y))$

$f_2 = p(a,b) \wedge p(b,a)$

$f_3 = \exists x\ p(x,y)$

On n'interprétera que des formules closes

Homomorphisme et Conséquence logique

Etant données deux formules *f* et *g*,

f $\models$ ***g*** (*g* est conséquence de *f*)

signifie que

tout modèle de *f* est un modèle de *g*

(« dans toute situation où *f* est vraie, *g* est forcément vraie aussi »)

Base de faits *F*

CQ booléenne q()

F $\models$ *q()* **ssi** il existe un homomorphisme de q dans *F*

Pourquoi ?

Modèles d'une base de faits (sans variables)

F = {p(a,b), p(b,c),q(c) }

Si une interprétation *I* est un modèle de *F*, que contient-elle *forcément* ?

p^I contient forcément (a,b) et (b,c)
q^I contient forcément c

Qu'y a-t-il de commun à *tous* les modèles de *F* ?

p^I = { (a,b), (b,c) }
q^I = { c }

Un **plus petit modèle** d'une formule f est un modèle de f qui n'est plus un modèle si on enlève un élément de l'interprétation d'un prédicat

Une base de faits a un **unique plus petit modèle**

$\mathcal{I}$: $D_{\mathcal{I}}$ = {a,b,c, ...}
p^I = { (a,b), (b,c) }
q^I = { c }

Remarque : on doit interpréter toutes les constantes du vocabulaire même si elles n'apparaissent pas dans F

Modèle canonique d'une base de faits (sans variables)

Vocabulaire $\mathcal{V} = (\mathcal{P}, C)$
Base de faits F sur $\mathcal{V}$

Modèle **canonique** de F

$\mathcal{M}$: $D^{\mathcal{M}} = C$
pour tout $p \in \mathcal{P}$ d'arité k, $p^{\mathcal{M}} = \{ (c_1, ..., c_k) \mid p(c_1, ..., c_k) \in F \}$

Le modèle canonique de F correspond à l'**intersection** de tous les modèles de F

$\mathcal{V} = (\{r_{/3}, p_{/2}, q_{/1} \}, \{a, b, c, d, e\})$

$F = \{ p(a,b), p(b,c), q(c) \}$

$\mathcal{M}$: $D_{\mathcal{M}} = \{a,b,c, d ,e\}$
$p^{\mathcal{M}} = \{ (a,b), (b,c) \}$
$q^{\mathcal{M}} = \{ c \}$
$r^{\mathcal{M}} = \emptyset$

CES FORMULES ONT-ELLES UN PLUS PETIT MODÈLE (UNIQUE) ?

$f = p(a) \vee p(b)$

$f = p(a) \rightarrow p(b)$

$\equiv$

$\neg p(a) \vee p(b)$

$f = p(a) \wedge (p(a) \rightarrow p(b))$

$f = p(a) \rightarrow \neg p(a)$

$f = p(a) \wedge \neg p(a)$

QU'EST-CE Q'UN MODÈLE D'UNE CQ BOOLÉENNE ?

$q() = \exists x \exists y \exists z (p(x,y) \wedge p(y,z) \wedge r(x,z,a))$

$\mathcal{I}$: $D_{\mathcal{I}} = \{a,b,c\}$
$p^{\mathcal{I}} = \{ (a,b), (b,c) \}$
$r^{\mathcal{I}} = \{ (a,b,c), (b,c,a) \}$

Une interprétation *I* est un modèle de *q* si :

il existe une application *f* des termes de *q()* dans D_I telle que :

1. $f(c) = c$ pour toute constante *c*

2. pour tout atome $p(e_1,...,e_k)$ de *q*, on a $(f(e_1), ... f(e_k)) \in p^I$

Homomorphisme et Conséquence logique

Base de faits *F*
CQ booléenne q()
$F \models q()$ **ssi** il existe un homomorphisme de *q* dans *F*

Pourquoi ?

