TD8 : Règles positives "à la Datalog"

Dans ce TD, nous nous plaçons en logique du premier ordre et considérons les notions suivantes :

- **terme** : une variable ou une constante (pas d'autres symboles fonctionnels)
- **atome** : $p(t_1, ..., t_k)$ où p est un prédicat d'arité k et chaque t_i est un terme
- **fait** : un atome instancié (sans variable)
- **règle** (conjonctive et positive, aussi appelée "règle Datalog de base") :
  $\forall x_1 \ldots x_n$ (H → C), où :
  - H est une conjonction d'atomes et C est un atome
  - $x_1 \ldots x_n$ sont les variables de H
  - *toutes* les variables de C apparaissent dans H.
- **homomorphisme** : étant donnés deux ensembles d'atomes A et B, un homomorphisme de A dans B est une *application* des variables de A dans les termes de B telle que $h(A) \subseteq B$, où h(A) est l'ensemble d'atomes obtenu de A en substituant chaque variable x par h(x).
- **requête conjonctive** (en fait, une simplification de la notion classique) : une conjonction d'atomes ; l'ensemble des réponses à une requête q sur une base de faits F est l'ensemble des homomorphismes de q dans F; si q n'a pas de variable, le seul homomorphisme possible de q dans F est la substitution vide, auquel cas $q \subseteq F$; on voit alors q comme une requête booléenne et on dit que F répond oui à q si et seulement si $q \subseteq F$.

Exercice 1. Calcul d'homomorphismes

Soit Q et F deux ensembles d'atomes (représentant respectivement une requête conjonctive et une base de faits). On n'a donc pas de règles.

Q = { p(x,y), p(y,z), p(x,u), p(z,z), r(x), r(u) } où x, y, z et u sont des variables

F = { p(a,b), p(a,c), p(b,c), p(b,d), p(b,e), p(d,e), p(e,e), r(a), r(b), r(c) } où a, b, c, d et e sont des constantes.

1- L'application suivante des variables de Q dans les constantes de F est-elle un homomorphisme de Q dans D ? Justifier votre réponse.
   $x \rightarrow a \quad y \rightarrow b \quad z \rightarrow d \quad u \rightarrow c$
2- Donner tous les homomorphismes de Q dans F

Exercice 2. Chaînage avant

On considère la base de connaissances suivante (où sg signifie intuitivement "same ground") :

- Règles

  R1: flat(x1,y1) → sg(x1,y1)
  R2 : up(x2,y2) ∧ sg(y2,z2) ∧ up(t2,z2) → sg(x2,t2)

- Faits (où les termes sont des constantes)

  flat(a,b) flat(b,c) up(d,a) up(d,b) up(e,c) up(f,d) up(g,e)

1) **Saturez** la base de faits avec les règles, en procédant **en largeur**. A chaque étape, on ne considère que les **nouveaux** homomorphismes. On rappelle qu'une application de règle est dite utile si elle produit un fait qui n'appartient pas à la base de faits courante.

Etape	Règle applicable	Homomorphisme	Fait produit	Application utile ?
n° étape	n° règle	…	…	oui/non
…	…	…	…	…

2) On dit qu'un prédicat est *calculé* s'il apparaît au moins une fois en conclusion de règle : ici, *sg* est un prédicat calculé, et c'est le seul. L'ensemble de règles ci-dessus a une particularité : la condition (l'hypothèse) de chaque règle contient au plus un atome ayant un prédicat calculé. Un tel ensemble de règles est appelé *datalog linéaire*. Comment exploiter le fait qu'un ensemble de règles soit linéaire pour se restreindre facilement aux homomorphismes *nouveaux* à chaque étape de largeur ?

3) Soit la requête qui demande de trouver x, y et z tels que $sg(x,y) \wedge up(y,z) \wedge flat(z,c)$, où *c* est une constante. Quel est l'ensemble de réponses à cette requête sur la base de connaissances ? Justifiez votre réponse en vous basant sur le mécanisme de *chaînage avant*.

Exercice 3. De CSP à Homomorphisme et réciproquement

Le problème **CSP** *(satisfaction de contraintes)* prend en entrée un réseau de contraintes et demande si ce réseau admet une solution. Un réseau de contraintes est composé d'un ensemble $X = \{x_1, \ldots, x_n\}$ de variables, d'une fonction D qui associe un domaine D_i à chaque variable x_i, et d'un ensemble C de contraintes portant sur les variables de X. On se restreint ici à des contraintes données en extension (c'est-à-dire par énumération des tuples de valeurs compatibles).

Le problème **HOM** *(homomorphisme)* prend en entrée 2 ensembles d'atomes représentant des formules existentielles conjonctives de la logique du premier ordre (soient Q et F), et demande s'il existe un homomorphisme du premier (Q) dans le second (F). On suppose que F ne contient pas de variable.

Nous allons voir dans cet exercice que les deux problèmes sont réductibles l'un à l'autre par des transformations assez naturelles. Intuitivement, une solution au réseau (X,D,C) est une application de X dans D qui respecte les contraintes de C, et un homomorphisme de Q dans F est une application des variables de Q dans les termes (constantes) de F qui « respecte les atomes de Q ».

1) De CSP à HOM

a) On considère le réseau de contraintes (X,D,C) suivant :

X={x1,x2,x3,x4}
D(xi) = {a,b} pour tout i : 1 ... 4
C={C1,C2,C3} avec :
C1[x1,x2,x3] = {[a,a,b], [a,b,a], [b,a,a]}
C2[x2,x3,x4] = {[a,b,a], [b,a,b]}
C3[x1,x4] = {[a,b],[b,b]}

Convertir ce réseau en une instance (Q,F) du problème HOM de façon à ce que le réseau ait une solution si et seulement s'il existe un homomorphisme de Q dans F. De plus, on voudrait que les solutions du réseau correspondent exactement aux homomorphismes de Q dans F.

b) On considère le réseau de contraintes modélisant le problème de coloration des États de l'Australie, où les contraintes de différence sont exprimées en extension : le traduire en une instance du problème HOM.
Définir une réduction de CSP à HOM dans le cas général.

2) De HOM à CSP

On considère Q et F de l'exercice 2 :
Q = { p(x,y), p(y,z), p(x,u), p(z,z), r(x), r(u) }
F = { p(a,b), p(a,c), p(b,c), p(b,d), p(b,e), p(d,e), p(e,e), r(a), r(b), r(c) }

a) Transformer Q et F en un réseau de contraintes au plus binaires P = <X,D,C> de façon à ce que tout homomorphisme de Q dans F soit une solution à P, et réciproquement. Penser à traiter le cas où une variable apparaît deux fois dans le même atome de Q (cf. p(z,z)).

b) Définir une réduction de HOM à CSP dans le cas général. Ne pas oublier que Q peut comporter des constantes.