Différentes formes de négation

Théorie des bases de connaissances
HAI933I

Contenu du cours

- Interrogation de bases de faits
  - Rappels : hypothèses du monde ouvert et du monde clos
  - Bases de faits et requêtes avec négation atomique
  - Traitement des requêtes selon les deux visions
- Interrogation de bases de connaissances
  (on ajoute des règles)

Hypothèse du monde ouvert (OWA)

OWA : open-world assumption

- On suppose une connaissance **incomplète** du monde

  → ce qui est dans la base de faits est vrai, mais pour le reste on ne sait pas

- Pour déterminer les réponses à une requête « de façon certaine », on considère toutes les extensions possibles de la base de faits

  → cela correspond à considérer **tous les modèles** de la base de faits

  donc la conséquence logique classique

- C'est l'hypothèse généralement faite dans les systèmes à base de connaissances (y compris dans le web sémantique)

  ex : arbre généalogique, liste des témoins d'un accident,
  liste des fans d'une star, ...

Hypothèse du monde clos (CWA)

CWA : closed-world assumption

On suppose qu'on a une connaissance complète du monde

→ la base de faits contient tous les atomes (instanciés) vrais, les autres sont faux

Pour déterminer les réponses à une requête,
il suffit de considérer l'unique modèle associé à la base de faits
[c'est-à-dire le modèle isomorphe à la base de faits]

Rappel : une interprétation correspond à un monde « complètement connu »

- C'est l'hypothèse faite en bases de données

  Tout ce qui est stocké est vrai, le reste est faux
  ex : liste des utilisateurs enregistrés, plan du métro, ...

Cela fait-il une différence dans le cadre vu jusqu'à présent ?

[base de faits positive et (U)CQ]

Non

car, pour une BCQ Q,

tout modèle de *F* est un modèle de Q

ssi

le modèle isomorphe à *F* est un modèle de *Q*

Mais ajoutons aux requêtes (et aux faits) des littéraux négatifs ...

Rappel : un **littéral** est un atome ou la négation d'un atome

Négation atomique dans les requêtes et les faits

- **CQ¬** : conjonction de littéraux quantifiée existentiellement

$$Q(x) = \exists y\ (p(x,y) \land \neg p(y,x))$$

Notation:

Q+ est l'ensemble des littéraux positifs de Q

Q- est l'ensemble des atomes niés dans les littéraux négatifs

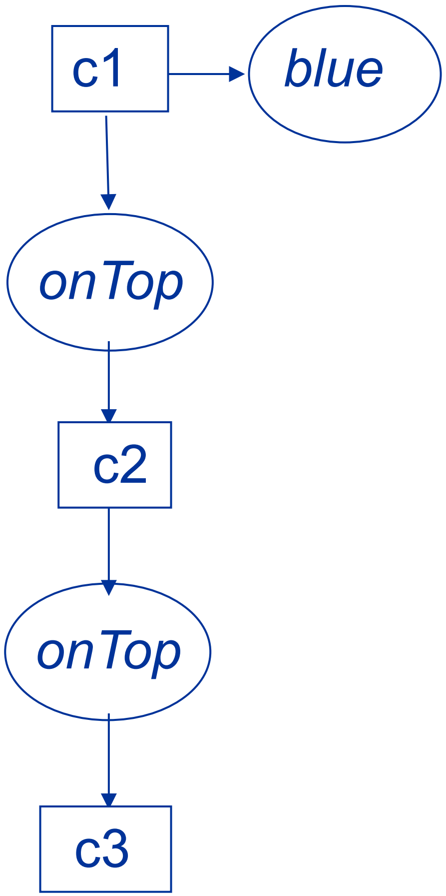
$$Q+ = \{\ p(x,y)\ \}$$

$$Q- = \{\ p(y,x)\ \}$$

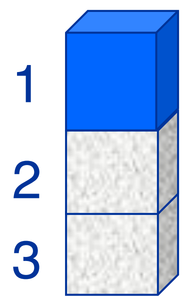
- **Base de faits** : conjonction de littéraux fermée existentiellement