- (⇒) Supposons que $F \models q$, c'est-à-dire « tout modèle de *F* est un modèle de *q* »
  Prenons en particulier le modèle de canonique de *F* (soit *M*)
  M est un modèle de *q*
  Il existe donc une application *f* des termes de *q* dans D_M
  telle que :
  1. $f(c) = c$ pour toute constante *c*
  2. pour tout atome $p(e_1,...,e_k)$ de *q*, $(f(e_1), ... f(e_k)) \in p^M$

  f définit un homomorphisme de *q* dans *F*

- (⇐) Soit *h* un homomorphisme de *q* dans *F*
  h montre que le modèle canonique de *F* est un modèle de *q*
  donc tout modèle de *F* est un modèle de *q*
  c'est-à-dire $F \models q$

MODÈLES D'UNE KB (BASE DE FAITS, RÈGLES DATALOG)

K = (F, $\mathcal{R}$) est vue d'un point de vue logique comme la conjonction de F et de toutes les règles de $\mathcal{R}$

donc : un modèle de K est un modèle de chaque fait de F et chaque règle de $\mathcal{R}$

K = (F, $\mathcal{R}$)
F = {p(a,b), p(b,c) }
$\mathcal{R}$ = {R_1,R_2} avec R_1 : p(x,y) → q(y)
R_2 : q(x), p(x,y) → r(y) }

I est un modèle de K **ssi** :

- I modèle de F : (a,b) ∈ p^I et (b,c) ∈ p^I
- I modèle de R_1 : pour tout couple (d_1,d_2) ∈ p^I, on a d_2 ∈ q^I
- I modèle de R_2 : pour tout d_1 ∈ q^I et (d_1,d_2) ∈ p^I, on a d_2 ∈ r^I

EXEMPLE

K = (F, $\mathcal{R}$)

F = {p(a,b), p(b,c) }

$\mathcal{R}$ = {R_1,R_2} avec R_1 : p(x,y) → q(y)

R_2 : q(x), p(x,y) → r(y) }

I = (D, .I) avec D = { a, b, c, e } Rappel : *a, b et c désignent en fait a^I, b^I et c^I*

p^I = { (a,b), (b,c), (e,c) }

q^I = { b, c }

r^I= { a }

I est-elle un modèle de K ?

I = (D, .I) avec D = { a, b, c, e }

p^I = { (a,b), (b,c) }

q^I = { b, c }

r^I= { a,c }

I est-elle un modèle de K ?

I = (D, .I) avec D = { a, b, c, e } p^I = { (a,b),(b,c) }

q^I = { b, c }

r^I= { c }

I est-elle un modèle de K ?

PROPRIÉTÉ DU PLUS PETIT MODÈLE UNIQUE

Toute base de connaissances K =(F,$\mathcal{R}$) où $\mathcal{R}$ est un ensemble de règles Datalog possède un unique plus petit modèle M:

pour tout modèle I de K, pour tout prédicat p, on a $p^M \subseteq p^I$

F = {p(a,b), p(b,c) }
$\mathcal{R}$ = {R_1,R_2} avec R_1 : p(x,y) → q(y)
R_2 : q(x), p(x,y) → r(y) }

Quel est son plus petit modèle ?

Etant donnée une CQ booléenne q, pour déterminer si K ⊨ q il suffit donc de vérifier si le plus petit modèle de K est un modèle de q :

- si oui, tout modèle de K contient ce modèle, c'est donc un modèle de q
- si non, on a un modèle de K qui n'est pas un modèle de q

How to actually compute the answers to a query on a KB?

Forward chaining : starting from F, we iteratively compute all the facts that are consequences of the current factbase and the rules.

