Database Theory and Knowledge Representation

3rd Lecture

David Carral

University of Montpellier

October 13, 2023

Announcements

1. Exam in two weeks!
   - Includes content from Marie-Laure's part.

2. Master's topics available at Moodle

Summary and Outlook

We have covered the following topics:

- The Relational Calculus: RA and FO Queries
- Complexity of Query Answering

Future Content:

- Query expressivity: Comparing RA and FO Queries
- Tractable Query Entailment
- Questions about the exam?

Equivalent Queries

The same query can be expressed with different languages:

Example

The query mapping

Who is the director of “The Imitation Game”?

can be expressed using the relational algebra

$$\pi_{Director}(\sigma_{Title=\text{“The Imitation Game”}}(Films))$$

or an FO query

$$\exists y_A.\ \mathrm{Films}(\texttt{"The Imitation Game"}, x_D, y_A)[x_D].$$

How to Compare Query Languages

We have studied two different query languages
$\rightsquigarrow$ how to compare them?

Definition

The set of query mappings that can be described in a query language L is denoted $\mathbf{QM}(\mathsf{L})$.

- L_1 is subsumed by L_2, written $\mathsf{L}_1 \sqsubseteq \mathsf{L}_2$, if $\mathbf{QM}(\mathsf{L}_1) \subseteq \mathbf{QM}(\mathsf{L}_2)$
- L_1 is equivalent to L_2, written $\mathsf{L}_1 \equiv \mathsf{L}_2$, if $\mathbf{QM}(\mathsf{L}_1) = \mathbf{QM}(\mathsf{L}_2)$

Theorem

The following query languages are equivalent:

- Relational algebra (RA)
- First-order queries (FO)

Comparing Query Languages: A Simple Example

Example

Consider the $\mathrm{RA}^{\setminus\cap}$, which is a restricted version of the RA that only allows for the use of $\{\sigma, \pi, \cup, -, \bowtie, \delta\}$. We can show that RA and $\mathrm{RA}^{\setminus\cap}$ are equivalent.

Solution

- Trivial: $\mathrm{RA}^{\setminus\cap}$ is subsumed by the RA.
- To show that RA is subsumed by $\mathrm{RA}^{\setminus\cap}$ note that, given some RA queries q and s:

$$q \cap s \equiv q \bowtie s$$

RA $\sqsubseteq$ FO

Definition

For a given RA query $q[a_1, \ldots, a_n]$, we recursively construct a FO query $\varphi_q[x_{a_1}, \ldots, x_{a_n}]$ as follows:

1. If $q = R$ with signature $R[a_1, \ldots, a_n]$, then $\varphi_q = R(x_{a_1}, \ldots, x_{a_n})$.
2. If $n = 1$ and $q = \{\{a_1 \mapsto c\}\}$, then $\varphi_q = (x_{a_1} \approx c)$.
3. If $q = \sigma_{a_i=c}(q')$, then $\varphi_q = \varphi_{q'} \wedge (x_{a_i} \approx c)$
4. If $q = \sigma_{a_i=a_j}(q')$, then $\varphi_q = \varphi_{q'} \wedge (x_{a_i} \approx x_{a_j})$
5. If $q = \delta_{b_1,\ldots,b_n \to a_1,\ldots,a_n} q'$,[1] then $\varphi_q = \exists x_{b_1}, \ldots, x_{b_n}.(\bigwedge_{1 \le i \le n} x_{a_i} \approx x_{b_i}) \wedge \varphi_{q'}$.

[1] We assume that $\{b_1, \ldots, b_n\} \cap \{a_1, \ldots, a_n\} = \emptyset$ without loss of generality.

RA $\sqsubseteq$ FO (cont'd)

Definition (cont'd)

6. If $q = \pi_{a_1,\ldots,a_n}(q')$ for a subquery $q'[b_1, \ldots, b_m]$, then $\varphi_q = \exists x_{c_1}, \ldots, x_{c_k}.\varphi_{q'}$ where $\{c_1, \ldots, c_k\} = \{b_1, \ldots, b_m\} \setminus \{a_1, \ldots, a_n\}$.
7. If $q = q_1 \bowtie q_2$, then $\varphi_q = \varphi_{q_1} \wedge \varphi_{q_2}$.
8. If $q = q_1 \cup q_2$, then $\varphi_q = \varphi_{q_1} \vee \varphi_{q_2}$.
9. If $q = q_1 - q_2$, then $\varphi_q = \varphi_{q_1} \wedge \neg\varphi_{q_2}$.

Remarks

- We can show that φ_q is equivalent to q via structural induction.
- We have not defined a translation for queries of the form $q \cap s$. Is our proof incomplete?

