Université de Montpellier - Master Informatique

Théorie des bases de connaissances (HAI933I)

Contrôle n°2 - Décembre 2024

Durée de l'épreuve : 1 heure

Aucun document autorisé

Le barème est donné à titre indicatif.

Exercice 1. Datalog et UCQs (3 pts)

On rappelle qu'une union de requêtes conjonctives (UCQ) est de la forme $Q(x_1, \ldots, x_k) = q_1(x_1, \ldots, x_k) \vee \ldots \vee q_n(x_1, \ldots, x_k)$, où chaque q_i $(1 \leq i \leq n)$ est une requête conjonctive dont les variables réponses sont $x_1, \ldots, x_k$.

Question 1.1. Comment transformer une UCQ en une requête Datalog équivalente, c'est-à-dire ayant les mêmes réponses sur n'importe quelle base de faits ? Dans votre réponse, utilisez les notations de l'énoncé.

Question 1.2. Datalog (positif) est-il plus expressif que le langage des UCQs, c'est-à-dire permet-il d'exprimer des requêtes qui n'ont pas de requête UCQ équivalente ? Justifiez votre réponse.

Exercice 2. Règles existentielles (10 pts)

On se place dans le formalisme des règles existentielles et on considère deux variantes du chase :

- l'*oblivious* chase, qui effectue toutes les applications de règle possibles
- le *restricted* chase, qui n'effectue l'application d'une règle $R = B \rightarrow H$ selon un homomorphisme h de B dans la base de faits courante F_i que si h ne *s'étend pas* à un homomorphisme de H dans F_i;

Question 2.1. Qu'est-ce qu'un modèle *universel* d'une base de connaissances ?

Question 2.2 Soit une base de connaissances $\mathcal{K} = (F, \mathcal{R})$ avec $F = \{p(a,b), p(b,a)\}$, où a et b sont des constantes, et $\mathcal{R} = \{R_1, R_2\}$, où :
$R_1 : p(x,y) \rightarrow r(y)$
$R_2 : r(x) \rightarrow \exists z\, p(x,z)$
Quel est le résultat de l'oblivious chase appliqué à $\mathcal{K}$? On ne vous demande pas d'indiquer les étapes du chase.

Question 2.3 Quel est le résultat du restricted chase appliqué à $\mathcal{K}$? Détaillez ici les applications des règles.

Question 2.4 $\mathcal{K}$ possède-t-elle un modèle universel fini ? Justifiez votre réponse.

Question 2.5 On considère l'interprétation I dont le domaine est $D = \{a, b\}$ (avec la simplification adoptée en cours : chaque constante est interprétée par un élément du domaine de même nom) et telle que : $I(r) = D = \{a, b\}$ $I(p) = \{(a, a), (a, b), (b, a)\}$.
On peut aussi voir cette interprétation comme l'ensemble d'atomes $\{r(a), r(b), p(a, a), p(a, b), p(b, a)\}$.
(a) I est-elle un modèle de $\mathcal{K}$? Expliquez.
(b) I est-elle un modèle universel de $\mathcal{K}$? Expliquez.

Exercice 3. Modèles stables - version propositionnelle (7+ pts)

Dans cet exercice, nous ne considérons que des programmes propositionnels. Les définitions et notations utiles sont rappelées en annexe (faites attention à la définition du corps négatif qui, dans le cas propositionnel, est simplifiée). Pour les questions marquées par $*$, un peu plus difficiles, des points bonus seront accordés suivant la qualité de la justification ou de la démonstration.

Utilisation de la définition par point fixe

Soit Π le programme suivant (où a,b, etc. sont des symboles propositionnels).

$$a \qquad b, not\, c \to d \qquad d, not\, f \to f$$

$$a \to b \qquad b, not\, d \to c \qquad c, not\, g \to h$$

Question 3.1 Utiliser la méthode du point fixe pour savoir si $E_1 = \{a, b, c, h\}$ est un modèle stable de Π. Vous donnerez explicitement le programme réduit pour justifier votre réponse.

