Databases

L2 sciences et technologies, mention informatique

conjunctive queries

how to query this database (with a logic-based language)?

movies	title	director	year
	starwars	lucas	1977
	nikita	besson	1990
	locataires	ki-duk	2005
	dune	lynch	1984

directors	name	nationality
	lucas	american
	lynch	american
	besson	french
	ki-duk	korean

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

queries

examples of queries:

1. who directed “dune” ?
2. what is the release year of “nikita” ?
3. what is the nationality of the director of “locataires” ?
4. list movies directed by americans

query #4

"list movies directed by americans"

with variables ranging over tuples:

if there are tuples t_1, t_2 in *movies* and *directors*
such that *nationality* of t_2 is "american"
and *director* of t_1 = *name* of t_2

then answer contains the *title* of t_1

query #4

“list movies directed by americans”

with variables ranging over the constants of **Dom**:

if there are tuples $(r,$"american"$),(t,r,a)$ in *directors* and *movies*
then tuple (t) is included in the answer

query #4

with a rule-based formulation:

american_movies(t) $\leftarrow$directors(r,"american"),movies(t,r,a).

if

- there exists a value for r associated with "american" in the instance of directors, and
- this value is also in the instance of movies associated with some values for title and year,

then the value of t associated with the value of r in the instance of movies is included in the answer

rule-based conjunctive queries

rule-based language

a *conjunctive query* over a database schema D is an expression of the form:

$$ans(u) \leftarrow R_1(u_1), \ldots, R_n(u_n)$$

rule-based language

a *conjunctive query* over a database schema D is an expression of the form:

$$ans(u) \leftarrow R_1(u_1), \ldots, R_n(u_n)$$

- $ans(u)$ is called the *head* of the rule
- $R_1(u_1), \ldots, R_n(u_n)$ is called the *body* of the rule
- the $R(u_i)$'s are called *atoms*

in this rule

R_i is a relation name in D

$ans \notin D$ is a relation name

u_i is an expression of the form $e_1, \ldots, e_{m_i}$

the e_j's are variables of **var** or constants of **dom**

the variables of this rule

they are *range restricted*:

each variable appearing in u must appear at least once in $u_1, \ldots, u_n$

the set of variables of query q is noted $var(q)$

example

"who is the director of dune?"

$$ans(r) \leftarrow movies(\text{"}dune\text{"}, r, a).$$

example

“who is the director of dune?”

$$ans(r) \leftarrow movies(\text{“}dune\text{''}, r, a).$$

“what is the release year of nikita?”

$$ans(a) \leftarrow movies(\text{“}nikita\text{''}, r, a).$$

valuation

let $V \subset \mathbf{var}$

a *valuation* v over V is a function from V to $\mathbf{dom}$

valuation

let $V \subset \mathbf{var}$

a *valuation* v over V is a function from V to **dom**

a valuation v associates a value with each variable

free tuple

let U be a set of attributes in the named approach

a *free tuple* over U is a function form U to $\mathbf{var} \cup \mathbf{dom}$

free tuple

let U be a set of attributes in the named approach

a *free tuple* over U is a function form U to $\mathbf{var} \cup \mathbf{dom}$

let t be a free tuple and v be a valuation

$v(t)$ is the tuple t where variables are replaced by their valuation

example

let $V = \{t, r, a\} \subset \mathbf{var}$

v_1, v_2, v_3 are three valuations :

- $v_1(t) = \text{starwars},\ v_1(r) = \text{lucas},\ v_1(a) = 1977$

example

let $V = \{t, r, a\} \subset \mathbf{var}$

v_1, v_2, v_3 are three valuations :

- $v_1(t) = \text{starwars},\ v_1(r) = \text{lucas},\ v_1(a) = 1977$
- $v_2(t) = \text{dune},\ v_2(r) = \text{lynch},\ v_2(a) = 1984$

example

let $V = \{t, r, a\} \subset \textbf{var}$

v_1, v_2, v_3 are three valuations :