F = { fundedBy(Bob,C), Company(C) }
R = ∀x∀y (fundedBy(x,y) → relatedTo(x,y))
F,R ⊨ relatedTo(Bob,C)

A rule *R: B → H* is applicable to a factbase *F* if there is a homomorphism *h* from *B* to *F*

Applying *R* to *F* according to *h* consists of adding *h(H)* to *F*

h : body(R) → F
x ↦ Bob
y ↦ C

EXEMPLE (PISTES CYCLABLES)

F

Direct(A,B)

Direct(B,C)

Direct(C,D)

Direct(D,B)

$\mathcal{R}$

R1 : **Direct(x,y) → Chemin(x,y)**

R2 : Direct(x,y) ∧ Chemin(y,z) → Chemin(x,z)

PROPERTIES OF DATALOG RULES

- $K = (F, \mathcal{R})$ where F is a set of (ground) facts
  $\mathcal{R}$ is a set of Datalog rules

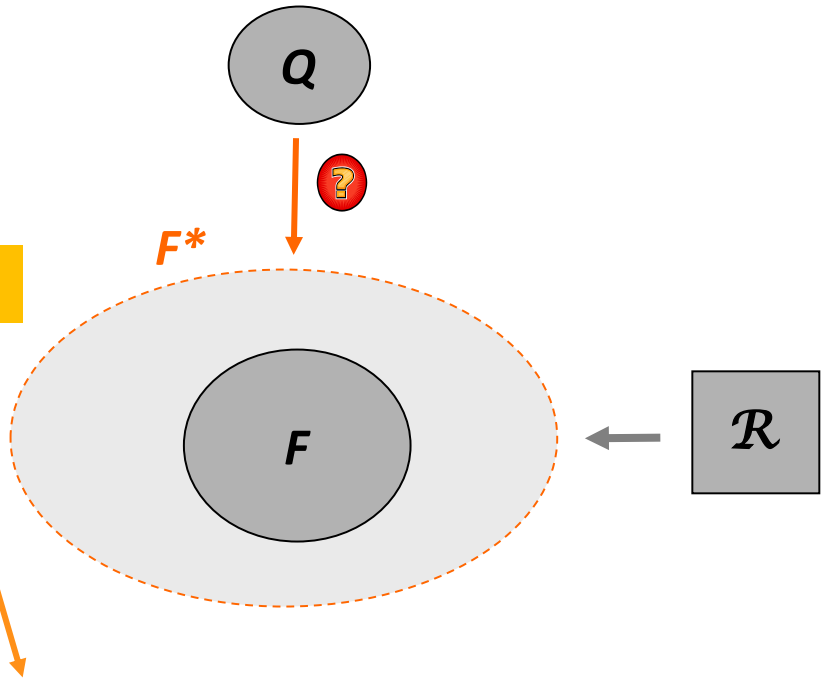
By applying rules from $\mathcal{R}$ starting from F, a unique result is obtained:
the **saturation** of F by $\mathcal{R}$ (denoted here by ***F****)

F^* is **finite** since no new variable is created

F^* allows to compute the **answers** to a CQ on K:

$(a_1, \ldots, a_k)$ is an answer to $q(x_1,\ldots,x_k)$ on K iff there is a homomorphism from q to K that maps each x_i to a_i

If k=0: () is an answer means « yes »



Why? Because the canonical model of the saturated factbase F* is the **unique smallest model** of K

EXEMPLE (PISTES CYCLABLES)

F

Direct(A,B)
Direct(B,C)
Direct(C,D)
Direct(D,B)

$\mathcal{R}$

R1 : **Direct(x,y) → Chemin(x,y)**

R2 : Direct(x,y) ∧ Chemin(y,z) → Chemin(x,z)

F* = F ∪ { Ch(A,B), Ch(B,C), Ch(C,D), Ch(D,B)
Ch(A,C), Ch(B,D), Ch(C,B), Ch(D,C)
Ch(A,D), Ch(B,B), Ch(C,C), Ch(D,D)
}

Q(x) = Chemin(A,x) ∧ Chemin(x,D) « trouver tous les x qui sont sur un chemin de A à D »

Pour toute constante c, (c) est une réponse à Q(x) sur K si
K ⊨ Chemin(A,c) ∧ Chemin(c,D)