Remarque : En CWA, une base de faits est forcément positive

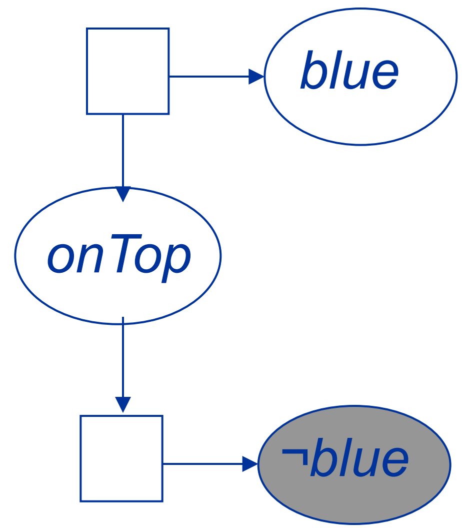
Closed World Assumption



F



Q



[Q as a Boolean CQ¬]
$\exists x\ \exists y\ (\mathrm{blue}(x) \wedge \neg\mathrm{blue}(y) \wedge \mathrm{onTop}(x,y))$?

yes { () }

[Q with answer variable x]
$\exists y\ (\mathrm{blue}(x) \wedge \neg\mathrm{blue}(y) \wedge \mathrm{onTop}(x,y))$

{ (c1) }

Évaluer une $CQ^{\neg}$ avec CWA

- On considère des requêtes sûres (safe) :

toute variable qui apparaît dans un littéral négatif apparaît aussi dans un littéral positif

$Q(x) = \exists y \neg p(x,y)$

Requête pas sûre : quelles sont les valeurs possibles pour x et y?

- Tout le domaine ? L'ensemble des réponses peut être infini
- Les constantes de F ?
  Des modifications de F sur des prédicats qui n'apparaissent pas dans Q vont avoir une influence sur Q(F)

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

- Etant donnée une base de faits (forcément) positive F et une BCQ¬ (sûre) Q:

F répond positivement à Q si le modèle isomorphe à F est un modèle de Q

Autrement dit : il existe un homomorphisme h de Q+ dans F tel que $h(Q-) \cap F = \emptyset$

Évaluer une $CQ^{\neg}$ avec OWA

- Observation 1 : une base de faits positive répond toujours non à une $BCQ^{\neg}$ qui contient au moins un littéral négatif

  L'interprétation qui « rend tout vrai » est un modèle de toute base de faits positive et ne satisfait pas une $BCQ^{\neg}$ ayant un littéral négatif

- Observation 2 : une base de faits peut être inconsistante (insatisfiable)

<u>Propriété</u> : F est inconsistante **ssi**

elle contient deux littéraux opposés $p(e_1 \ldots e_k)$ et $\neg p(e_1 \ldots e_k)$

Preuve :

($\Leftarrow$) Evident

($\Rightarrow$) Si F n'a pas de littéraux opposés,
le modèle isomorphe à (la partie positive de) F est un modèle de F
donc F est satisfiable

ÉVALUER UNE CQ$^\neg$ AVEC OWA (SUITE)

- Etant donnée une base de faits F et une BCQ¬ Q:

F répond positivement à Q si tout modèle de F est un modèle de Q ($F \models Q$)

Plus généralement :

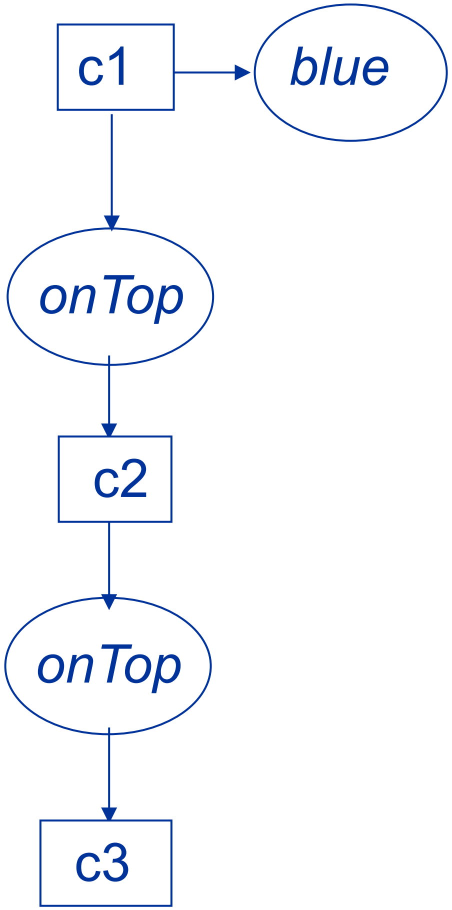
un k-uplet (a_1 , …, a_k) de constantes est une réponse (certaine)

à une CQ$^\neg$ $Q(x_1,\ldots,x_k)$ sur F
si
$F \models Q[a_1,\ldots,a_k]$

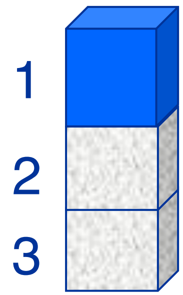
où $Q[a_1,\ldots,a_k]$ est obtenu de $Q(x_1,\ldots,x_k)$ en remplaçant chaque x_i par a_i

Donc : peut-on procéder par recherche d'homomorphismes comme dans le cas positif ?