FO $\sqsubseteq$ RA

To define this direction, we first define a preliminary RA query:

Definition

For a FO query q, a database schema $\mathcal{S}$, and some attribute a; let $Dom^{a}_{\mathcal{S},q}$ be the following RA expression:

$$\left(\bigcup_{R \in \mathsf{Tables}(\mathcal{S})} \bigcup_{b \in \mathsf{Atts}(R)} \delta_{b \to a}(\pi_b(R)) \right) \cup \left\{ \{ a \mapsto c \} \,\middle|\, c \in \mathbf{dom}(q) \right\}.$$

Remark

Note that $Dom^{a}_{\mathcal{S},q}(\mathcal{I}) = \{\{a \mapsto c\} \mid c \in \mathbf{dom}(\mathcal{I}, q)\}$ for any database $\mathcal{I}$ defined over $\mathcal{S}$.

FO $\sqsubseteq$ RA (cont'd)

Definition

Consider an FO query $q = \varphi[x_1, \ldots, x_n]$ that is defined for a database with schema $\mathcal{S}$. For every variable x, we use a fresh attribute name a_x.

- If $\varphi = R(x_1, \ldots, x_m)$,[2] then $E_\varphi = R$ with $R[a_{x_1}, \ldots, a_{x_m}]$.
- If $\varphi = (x \approx c)$, then $E_\varphi = \{\{a_x \mapsto c\}\}$.
- If $\varphi = (x \approx y)$, then $E_\varphi = \sigma_{a_x = a_y}(Dom^{a_x}_{\mathcal{S},\varphi} \bowtie Dom^{a_y}_{\mathcal{S},\varphi})$.
- Other forms of equality atoms are analogous.

[2] Without loss of generality, we assume that all $x_1, \ldots, x_m$ are variables, and that $x_i \neq x_j$ for every $1 \leq i < j \leq m$.

FO $\sqsubseteq$ RA (cont'd)

Definition (cont'd)

- If $\varphi = \neg\psi$, then $E_\varphi = (Dom^{a_{x_1}}_{\mathcal{S},\varphi} \bowtie \ldots \bowtie Dom^{a_{x_n}}_{\mathcal{S},\varphi}) - E_\psi$.
- If $\varphi = \varphi_1 \wedge \varphi_2$, then $E_\varphi = E_{\varphi_1} \bowtie E_{\varphi_2}$.
- If $\varphi = \exists y.\psi$ where φ has free variables $x_1, \ldots, x_n$, then $E_\varphi = \pi_{a_{x_1},\ldots,a_{x_n}} E_\psi$.

Remark

The cases for $\vee$ and $\forall$ can be constructed from the above:

$$E_{\forall y.\psi} \equiv E_{\neg\exists y.\neg\psi} \quad E_{\psi\vee\varphi} \equiv E_{\neg(\neg\psi\wedge\neg\varphi)}$$

Conjunctive Queries

- Problem: answering FO queries is hard.
- Idea: restrict FO queries to conjunctive, positive features

Definition: Conjunctive Queries

A conjunctive query (CQ) is an expression of the form $\exists y_1, \ldots, y_m . A_1 \wedge \ldots \wedge A_\ell$ where each A_i is an atom of the form $\mathsf{R}(t_1, \ldots, t_k)$. In other words, a CQ is an FO query that only uses conjunctions of atoms and (outer) existential quantifiers.

Example: "Find all lines with an accessible stop":

$\exists y_{\mathsf{SID}}, y_{\mathsf{Stop}}, y_{\mathsf{To}} . \mathsf{Stops}(y_{\mathsf{SID}}, y_{\mathsf{Stop}}, \texttt{"true"}) \wedge$
$\mathsf{Connect}(y_{\mathsf{SID}}, y_{\mathsf{To}}, x_{\mathsf{Line}})$

Discuss

Can we express all FO queries as CQs? What is the complexity of BCQ entailment?

Exercises: CQ Examples

Films

Title	Director	Actor
The Imitation Game	Tyldum	Cumberbatch
The Imitation Game	Tyldum	Knightley
...	...	...
Internet's Own Boy	Knappenberger	Swartz
Internet's Own Boy	Knappenberger	Lessig
Internet's Own Boy	Knappenberger	Berners-Lee
...	...	...
Dogma	Smith	Damon
Dogma	Smith	Affleck

Venues

Cinema	Address	Phone
UFA	St. Peter St. 24	4825825
Diagon	King St. 55	8032185
...	...	...