On cherche les homomorphismes de Q dans F* $x \mapsto B$ $x \mapsto C$ $x \mapsto D$

Q(F*) = { (B), (C), (D) }

Cadre étudié dans ce cours

- **Base de connaissances (KB)** composée :
  - d'une **base de faits**

    (qu'on peut voir comme une base de données relationnelle)
  - d'une **base de règles** positives et conjonctives

    (Datalog)
- **Requêtes conjonctives**

  (correspondant à des requêtes de base en SQL / SPARQl)
- **Problème fondamental : interrogation de la KB**

  (calculer toutes les réponses à une requête conjonctive sur la KB)

Extensions

- **Contraintes négatives**
- (on évoquera les règles existentielles qui généralisent Datalog)
- **Mappings** pour extraire une partie d'une base de données relationnelle et la traduire en une base de faits

SI ON AJOUTE DES CONTRAINTES NÉGATIVES

- Une **contrainte négative** est de la forme

$$\forall X \, (\text{Condition}[X] \rightarrow \bot)$$

où *Condition* est une conjonction d'atomes et $\bot$ le symbole absurde

$$\forall x \, (\text{Film}(x) \wedge \text{Personne}(x) \rightarrow \bot)$$

- Une base de faits *F* satisfait une **contrainte négative** *C* s'il n'y a pas d'homomorphisme de la condition de *C* dans *F*
  *(*autrement dit, *C* vue comme une règle n'est pas applicable)

  Remarque : $F \cup \{C\}$ est consistante (satisfiable) **ssi** *F* satisfait *C*

- Une base de connaissances $K = (F, \mathcal{R}, \mathcal{C})$
  où $\mathcal{C}$ est un ensemble de contraintes négatives est consistante (satisfiable) ssi F^* (la saturation de *F* par $\mathcal{R}$) satisfait toutes les contraintes de $\mathcal{C}$

EXERCICE (APPLICATION DIRECTE DU COURS)

Soit la KB $\mathcal{K} = (F, \mathcal{R}, C)$

F = { r(a,b), r(b,c), r(c,a) }

$\mathcal{R}$ = {r(x,y) → s(x,y) ; s(x,y) ∧ s(y,z) →s(x,z) }

C = {s(x,y) ∧ s(y,x) → ⊥}

- L'interprétation I :

  D = {a,b,c, d,e}

  r^I = {(a,b), (b,c), (c,a), (d,e)}

  s^I= D x D

  est-elle un modèle de $(F,\mathcal{R})$?
- Quel est le plus petit modèle de $(F,\mathcal{R})$?
- F satisfait-elle C ? $\mathcal{K}$ satisfait-elle C ?
- $\mathcal{K}$ est-elle consistante (satisfiable) ?
- Soit q() = ∃x lapin(x). $\mathcal{K}$ répond-t-elle oui à q?

INTERROGATION DE KBS AVEC CONTRAINTES NÉGATIVES

Soit une base de connaissances $K = (F, \mathcal{R}, C)$

1. *K* est-elle satisfiable ?

   On calcule F^* la saturation de F par $\mathcal{R}$
   Puis on teste si F^* satisfait C.

2. Interrogation de *K*

   Si K n'est pas satisfiable, le problème d'interrogation « trivialise ».

   Sinon, les réponses à une CQ *q* sont données par
   les homomorphismes de *q* dans F^*.

ALGORITHME DE CHAÎNAGE AVANT NAÏF

Algorithme ForwardChaining (K) *// Données : K = (F, $\mathcal{R}$)*
Début *// Résultat : F* (F saturée par $\mathcal{R}$)*

Fin ← faux

i ← 0 *// numéro d'étape*
F(0) ← F // base de faits à l'étape 0
Tant que non fin
    i ← i + 1
    nouvFaits(i) ← ∅ *// ensemble des nouveaux faits obtenus à l'étape i*
    Pour toute règle R : B → H ∈ $\mathcal{R}$
        Pour tout homomorphisme h de B dans F(i-1)
            Si h(H) ∉ (F(i-1) ∪ nouvFaits(i))
                Ajouter h(H) à nouvFaits(i)
    Si nouvFaits(i) = ∅, Fin ← vrai
    Sinon F(i) ← F(i-1) ∪ nouvFaits(i)

Retourner F(i)
Fin

| F*| dans le pire des cas ?

