Bases de données

L2 sciences et technologies, mention informatique

union et négation : requêtes relationnelles

ou : comment exprimer "lister les réalisateurs n'ayant pas réalisé un film chaque année" ?

films	titre	réalisateur	année
	starwars	lucas	1977
	nikita	besson	1990
	locataires	ki-duk	2005
	dune	lynch	1984

réalisateurs	nom	nationalité
	lucas	américaine
	lynch	américaine
	besson	française
	ki-duk	coréenne

patrick.marcel@univ-tours.fr
http://celene.univ-tours.fr/course/view.php?id=3131

plan de la séance

1. union

   quels sont les films réalisés par "lynch" ou "ki-duk" ?

2. négation

   quels sont les films qui ne sont pas réalisés par "besson" ?

   existe-t-il des réalisateurs dans l'instance de "films" qui ne sont pas dans l'instance de "réalisateurs" ?

3. le cas particulier du calcul relationnel

union

union

quels sont les films réalisés par "lynch" ou "ki-duk" ?

en algèbre, on aimerait écrire :

$$\pi_{titre}(\sigma_{réalisateur=''lynch''}(\text{films}) \cup \sigma_{réalisateur=''ki-duk''}(\text{films}))$$

ou encore

$$\pi_{titre}(\sigma_{réalisateur=''lynch'' \vee réalisateur=''ki-duk''}(\text{films}))$$

union

quels sont les films réalisés par “lynch” ou “ki-duk” ?

avec les règles, on aimerait écrire :

résultat(t) $\leftarrow$films(t,”lynch”,a)

résultat(t) $\leftarrow$films(t,”ki-duk”,a)

union

quels sont les films réalisés par “lynch” ou “ki-duk” ?

avec le calcul, on aimerait écrire :

$$\{t | \exists a_1, a_2,\ \text{films}(t, \text{"lynch"}, a_1) \vee \text{films}(t, \text{"ki-duk"}, a_2)\}$$

union et algèbre

opération $\cup$ ajoutée à SPC

soit I_1 et I_2 deux instances de relations ayant même arité :

$$I_1 \cup I_2 = \{t | t \in I_1 \text{ ou } t \in I_2\}$$

on obtient l'algèbre SPCU

union et algèbre

opération $\cup$ ajoutée à SPJR

I_1 et I_2 doivent avoir même sorte

on obtient l'algèbre SPJRU

union et algèbre

la sélection est généralisée aux formules disjonctives

les formes normales pour les requêtes sont définies comme suit :

une requête SPCU (resp., SPJRU) est sous forme normale si elle est l'union de deux requêtes SPC (resp., SPJR) sous forme normale

exemple

$$\pi_{titre}(\sigma_{réalisateur='' lynch'' \vee réalisateur='' ki-duk''}(\text{films}))$$

n'est pas sous forme normale

$$\pi_{titre}(\sigma_{réalisateur='' lynch''}(\text{films})) \cup \pi_{titre}(\sigma_{réalisateur='' ki-duk''}(\text{films}))$$

est sous forme normale

union et langage de règles

on peut définir des ensembles de règles ayant le même nom de relation dans la tête

un programme *nr-datalog* sur un schéma R est un ensemble de n règles où

- il n'y a pas de nom de relation de R dans la tête
- il existe un ordre des règles de r_1 à r_n tel que
- le nom de relation dans la tête de r_i n'apparait pas dans le corps de de r_j si $j \leq i$

évaluation des programmes

soit I une instance de base de données et P un programme nr-datalog

chaque règle est évaluée dans un ordre satisfaisant les critères définis plus haut

l'union des faits est réalisée si 2 règles ont le même nom de relation dans la tête

nr-datalog

le langage obtenu s'appelle *nr-datalog*

pourquoi "nr" ? pour *n*on *r*ecursive

parce que ce genre de règle n'est pas autorisée :

$$\text{ancêtre}(x,z) \leftarrow \text{parent}(x,y),\ \text{ancêtre}(y,z).$$

exemple

soit le programme suivant :

r_1 résultat(t) $\leftarrow$films(t,"lynch",a)
r_2 résultat(t) $\leftarrow$films(t,"ki-duk",a)

on ordonne les règles dans l'ordre r_1,r_2 et on évalue les règles dans cet ordre