Program

Cinema	Title	Time
Diagon	The Imitation Game	19:30
Diagon	Dogma	20:45
UFA	The Imitation Game	22:45

5. List the pairs of persons such that the first directed the second in a film, and vice versa.

$$\exists y_T, z_T.\ \mathrm{Films}(y_T, x_D, x_A) \wedge \mathrm{Films}(z_T, x_A, x_D)[x_D, x_A]$$

6. List the names of directors who have acted in a film they directed.

$$\exists y_T.\ \mathrm{Films}(y_T, x_D, x_D)[x_D]$$

Exercises: CQ Examples

Films

Title	Director	Actor
The Imitation Game	Tyldum	Cumberbatch
The Imitation Game	Tyldum	Knightley
...	...	...
Internet's Own Boy	Knappenberger	Swartz
Internet's Own Boy	Knappenberger	Lessig
Internet's Own Boy	Knappenberger	Berners-Lee
...	...	...
Dogma	Smith	Damon
Dogma	Smith	Affleck

Venues

Cinema	Address	Phone
UFA	St. Peter St. 24	4825825
Diagon	King St. 55	8032185
...	...	...

Program

Cinema	Title	Time
Diagon	The Imitation Game	19:30
Diagon	Dogma	20:45
UFA	The Imitation Game	22:45

9. Find the actors that are NOT cast in a movie by "Smith."

$$\exists y_T, y_D.\mathrm{Films}(y_T, y_D, x_A) \wedge$$
$$\forall x_T, x_D.(\mathrm{Films}(x_T, x_D, x_A) \to x_D \not\approx \texttt{"Smith"}[x_A]$$

10. Find all pairs of actors who act together in at least one film.

$$\exists y_T, y_D, y'_D.\ \mathrm{Films}(y_T, y_D, x_A) \wedge \mathrm{Films}(y_T, y'_D, x_{A'}) \wedge x_A \not\approx x_{A'}[x_A, x_A$$

Extensions of Conjunctive Queries

Two features are often added:

- Equality: CQs with equality can use atoms of the form $t_1 \approx t_2$ (in relational calculus: table constants)
- Unions: unions of conjunctive queries are called UCQs (in this case the union is only allowed as outermost operator)

Both extensions truly increase expressive power

Features omitted on purpose: negation and universal quantifiers
$\leadsto$ the reason for this is query complexity

Boolean Conjunctive Queries

A Boolean conjunctive query (BCQ) asks for a mapping from query variables to domain elements such that all atoms are true

Example: "Is there an accessible stop where some line departs?"

$$\exists y_{\mathsf{SID}}, y_{\mathsf{Stop}}, y_{\mathsf{To}}, y_{\mathsf{Line}}.\mathsf{Stops}(y_{\mathsf{SID}}, y_{\mathsf{Stop}}, \texttt{"true"}) \wedge \mathsf{Connect}(y_{\mathsf{SID}}, y_{\mathsf{To}}, y_{\mathsf{Line}})$$

Stops:

SID	**Stop**	**Accessible**
17	Hauptbahnhof	true
42	Helmholtzstr.	true
57	Stadtgutstr.	true
123	Gustav-Freytag-Str.	false
...	...	...

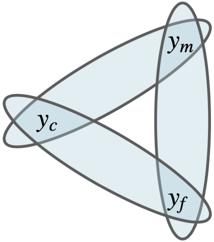
Connect:

From	**To**	**Line**
57	42	85
17	789	3
...	...	...

Example: Cyclic CQs

"Is there a child whose parents are married with each other?"

$$\exists y_c, y_m, y_f.\mathrm{mother}(y_c, y_m) \wedge \mathrm{father}(y_c, y_f) \wedge \mathrm{married}(y_m, y_f)$$

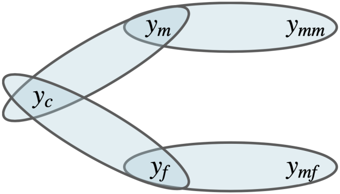


$\leadsto$ cyclic query

Example: Acyclic CQs

"Is there a child whose parents are married with someone?"

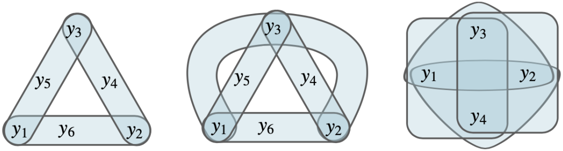
$$\exists y_c, y_m, y_f, y_{mm}, y_{mf}.\text{mother}(y_c, y_m) \wedge \text{father}(y_c, y_f) \wedge \\ \text{married}(y_m, y_{mm}) \wedge \text{married}(y_{mf}, y_f)$$



$\rightsquigarrow$ acyclic query

Defining Acyclic Queries

Queries in general are hypergraphs
⇝ What does "acyclic" mean?