#atomes ayant un prédicat p d'arité k :
au plus [#constantes(K)]^k

=> | F*| **exponentielle** en
l'arité maximale des prédicats

ALGORITHME DE CHAÎNAGE AVANT NAÏF

Algorithme ForwardChaining (K) *// Données : K = (F, $\mathcal{R}$)*
Début *// Résultat : F* (F saturée par $\mathcal{R}$)*

Fin ← faux

i ← 0 *// numéro d'étape*
F(0) ← F // base de faits à l'étape 0
Tant que non fin
    i ← i + 1
    nouvFaits(i) ← ∅ *// ensemble des nouveaux faits obtenus à l'étape i*
    Pour toute règle R : B → H ∈ $\mathcal{R}$
        Pour tout homomorphisme h de B dans F(i-1)
            Si h(H) ∉ (F(i-1) ∪ nouvFaits(i))
                Ajouter h(H) à nouvFaits(i)
    Si nouvFaits(i) = ∅, Fin ← vrai
    Sinon F(i) ← F(i-1) ∪ nouvFaits(i)

Retourner F(i)
Fin

AMÉLIORATIONS DE L'ALGORITHME NAÏF DE CHAINAGE AVANT

- Test d'homomorphisme :
  étant donnés B et F, déterminer s'il existe un homomorphisme de B dans F est un problème NP-complet

  Mais B est petit ...

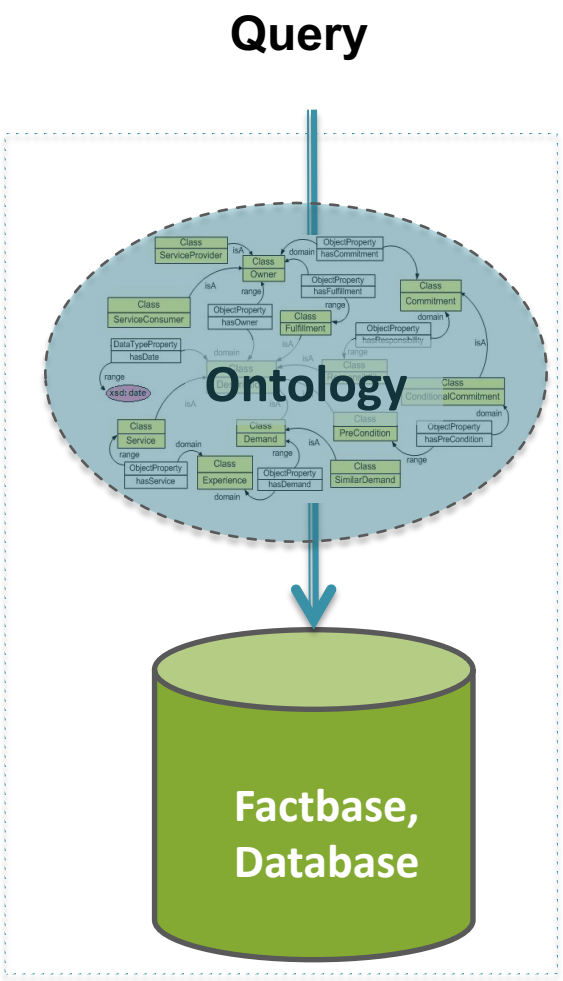
- Eviter de recalculer **tous** les homomorphismes de B dans F à chaque étape

  ⇒ ne rechercher que les homomorphismes **nouveaux**

Un homomorphisme h est nouveau à l'étape i s'il envoie au moins un atome de B dans un fait produit à l'étape i-1
Autrement dit : $h(B) \not\subseteq F(i-2)$
ou encore : $h(B) \cap \text{nouveauxFaits}(i-1) \neq \emptyset$