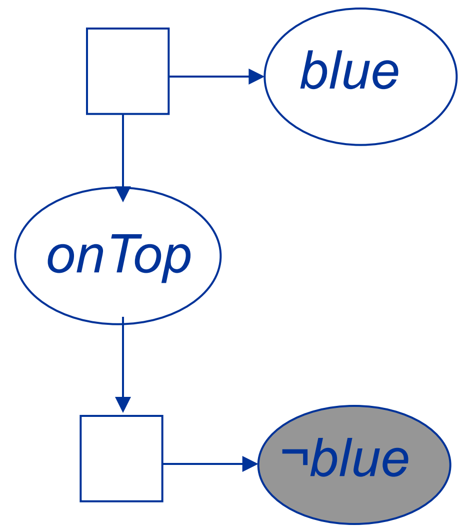
Open World Assumption (positive factbase)



F



Q



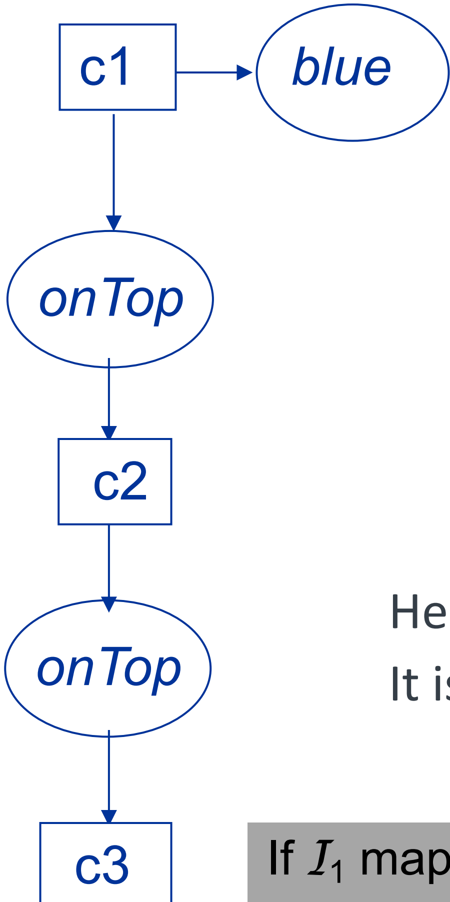
[Q as a Boolean CQ¬]
$\exists x\ \exists y\ (\mathrm{blue}(x) \wedge \neg\mathrm{blue}(y) \wedge \mathrm{onTop}(x,y))$?

no {}

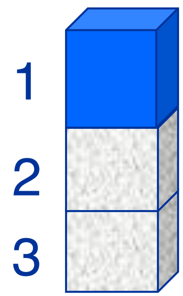
[Q with answer variable x]
$\exists y\ (\mathrm{blue}(x) \wedge \neg\mathrm{blue}(y) \wedge \mathrm{onTop}(x,y))$

{}

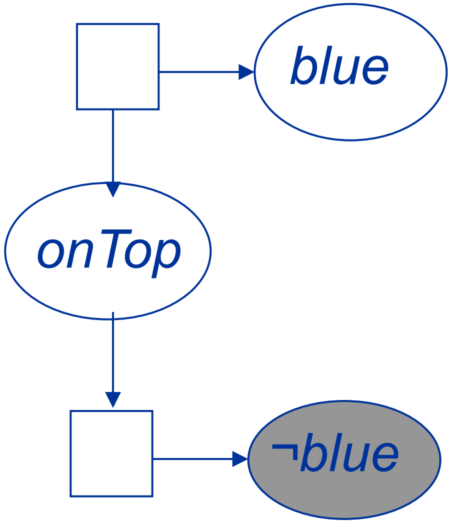
Open World Assumption (positive factbase)