- $v_1(t) = \text{starwars},\ v_1(r) = \text{lucas},\ v_1(a) = 1977$
- $v_2(t) = \text{dune},\ v_2(r) = \text{lynch},\ v_2(a) = 1984$
- $v_3(t) = 1977,\ v_3(r) = 1984,\ v_3(a) = 1977$

the image of a query q

let q be a query $ans(u) \leftarrow R_1(u_1), \ldots, R_n(u_n)$

let I be a database instance of schema D

the *image* of (the answer to) q over I is:

$$q(I) = \{v(u) | v \text{ is a valuation over } var(q) \text{ and } \\ v(u_i) \in I(R_i) \text{ for all } i \in [1, n] \}$$

example

query #4: $\text{american\_movies}(t) \leftarrow \text{directors}(r,\text{"american"}),\text{movies}(t,r,a).$

example

query #4: american_movies(t) $\leftarrow$directors(r,"american"),movies(t,r,a).

consider I the following database instance

I={movies(starwars,lucas,1977),movies(nikita,besson,1990),
movies(locataires,ki-duk,2005),movies(dune,lynch,1984)
directors(lucas,american),directors(lynch,american),
directors(ki-duk,korean),directors(besson,french)}

example

valuations v_1 and v_2 such that:

- $v_1(\mathrm{t}) = \mathrm{starwars}$, $v_1(\mathrm{r}) = \mathrm{lucas}$, $v_1(\mathrm{a}) = 1977$
- $v_2(\mathrm{t}) = \mathrm{dune}$, $v_2(\mathrm{r}) = \mathrm{lynch}$, $v_2(\mathrm{a}) = 1984$

example

valuations v_1 and v_2 such that:

- $v_1(\mathrm{t}) = \mathrm{starwars}$, $v_1(\mathrm{r}) = \mathrm{lucas}$, $v_1(\mathrm{a}) = 1977$
- $v_2(\mathrm{t}) = \mathrm{dune}$, $v_2(\mathrm{r}) = \mathrm{lynch}$, $v_2(\mathrm{a}) = 1984$

$q(I) = \{\mathrm{american\_movies}(\text{“starwars”}), \mathrm{american\_movies}(\text{“dune”})\}$

is the answer to the query

example

$$ans(w) \leftarrow movies(x, y, z)$$

is not range restricted

logically, the answer to this query is infinite...

active domain

for a database instance I and a query q

we note:

$adom(I)$	the set of constants appearing in I *the active domain of the instance*
$adom(q)$	the set of constants appearing in q *the active domain of the query*

example

in the previous example:

$adom(I)=\{\text{starwars, lucas, american, 1984, dune, } \ldots\}$

$adom(q)=\{\text{american}\}$

what is $q(I)$?

we note $adom(q, I) = adom(q) \cup adom(I)$

q is a range restricted query over I

therfore $adom(q(I)) \subseteq adom(q, I)$

therefore $q(I)$ is a finite set

therefore it is an instance

extension and intention

$$ans(u) \leftarrow R_1(u_1), \ldots, R_n(u_n)$$

if relations R_i are stored

they are said to be defined in *extension*

extension and intention

$$ans(u) \leftarrow R_1(u_1), \ldots, R_n(u_n)$$

if relations R_i are stored

they are said to be defined in *extension*

if *ans* is not stored

it is said to be an *intentional* definition

boolean query

example: is there a movie whose release year is 1990?

boolean query

example: is there a movie whose release year is 1990?

$$ans() \leftarrow film(t, r, 1990)$$

answer

$\{()\}$ then yes

$\emptyset$ otherwise

conjunctive calculus

conjunctive calculus

$$ans(e_1, \ldots, e_m) \leftarrow R_1(u_1), \ldots, R_n(u_n)$$

syntactical variation:

$$\{e_1, \ldots, e_m | \exists x_1, \ldots, x_k (R_1(u_1) \wedge \ldots \wedge R_n(u_n))\}$$

conjunctive calculus

$$ans(e_1, \ldots, e_m) \leftarrow R_1(u_1), \ldots, R_n(u_n)$$

syntactical variation:

$$\{e_1, \ldots, e_m | \exists x_1, \ldots, x_k (R_1(u_1) \wedge \ldots \wedge R_n(u_n))\}$$

- $x_1, \ldots, x_k$ are variables appearing in the body and not in the head
- $\wedge$ is the logical conjunction ("and")
- $\exists$ is the existential quantification ("there exists")

example

query #4 expressed in the conjunctive calculus:

$$\{t | \exists r, a, \text{ directors}(r, \text{"american"}) \wedge \text{movies}(t, r, a)\}$$

the syntax of the conjunctive calculus

let D be a database schema

a formula over D is an expression of the form:

1. an atom $R(e_1, \ldots, e_n)$ over D
2. $(\varphi \wedge \psi)$ where φ and ψ are formulas over D, or
3. $\exists x \varphi$ where x is a list of variables and φ is a formula over D

example

conjunctive calculus formulas:

movies(”starwars”, r,”1977”)

example

conjunctive calculus formulas:

movies("starwars", r, "1977")

directors("lucas", n) $\wedge$ directors("lynch", n)

example

conjunctive calculus formulas:

movies("starwars", r, "1977")

directors("lucas", n) $\wedge$ directors("lynch", n)

$\exists y$ directors(x, "french") $\wedge$ movies("starwars", x, y)

free/bound variable

an occurrence of a variable x in a formula φ is *free* if

1. φ is an atom, or
2. $\varphi = (\psi \wedge \xi)$ where x is free in ψ or ξ
3. $\varphi = \exists y \psi$ where y is distinct of x and x is free in ψ

free/bound variable

an occurrence of a variable x in a formula φ is *free* if

1. φ is an atom, or
2. $\varphi = (\psi \wedge \xi)$ where x is free in ψ or ξ
3. $\varphi = \exists y \psi$ where y is distinct of x and x is free in ψ

a variable that is not free is *bound*

free(φ): the set of free variables of φ

conjunctive calculus query

a query is an expression of the form

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

where φ is a formula, and variables in $(e_1, \ldots, e_n)$ are exactly $\mathit{free}(\varphi)$

example

in

$$\{t|\exists r, a\ (\text{directors}(r,\text{"american"}) \wedge \text{ movies}(t,r,a))\}$$

t is free

r and a are bound

valuation

defined as previously, and written $\{x_1/a_1, \ldots, x_n/a_n\}$

we note $v|_V$ the restriction of v to the set V

valuation

defined as previously, and written $\{x_1/a_1, \ldots, x_n/a_n\}$

we note $v|_V$ the restriction of v to the set V

v is a valuation over V, $x \notin V$, $c \in \mathbf{dom}$, $v \cup \{x/c\}$ is a valuation over $V \cup \{x\}$

- identical to v over V
- associating x with c

satisfaction of a formula

I a database instance *satisfies* a formula φ under valuation v (noted $I \models \varphi[v]$) if

1. $\varphi = R(u)$ is an atom and $v(u) \in I(R)$, or

satisfaction of a formula

I a database instance *satisfies* a formula φ under valuation v (noted $I \models \varphi[v]$) if

1. $\varphi = R(u)$ is an atom and $v(u) \in I(R)$, or
2. $\varphi = (\psi \wedge \xi)$ and $I \models \psi[v|_{free(\psi)}]$ and $I \models \xi[v|_{free(\xi)}]$, or

satisfaction of a formula

I a database instance *satisfies* a formula φ under valuation v (noted $I \models \varphi[v]$) if

1. $\varphi = R(u)$ is an atom and $v(u) \in I(R)$, or
2. $\varphi = (\psi \wedge \xi)$ and $I \models \psi[v|_{free(\psi)}]$ and $I \models \xi[v|_{free(\xi)}]$, or
3. $\varphi = \exists x \psi$ and there exists $c \in dom$, $I \models \psi[v \cup \{x/c\}]$

example

let I be the database instance depicted on slide #1

consider the formula $\varphi = \exists r, a$ (directors(r,"american")$\wedge$ movies(t,r,a))

I satisfies φ under v if v is such that $v(\mathrm{t}) =$ starwars

example

let I be the database instance depicted on slide #1

consider the formula $\varphi = \exists r, a$ (directors(r,"american")$\wedge$ movies(t,r,a))

I satisfies φ under v if v is such that $v(\mathrm{t}) =$ starwars

I does not satisfy φ under v' such that $v'(\mathrm{t}) =$ nikita

image

let $q = \{e_1, \ldots, e_m | \varphi\}$ be a conjunctive query over D and I an instance of D

the image of I by q, noted $q(I)$, is:

$$q(I) = \{v((e_1, \ldots, e_m)) | I \models \varphi[v] \text{ and } v \text{ is a valuation over } \mathit{free}(\varphi)\}$$

example

consider query $q = \{t | \exists r, a \text{ (directors}(r, \text{"american"}) \wedge \text{ movies}(t, r, a))\}$

and let I be the database instance depicted on slide $\#1$

$q(I) = \{(\text{"starwars"}), (\text{"dune"})\}$

properties of conjunctive queries

why studying conjunctive queries ?