- Ne pas considérer toutes les règles à chaque étape, mais seulement celles pour lesquelles il y a potentiellement un nouvel homomorphisme de B (voir exercice)

Ontology-Mediated Query Answering



Compute answers to queries

while taking into account inferences

enabled by an ontology

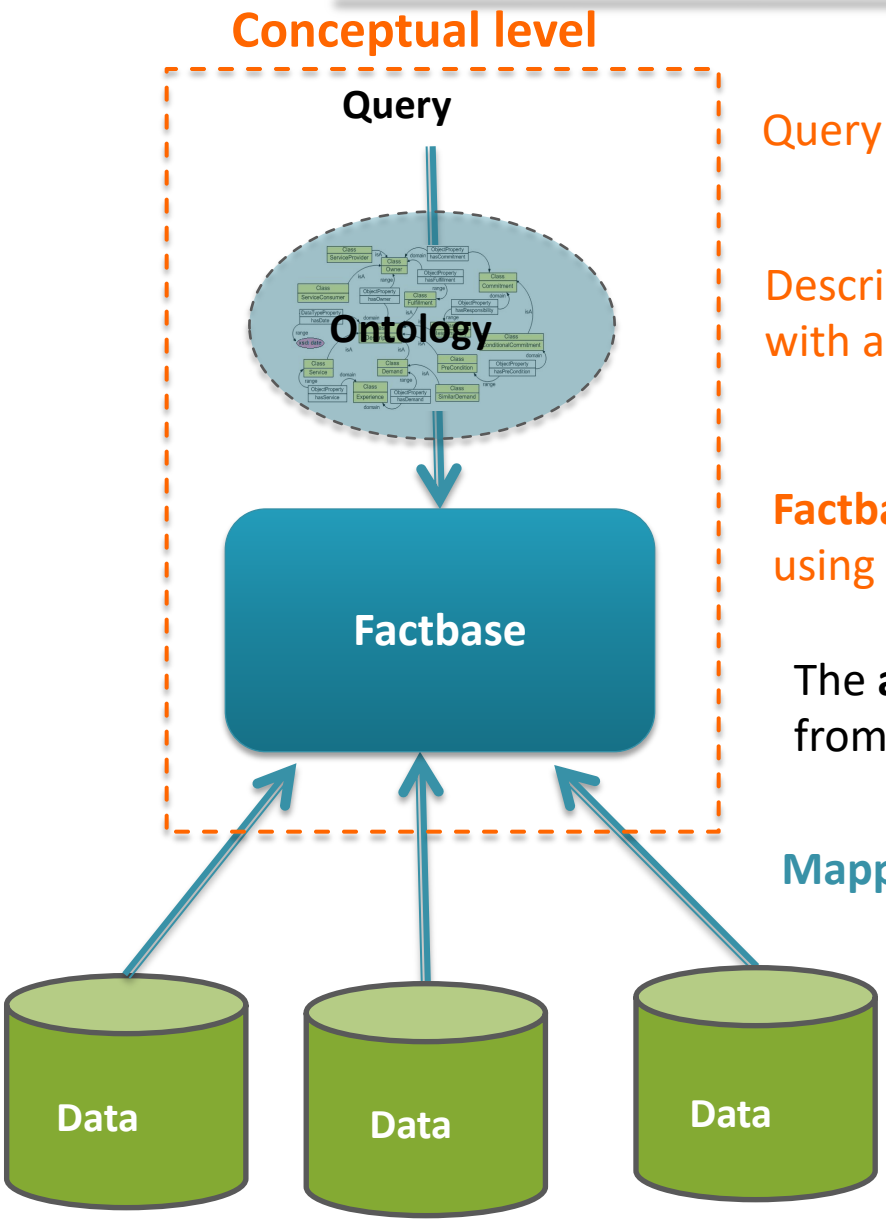
Limitation up to now:

facts and data expressed on the same vocabulary

(or on similar vocabularies, cf. the natural translation from a relational database to a factbase)