Question 3.2 Même question avec $E_2 = \{a, b, d\}$.

Question 3.3 Même question avec $E_3 = \{a, g\}$. Pour cette question, vous pourrez éviter le calcul du programme réduit par une remarque judicieuse.

Génération de tous les modèles stables

Un algorithme immédiat pour générer tous les modèles stables d'un programme Π est de calculer son *vocabulaire* V_Π (c'est-à-dire l'ensemble des atomes apparaissant dans une règle de Π), de générer chaque sous-ensemble E de V_Π, puis de tester E par la définition par point fixe. Cet algorithme fonctionne en temps $2^{|V_\Pi|} \times poly(|\Pi|)$. Il est donc urgent de réduire le nombre de sous-ensembles candidats à tester. Dans les questions suivantes, nous allons calculer deux sous-ensembles V_Π^{min} et V_Π^{max} de V_Π qui ont les propriétés suivantes :

- si $a \in V_\Pi^{min}$, alors a est dans tous les modèles stables de Π;
- si $a \notin V_\Pi^{max}$, alors a n'est dans aucun modèle stable de Π.

Ainsi, il sera suffisant de tester tous les sous-ensembles de V_Π^{max} contenant V_Π^{min} pour trouver tous les modèles stables de Π.

Question 3.4 $*$ Trouvez une méthode pour calculer un V_Π^{max} (qui sera, pour certains programmes, strictement inclus dans V_Π). *Indice :* revoyez la question 3.3. Vous justifierez votre réponse.

Question 3.5 * ChatGPT m'a proposé pour calculer V_Π^{min} la notion d'*ensemble garanti*, défini comme l'ensemble des faits du programme Π. Il m'a même (correctement) démontré que tous les atomes de l'ensemble garanti sont dans tous les modèles stables de Π. Proposez-moi pour V_Π^{min} un ensemble possiblement plus grand que l'ensemble garanti utilisé par ChatGPT. *Indice* : vous pouvez penser à la règle $a \rightarrow b$ du programme précédent. Vous justifierez soigneusement votre réponse, par une démonstration rigoureuse.

Annexe A: rappels de cours, et notations

En version propositionnelle, une *règle* est un objet R de la forme

$$p_1, \ldots, p_k, not\, n_1, \ldots, not\, n_q \rightarrow h$$

où $h, p_1, \ldots, p_k, n_1, \ldots, n_q$ sont des atomes propositionnels. On note $head(R) = h$, $body^+(R) = \{p_1, \ldots, p_k\}$ et $body^-(R) = \{n_1, \ldots, n_q\}$. Une règle R est un *fait* lorsque $body^+(R) = body^-(R) = \emptyset$ (et dans ce cas on la note simplement h), et elle est dite *positive* lorsque $body^-(R) = \emptyset$. La *partie positive* d'une règle R est la règle $pos(R) = p_1, \ldots, p_k \rightarrow h$ obtenue en enlevant tous les atomes du corps négatif.

Un programme Π est un ensemble de règles. Un programme sera dit *positif* lorsque toutes ses règles sont positives. La *partie positive* de Π est le programme (positif) Π^+ constitué de toutes les règles positives de Π. Si Π est un programme positif, la *saturation* de Π est l'ensemble d'atomes Π^* obtenu par une dérivation complète de Π (c'est donc son *modèle canonique*).

Soit Π un programme, et E un ensemble de règles, alors le *programme* Π *réduit par* E, noté $\Pi_{|E}$, est l'ensemble de règles obtenu de la façon suivante: pour chaque règle R de Π, si $body^-(R) \cap E = \emptyset$, alors $\Pi_{|E}$ contient $pos(R)$, sinon la règle est absente de $\Pi_{|E}$. Un *modèle stable* de Π est un ensemble d'atomes E tel que $(\Pi_{|E})^* = E$ (c'est la *définition par point fixe*).