- they are simple
- they represent an important part of usual queries
- they have interesting properties

monotony

a query q over D is *monotonous* if for any instance I, J of D :

$I \subseteq J$ implies $q(I) \subseteq q(J)$

example

query q=american_movies(t) ←directors(r,"american"),movies(t,r,a).

example

query q=american_movies(t) $\leftarrow$directors(r,"american"),movies(t,r,a).

I and J: databases instances with

I={movies(starwars,lucas,1977),movies(nikita,besson,1990),
movies(locataires,ki-duk,2005),movies(dune,lynch,1984),
directors(lucas,american),directors(lynch,american),
directors(ki-duk,korean),directors(besson,french)}

example

J={movies(nikita,besson,1990),movies(locataires,ki-duk,2005),
movies(dune,lynch,1984), directors(lynch,american),
directors(ki-duk,korean),directors(besson,french)}

$J \subset I$

example

$$q(I) = \{\text{american\_movies(“starwars”)},\text{american\_movies(“dune”)}\}$$

$$q(J) = \{\text{american\_movies(“dune”)}\}$$

example

$$q(I) = \{\text{american\_movies}(\text{“starwars”}), \text{american\_movies}(\text{“dune”})\}$$

$$q(J) = \{\text{american\_movies}(\text{“dune”})\}$$

$$q(J) \subset q(I)$$

non monotonous queries

example of non monotonous queries:

consider relation actors of schema actors[name,directed_by]

who are the actors who were directed only by "lucas"?

non monotonous queries

example of non monotonous queries:

consider relation actors of schema actors[name,directed_by]

who are the actors who were directed only by "lucas"?

$I(\text{actors}) = \{(\text{ford,lucas}),(\text{ford,spielberg})\}$, $q(I) = \emptyset$

$J(\text{actors}) = \{(\text{ford,lucas})\}$, $q(J) = \{\text{ford}\}$

satisfiability

a query q is *satisfiable* if there exists an instance I such that $q(I)$ is non empty

satisfiability

a query q is *satisfiable* if there exists an instance I such that $q(I)$ is non empty

example of unsatisfiable query:

is there a movie called “starwars” and “dune”?

properties of conjunctive queries

theorem:

conjunctive queries are monotonous and satisfiable

properties of conjunctive queries

theorem:

conjunctive queries are monotonous and satisfiable

demonstration left as an exercise...

properties of conjunctive queries

any conjunctive query q can be written under the form

$$\{e_1, \ldots, e_m | \exists x_1, \ldots, x_p (R_1(u_1) \wedge \ldots \wedge R_n(u_n))\}$$

evaluating q over an instance I just needs constants in $adom(q, I)$

properties of conjunctive queries

let $q = \{u|\varphi\}$ and $q' = \{w|\psi\}$ be conjunctive queries

with $free(q) = free(q')$

q and q' are *equivalent* ($q \equiv q'$) if

for any I and v, $I \models \varphi[v] \iff I \models \psi[v]$

example

$\{x|\exists y, z \text{ movies}(y, x, z) \wedge \text{directors}(x,\text{korean})\ \}$

and

$\{a|\exists b, c \text{ directors}(a,\text{korean}) \wedge \text{movies}(b, a, c)\ \}$

are two equivalent queries

properties of conjunctive queries

for conjunctive queries, the rule-based language Q_1 and the conjunctive calculus Q_2 are equivalent

they can express *exactly* the same queries

formally:

$$\forall q_1 \in Q_1, \exists q_2 \in Q_2, q_1 \equiv q_2$$

$$\forall q_1 \in Q_2, \exists q_2 \in Q_1, q_1 \equiv q_2$$

query composition

a *conjunctive program* P on a database D is a sequence of conjunctive queries

$S_1(u_1) \leftarrow body_1$
$S_2(u_2) \leftarrow body_2$
$\ldots$
$S_n(u_n) \leftarrow body_n$

where the S_i's are pairwise distinct, and are not in D

query composition

the relations that can appear in $body_i$ are

- relations of D and
- $S_1, \ldots, S_{i-1}$

any conjunctive program can be written under the form of a single rule

example

the program

$$S(x,y) \leftarrow R(x,y), Q(y).$$

$$T(y) \leftarrow Q(x), S(x,y).$$

$$U(x,y) \leftarrow T(x), R(x,y).$$

can be written

$$U(x,y) \leftarrow R(x,y), Q(z), R(z,x), Q(x).$$

closure by composition

theorem:

the composition of conjunctive queries is a conjunctive query