F



Q



Here, the model isomorphic to *F* is a model of *Q*

It is a universal model but it is not sufficient anymore

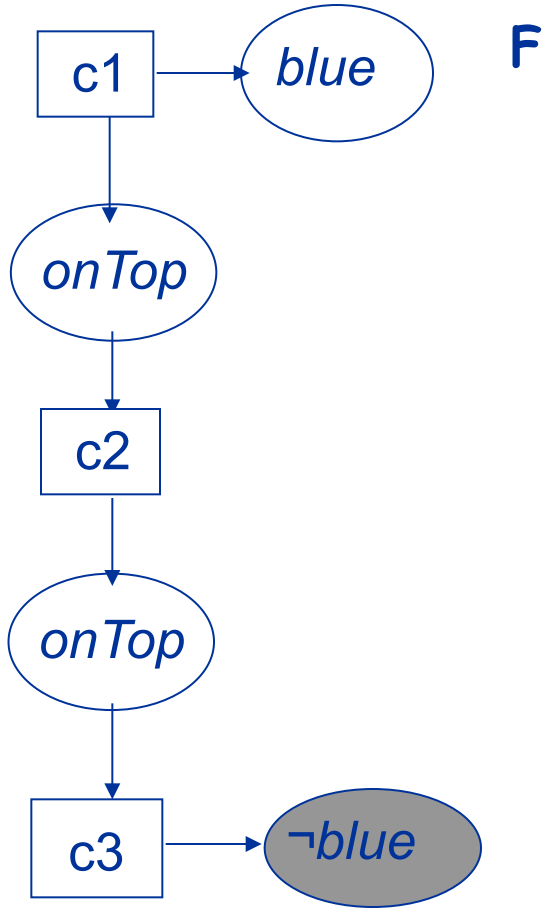
Why ?

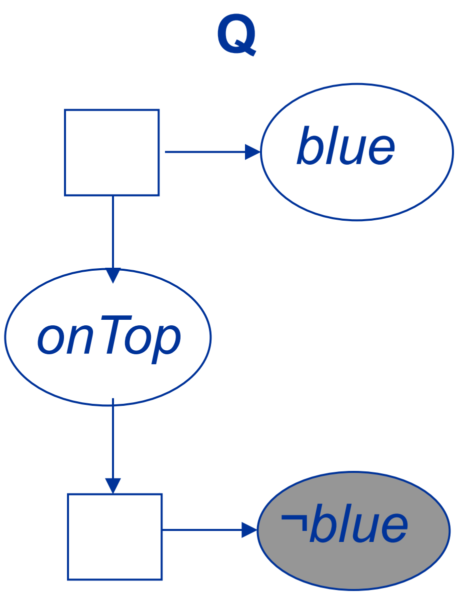
If $\mathcal{I}_1$ maps by homomorphism to $\mathcal{I}_2$ then, for any closed *f* in FOL(∃,∧),