A more General Schema

« Ontology-Based **Data Access** »
[Poggi et al., JoDS, 2008]



Query using the vocabulary of the ontology

Description of the application domain with a high abstraction level

Factbase (possibly virtual)
using the vocabulary of the ontology

The **answers** to the query are **inferred** from the knowledge base

Mappings from data to facts

{ Database query ⇝ Facts }

Independent and heterogeneous data sources

MAPPINGS

Patient_T [ID_PATIENT, NAME,SSN]

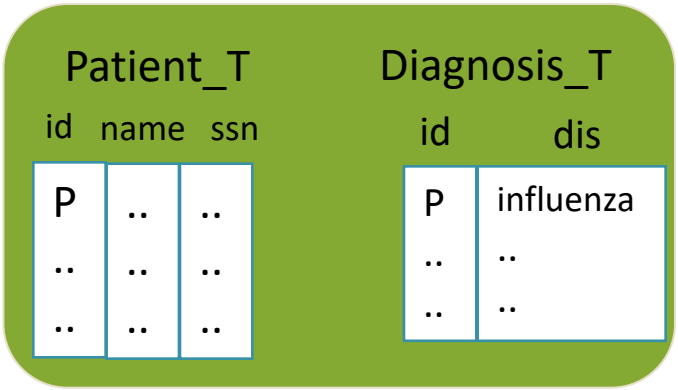
Diagnosis_T[ID_PATIENT, DISORDER]

Patient /1
Diagnosis / 2
Influenza /1

Mapping = database query(X) ⇝ conjunction with free variables X

q(x): ∃n∃s Patient_T (x,n,s) ⇝ Patient(x)

q'(x): ∃n∃s Patient_T (x,n,s) ∧ Diagnostic_T(x,y) ∧ y = « influenza »
⇝ ∃z (diagnosis(x,z) ∧ Influenza(z))



MAPPINGS CAN BE SEEN AS RULES

Patient_T [ID_PATIENT, NAME,SSN]

Diagnosis_T[ID_PATIENT, DISORDER]

q(x): ∃n∃s Patient_T (x,n,s) ⇝ Patient(x) **GAV (Global-As-View)**

q'(x): ∃n∃s Patient_T (x,n,s) ∧ Diagnosis_T(x,y) ∧ y = « influenza »
⇝ ∃z diagnosis(x,z) ∧ Influenza(z)

Patient_T (x,n,s) → Patient(x) Datalog Rule

Patient_T (x,n,s), Diagnosis_T(x, « influenza ») →∃z diagnosis(x,z), Influenza(z)
Existential Rule