évaluation de r_1 { résultat(dune) }

évaluation de r_2 : { résultat(dune), résultat(locataires) }

union et calcul

le problème : on peut écrire $\{x,y|R(x) \vee R(y)\}$

c'est une requête syntaxiquement correcte, mais...

les variables x et y sont libres

donc la réponse est infinie !

les requêtes de ce type ne sont pas *sûres*

on règle ça tout à l'heure...

théorème

les langages suivants sont équivalents :

- l'algèbre SPCU
- l'algèbre SPJRU
- nr-datalog

ces langages expriment :

- les requêtes conjonctives avec union
- qui sont closes par composition

négation

en algèbre

quels sont les films qui ne sont pas réalisés par “besson” ?

$$\pi_{titre}(films - \sigma_{réalisateur=''besson''}(films))$$

ou

$$\pi_{titre}(\sigma_{réalisateur\neq''besson''}(films))$$

en algèbre

existe-t-il des réalisateurs dans l'instance de "films" qui ne sont pas dans l'instance de "réalisateurs" ?

$$\pi_{réalisateur}(films) - \pi_{nom}(réalisateurs)$$

en langage de règles

quels sont les films qui ne sont pas réalisés par "besson" ?

$$ans(x) \leftarrow films(x,y,z), \neg films(x,''besson'',z).$$

existe-t-il des réalisateurs dans l'instance de "films" qui ne sont pas dans l'instance de "réalisateurs" ?

réalisateursFilms(r) $\leftarrow$ films(t,r,a)
réalisateursRéalisateurs(r) $\leftarrow$ réalisateurs(r,n)
ans(x) $\leftarrow$ réalisateursFilms(x), $\neg$ réalisateursRéalisateurs(x)

algèbre

opérateur $-$ défini sur des relations ayant même sorte (SPJRU) ou même arité (SPCU)

la sélection est généralisée pour permettre des formules autorisant la négation

$\pi_{titre}(\sigma_{année=1990 \wedge réalisateur \neq'' besson''}(films))$

attention !

la différence ne peut pas être exprimée comme une jointure

exemple : soit A={(1),(2)} et B={(1),(3)}

$A - B \neq \pi_1(\sigma_{1\neq 2}(A \times B))$

algèbre relationnelle

le langage obtenu s'appelle l'*algèbre relationnelle*

ses propriétés :

- permet d'exprimer des requêtes non monotones
- permet d'exprimer des requêtes non satisfiables
- clos par composition

nr-datalog$^\neg$

nr-datalog$^\neg$: nr-datalog dans lequel des litéraux négatifs sont permis dans le corps des règles
une règle nr-datalog$^\neg$ est de la forme :

$$q : S(u) \leftarrow L_1, \ldots, L_n.$$

- S est un nom de relation
- S n'apparait pas dans le corps de q
- u est un tuple libre
- les L_i sont des litéraux de la forme $R(v)$ ou $\neg R(v)$
  - R est un nom de relation
  - v un tuple libre

nr-datalog$^\neg$

$$q : S(u) \leftarrow L_1, \ldots, L_n.$$

q est à champs restreint si chaque variable apparait *au moins une fois dans un literal positif* de la forme $R(v)$ du corps

dans ce qui suit on considère uniquement des règles à champs restreint

exemple

$ans(x) \leftarrow films(x,y,z),\neg films(x,''besson'',z).$

ans(x) ←films(x,y,a), ¬ réalisateurs(y,"américaine").

sont à champs restreint

$ans(x) \leftarrow \neg films(x,''besson'',z).$

ans(y) ←films(x,y,z), ¬ films(w,y,"1991").

ne sont pas à champs restreint

nr-datalog$^\neg$

soient

- q une règle nr-datalog$^\neg$ $S(u) \leftarrow L_1, \ldots, L_n$.
- D un schéma de base de données qui inclut tous les noms de relation apparaissant dans le corps de q
- I une instance de D

l'image de I par q est :

$$q(I) = \{v(u) | v \text{ est une valuation et pour chaque } i \in [1,n],$$
$$v(u_i) \in I(R_i) \text{ si } L_i = R_i(u_i),\ v(u_i) \notin I(R_i) \text{ si } L_i = \neg R_i(u_i)\ \}$$

nr-datalog$^\neg$

un programme nr-datalog$^\neg$ sur un schéma R est un ensemble de n règles nr-datalog$^\neg$ tel que