$\mathcal{I}_1$ model of *f* $\Rightarrow$ $\mathcal{I}_2$ model of *f*

This property becomes false if *f* is a BCQ$^{\neg}$

Open World Assumption

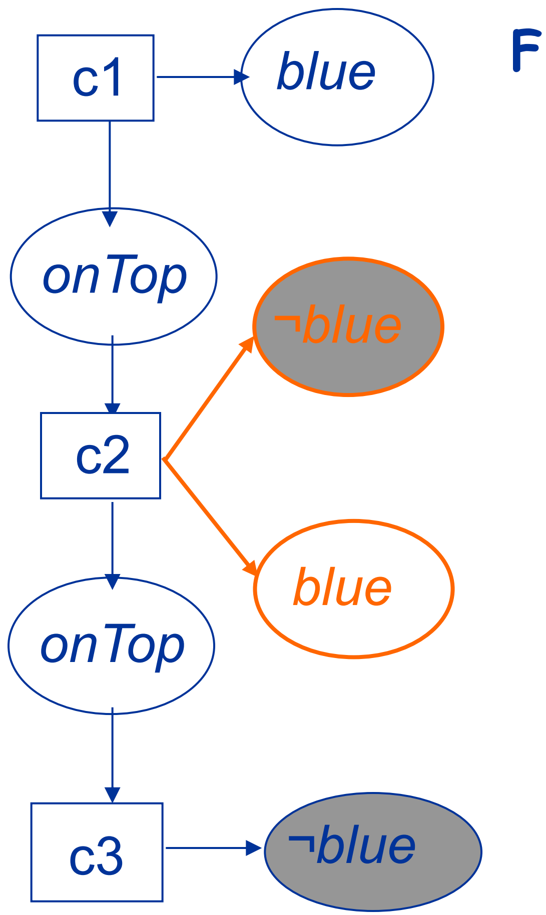




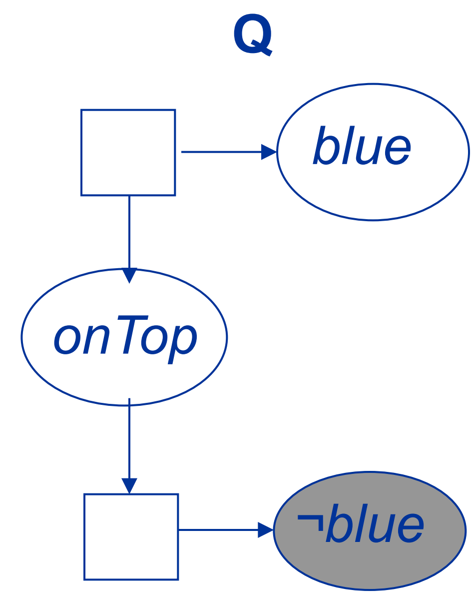
[Q as a Boolean CQ¬]
∃x ∃y (blue(x) ∧ ¬blue(y) ∧ onTop(x,y)) ?

[Q with answer variable x]
∃y (blue(x) ∧ ¬blue(y) ∧ onTop(x,y))

Where the « law of the excluded middle » comes in ...



In *any* model of *F*,
blue(c2) is either true or false



[Q as a Boolean CQ¬]
∃x ∃y (blue(x) ∧ ¬blue(y) ∧ onTop(x,y)) ?

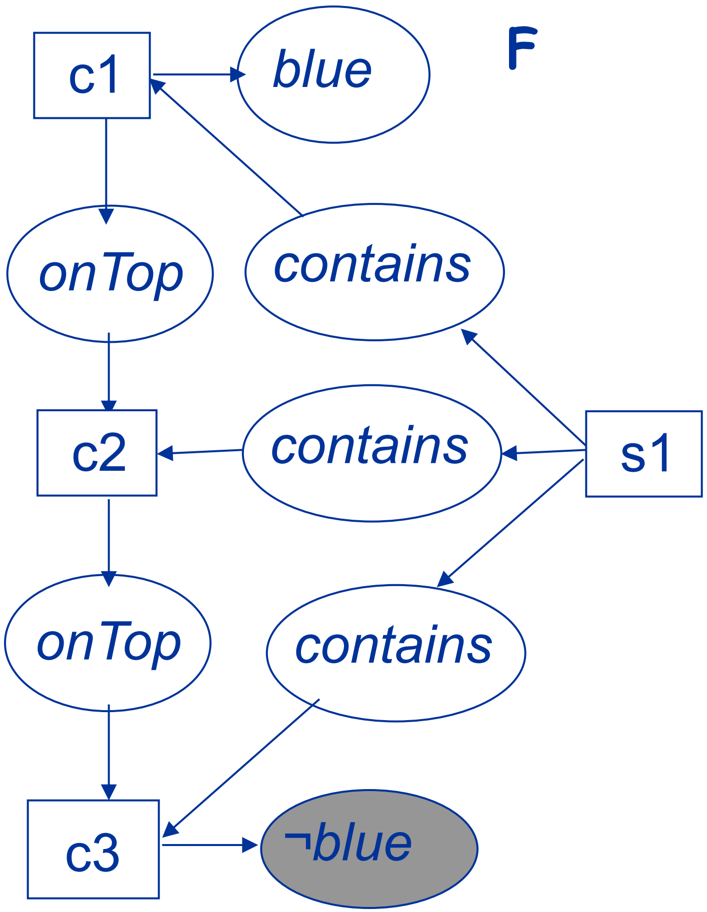
yes

[Q with answer variable x]
∃y (blue(x) ∧ ¬blue(y) ∧ onTop(x,y))