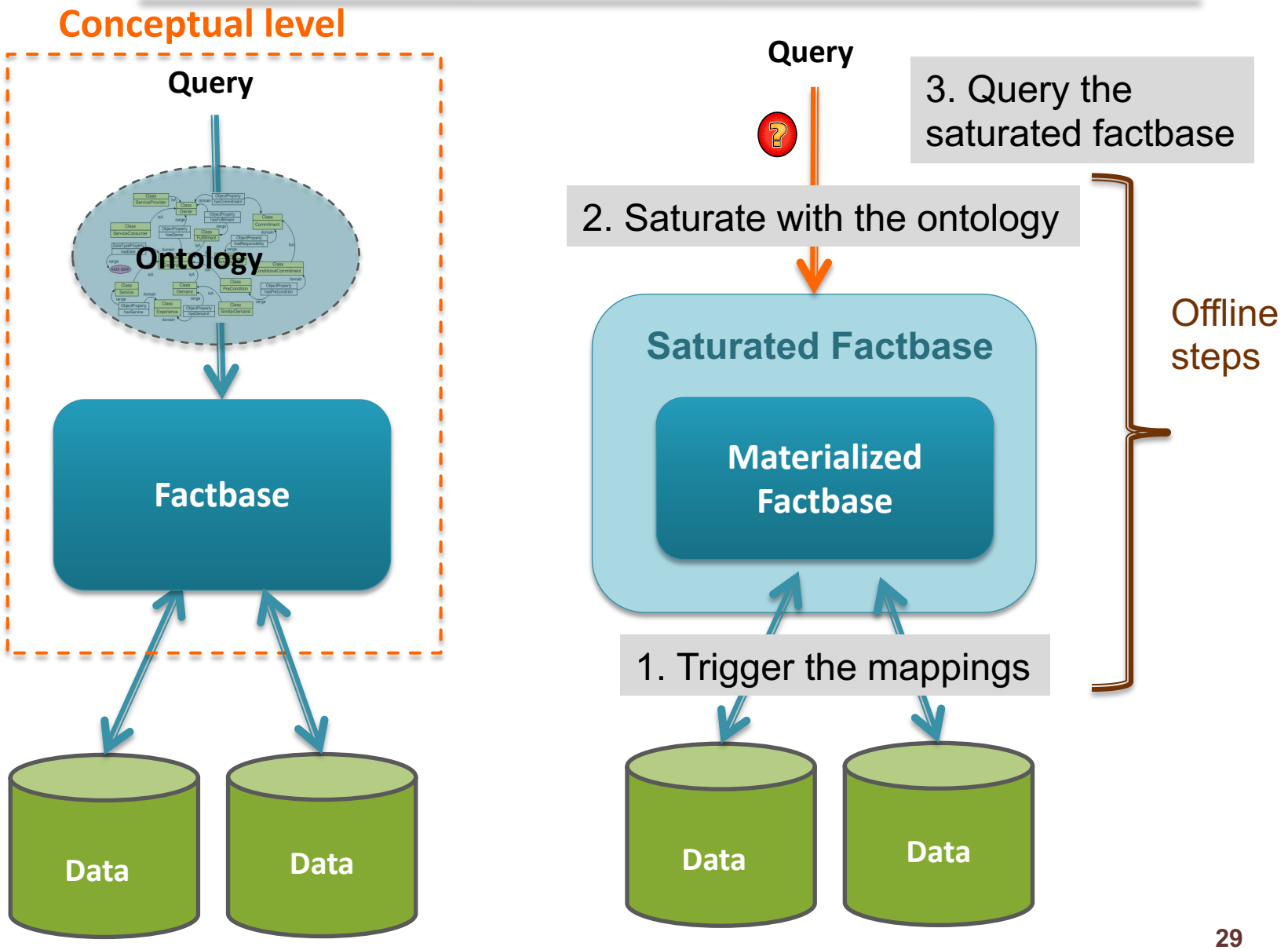
More generally: $\mathbf{q_1(X) \rightsquigarrow q_2(X)}$ where q_1 is expressed in a native query language

Decomposition of a mapping into 2 mappings

low level : $q_1(X) \rightsquigarrow view(X)$ Result of the query stored in a view

high level : $\mathbf{view(X) \rightarrow q_2(X)}$ Logical rule

OBDA : TOTAL MATERIALIZATION (FORWARD CHAINING)



CADRE ÉTUDIÉ DANS CE COURS

- **Base de connaissances (KB)** composée :
  - d'une **base de faits**
    (qu'on peut voir comme une base de données relationnelle)
  - d'une **base de règles** positives et conjonctives
    (Datalog)
- **Requêtes conjonctives**
  (correspondant à des requêtes de base en SQL / SPARQl)
- **Problème fondamental : interrogation de la KB**
  (calculer toutes les réponses à une requête conjonctive sur la KB)

Prochain cours : autre approche algorithmique pour l'interrogation

Extensions

- **Contraintes négatives**
- (on évoquera les règles existentielles qui généralisent Datalog)
- **Mappings** pour extraire une partie d'une base de données relationnelle et la traduire en une base de faits