- aucune relation de R apparait dans la tête
- il existe un ordre $r_1, \ldots, r_n$ tel que le nom de relation dans la tête de r_i n'apparait pas dans r_j si $i \geq j$.

exemple

r_1 réalisateursFilms(r) $\leftarrow$ films(t,r,a)
r_2 réalisateursRéalisateurs(r) $\leftarrow$ réalisateurs(r,n)
r_3 ans(x) $\leftarrow$ réalisateursFilms(x), $\neg$ réalisateursRéalisateurs(x)

on ordonne les règles, par exemple dans l'ordre r_2,r_1,r_3
et on évalue selon cet ordre :

{ réalisateursFilms(lucas), réalisateursFilms(besson),
réalisateursFilms(ki-duk), réalisateursFilms(lynch),
réalisateursRéalisateurs(lucas), réalisateursRéalisateurs(besson),
réalisateursRéalisateurs(ki-duk), réalisateursRéalisateurs(lynch) }

thèorème

l'algèbre relationnelle et les programmes nr-datalog$^{\neg}$ ont la même puissance d'expression

Calcul relationnel

le calcul relationnel

on ajoute au calcul conjonctif :

- la négation $\neg$
- la disjonction $\vee$
- la quantification universelle $\forall$

remarque

si φ et ψ sont deux formules :

$$\begin{aligned} \varphi \vee \psi &\equiv \neg(\neg\varphi \wedge \neg\psi) \\ \forall x \varphi &\equiv \neg\exists x \neg\varphi \end{aligned}$$

exemple

Soit la relation vu[spectateur, titre]

$$\forall s, vu(s, \text{"}starwars\text{"}) \equiv \neg \exists s \neg vu(s, \text{"}starwars\text{"})$$

syntaxe

soit D un schéma de base de données, $e, e' \in \textbf{dom} \cup \textbf{var}$

les formules du *calcul relationnel* sont :

1. un atome sur D
2. $e = e'$ ou $e \neq e'$
3. $\varphi \wedge \psi$ où φ et ψ sont des formules
4. $\varphi \vee \psi$ où φ et ψ sont des formules
5. $\neg\varphi$ où φ est une formule
6. $\exists x \varphi$ où φ est une formule
7. $\forall x \varphi$ où φ est une formule

syntaxe

on utilisera les abréviations suivantes :

$$\begin{aligned} \varphi \to \psi &\equiv \neg\varphi \vee \psi \\ \varphi \leftrightarrow \psi &\equiv (\varphi \wedge \psi) \vee (\neg\varphi \wedge \neg\psi) \end{aligned}$$

variable libre/liée

$free(f)$ est l'ensemble des variables libres de f

- si R est un atome, $free(R) = var(R)$
- $free(e = e') = free(e \neq e') = \{e, e'\}$
- $free(\varphi \wedge \psi) = free(\varphi) \cup free(\psi)$
- $free(\varphi \vee \psi) = free(\varphi) \cup free(\psi)$
- $free(\neg\varphi) = free(\varphi)$
- $free(\exists x \varphi) = free(\varphi) - \{x\}$
- $free(\forall x \varphi) = free(\varphi) - \{x\}$

un variable non libre est une variable *liée*

exemple

dans la formule φ suivante

$$\forall z \exists y, \mathit{films}(y,x,z)$$

$\mathit{free}(\varphi) = \{x\}$

requêtes

une requête du calcul relationnel est une expression de la forme

$$\{e_1, \ldots, e_n | \varphi\}$$

où φ est une formule du calcul relationnel, et les variables de $(e_1, \ldots, e_n)$ sont exactement $\mathit{free}(\varphi)$

exemple

“quels sont les films sortis en 2005 dont le réalisateur n’est pas américain”

$$\{t | \exists r, films(t, r, ''2005'') \land \neg réalisateurs(r, ''américaine'')\}$$

“quelles sont les nationalités des réalisateurs ayant sorti un film uniquement en 2000 ?”

$$\{n | \exists r, réalisateurs(r, n) \land \exists t films(t, r, 2000) \land \neg(\exists a, t'(films(t', r, a) \land a \neq 2000))\}$$

les problèmes (1)

$$q = \{x | \neg films(x, "lucas", "1977")\}$$

ou

$$q = \{x, y | films(x, "lucas", "1977") \vee films("starwars", "lucas", y)\}$$

$q(I)$ infinie quelle que soit I...