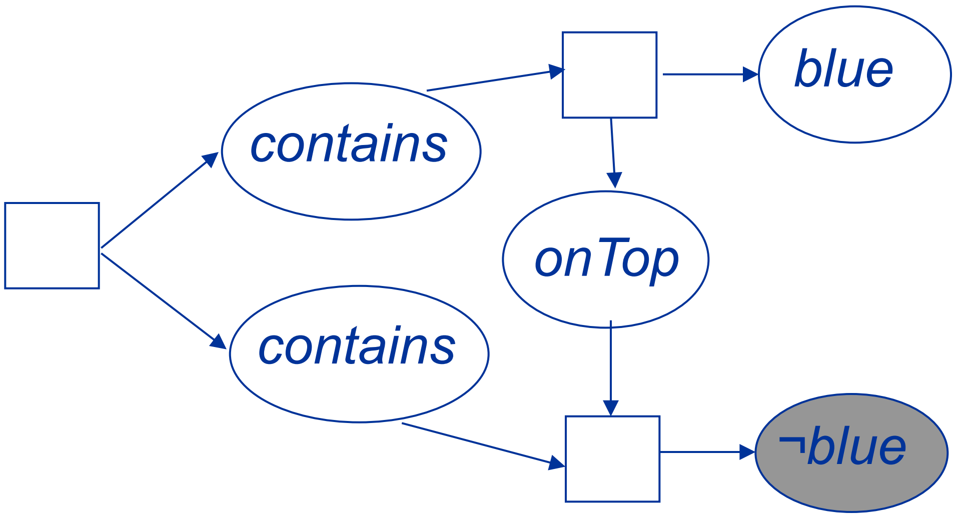
{} there is no (certain) answer

Where the « law of the excluded middle » comes in ...

F



Q



[Q with answer variable x]

$\exists y\ \exists z$ (contains(x,y) $\wedge$ contains(x,z) $\wedge$

blue(y) $\wedge$ $\neg$blue(z) $\wedge$ onTop(y,z))

Answer ?

{ (s1) }

ÉVALUER UNE CQ¬ AVEC OWA *(FIN)*

Il faut considérer toutes les façons de compléter F avec des littéraux positifs ou négatifs !

Soit F une base de faits satisfiable. Une base de faits G est une complétion de F si :

- $F \subseteq G$ et termes(G) = termes(F)
- Pour tout prédicat p d'arité k, pour tout k-uplet de termes (t1, …,tk), soit p(t1, …, tk) ∈ G, soit ¬ p(t1, …,tk) ∈ G (et pas les deux)

<u>Propriété</u> : Etant donnée une base de faits F et une BCQ¬ Q,

$F \models Q$ ssi pour toute complétion F* de F, il existe une application

h : variables(Q) → termes(F*)

telle que pour tout littéral L ∈ Q, h(L) ∈ F*

(autrement dit on étend la notion d'homomorphisme aux littéraux négatifs)

En pratique on évite de parcourir exhaustivement toutes les complétions de F

Règles avec négation : négation classique (¬)

- Déjà en logique des propositions, le mécanisme de chaînage avant n'est plus complet par rapport à la conséquence logique

  Même avec des règles positives :

  R1 : A → B

  R2 : A → C

  R3 : B ∧ C → D

  F = { ¬D }   [Question : a-t-on ¬A ?]

- En logique du premier ordre, la notion même d'application de règle devient très complexe :

  contains(x,y) ∧ contains(x,z) ∧ blue(y) ∧ ¬blue(z) ∧ onTop(y,z) → selected(x)

- Exemple de méthode complète (pour toute la logique du 1er ordre) : méthode de résolution (nécessite de mettre les formules sous forme clausale)

Règles avec négation : négation par défaut (not)

- Dans les systèmes à base de règles, on considère différentes variantes de la négation « par l'absence » :
  not P est vrai si « rien n'indique que P est vrai »
  - négation par l'échec (Prolog)
  - négation du monde clos (Datalog)
  - négation par défaut, négation stable (ASP)

- Une difficulté instrinsèque de cette négation est de définir la sémantique d'un ensemble de règles

F = { A(a) }

A(x), not B(x) → C(x)
A(x) → B(x)

A(x), not B(x) → C(x)
A(x), not C(x) → B(x)

C(a) ?

Plusieurs résultats possibles ?

Suite du cours

- Datalog avec négation (Datalog ¬)

  - Un ensemble de règles doit être stratifiable
    (donc tous les ensembles de règles ne sont pas admissibles)

  - Quand un ensemble de règles est stratifiable,
    toutes les « bonnes » dérivations donnent la même BF saturée

- Answer Set Programming (ASP), Règles existentielles avec négation

  - Tous les ensembles de règles sont admis

  - Les « bonnes » dérivations peuvent produire plusieurs BF saturées
    qui correspondent à différents « mondes possibles »
    qu'on appelle des « modèles stables » de la KB

Datalog¬ peut être vu comme un cas particulier d'ASP