car x et y prennent leurs valeurs dans **dom**, qui est infini !

donc il existe une infinité de valeurs possibles pour x et y qui satisfont ces expressions

exemple

$$q = \{x | \neg films(x, "lucas", "1977")\}$$

$$I(films) = \{films(starwars, lucas, 1977)\}$$

$$q(I) = \{ET, 2001, Dune, \ldots\}$$

solutions

1. les variables prennent leurs valeurs dans $adom(q,I)$
   ou
2. interdire (syntaxiquement) les requêtes non sûre

on étudiera la première solution, le langage $CALC_{adom}$

les problèmes (2)

$q = \{x | \forall y, R(x,y)\}$

si y prend ses valeurs sur **dom**, $q(I) = \emptyset$ quelle que soit I

si y prend ses valeurs sur un ensemble fini, $q(I)$ peut être non vide

cette requête est dépendante du domaine sur lequel elle est interprétée

exemple

Soit la relation vu[spectateur, titre]

$q = \{t \mid \forall s, vu(s,t)\}$

$I(vu) = \{vu(s1,t1), vu(s2,t1), vu(s1,t2)\}$

si s prend ses valeurs sur $\{s1\}$ alors $q(I) = \{t1,t2\}$

si s prend ses valeurs sur $\{s1,s2\}$ alors $q(I) = \{t1\}$

si s prend ses valeurs sur $\{s1,s2,s3\}$ alors $q(I) = \emptyset$

solutions

- fixer le domaine d'interprétation

  ou
- interdire (syntaxiquement) ce type de requêtes

on étudiera la première solution, le domaine d'interprétation est fixé à $adom(q,I)$

sémantique domaine actif

$q = \{e_1, \ldots, e_n | \varphi\}$ une requête et I une instance

on se restreind aux valuations de $free(\varphi)$ dans $adom(q,I)$

dans la suite :

- v est une valuation satisfaisant cette restriction
- on note $adom = adom(q,I)$

sémantique domaine actif

I satisfait φ pour v relativement à $adom(q,I)$, noté $I \models_{adom} \varphi[v]$ si

1. $\varphi = R(u)$ est un atome et $v(u) \in I(R)$
2. $\varphi = (s = s')$ où $s,s' \in \mathbf{dom} \cup \mathbf{var}$ et $v(s) = v(s')$
3. $\varphi = (\psi \wedge \xi)$ et $I \models_{adom} \psi[v|_{free(\psi)}]$ et $I \models_{adom} \xi[v|_{free(\xi)}]$
4. $\varphi = (\psi \vee \xi)$ et $I \models_{adom} \psi[v|_{free(\psi)}]$ ou $I \models_{adom} \xi[v|_{free(\xi)}]$
5. $\varphi = \neg\psi$ et $I \not\models_{adom} \psi[v]$
6. $\varphi = \exists x \psi$ et il existe $c \in adom$, $I \models_{adom} \psi[v \cup \{x/c\}]$
7. $\varphi = \forall x \psi$ et pour tout $c \in adom$, $I \models_{adom} \psi[v \cup \{x/c\}]$

image d'une requête

soit D un schéma de base de données, $q = \{e_1, \ldots, e_n | \varphi\}$ une requête sur D, et I une instance de D

l'image de I par q, notée $q(I)$, est :

$$q(I) = \{v((e_1, \ldots, e_n)) | I \models_{adom} \varphi[v] \text{ et } v \text{ est une valuation sur } free(\varphi) \text{ dont l'image} \subseteq adom(q, I)\}$$

exemple

soit l'instance I :

$I(films) = \{films(starwars, lucas, 1977)\}$
$I(vu) = \{vu(s1, t1), vu(s2, t1), vu(s1, t2)\}$

et les requêtes :

$q_1 = \{x | \neg films(x, "lucas", "1977")\}$
$q_2 = \{t \mid \forall s, vu(s, t)\}$

adom = {starwars, lucas, 1977, s1, s2, t1, t2 }

$q_1(I) = \{lucas, 1977, s1, s2, t1, t2\}$
$q_2(I) = \emptyset$

théorème

les langages suivants sont équivalents :

- algèbre relationnelle
- nr-datalog$^{\neg}$
- CALC$_{adom}$