View hypergraphs as graphs to check acyclicity?

- Primal graph: same vertices; edges between each pair of vertices that occur together in a hyperedge
- Incidence graph: vertices and hyperedges as vertices, with edges to mark incidence (bipartite graph)

However: both graphs have cycles in almost all cases

Acyclic Hypergraphs

GYO-reduction algorithm to check acyclicity:
(after Graham [1979] and Yu & Özsoyoğlu [1979])

Input: hypergraph $H = \langle V, E \rangle$ (we don't need relation labels here)
Output: GYO-reduct of H

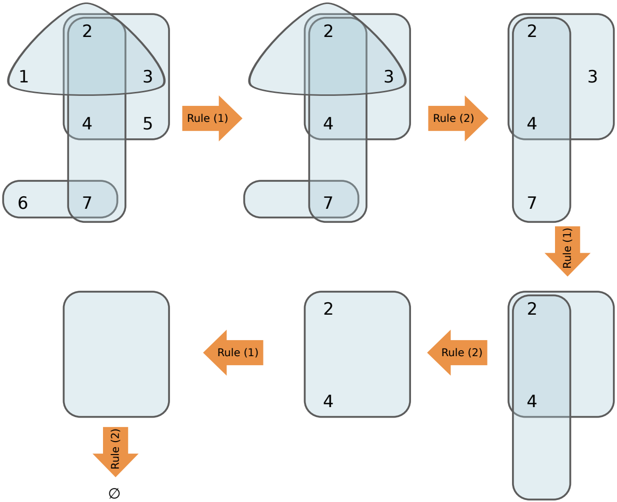
Apply the following simplification rules as long as possible:

(1) Delete all vertices that occur in at most one hyperedge
(2) Delete all hyperedges that are empty or that are contained in other hyperedges

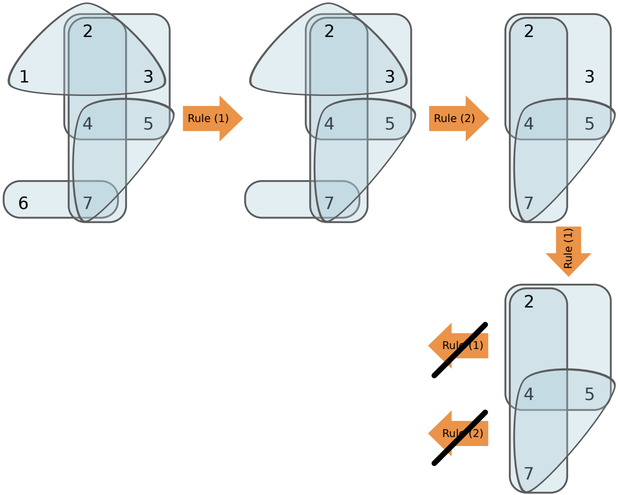
Definition

A hypergraph is acyclic if its GYO-reduct is $\langle \emptyset, \emptyset \rangle$.
A CQ is acyclic if its associated hypergraph is.

Example 1: GYO-Reduction



Example 2: GYO-Reduction



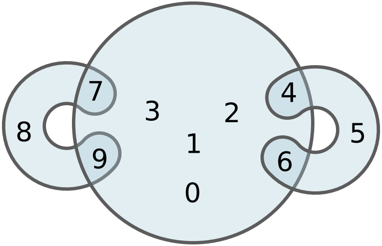
Alternative Version of GYO-Reduction

An ear of a hypergraph $\langle V, E\rangle$ is a hyperedge $e \in E$ that satisfies one of the following:

(1) there is an edge $e' \in E$ such that $e \neq e'$ and every vertex of e is either only in e or also in e', or

(2) e has no intersection with any other hyperedge.

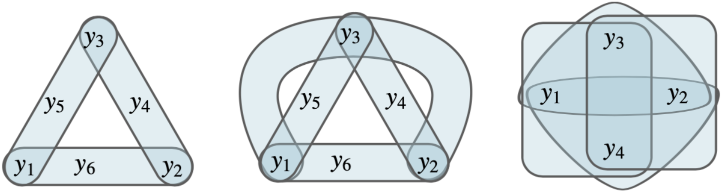
Example:



$\rightsquigarrow$ edges $\langle 4, 5, 6\rangle$ and $\langle 7, 8, 9\rangle$ are ears

Examples

Any ears?



GYO'-Reduction

Definition

Input: hypergraph $H = \langle V, E \rangle$
Output: GYO'-reduct of H

Apply the following simplification rule as long as possible:

- Select an ear e of H
- Delete e
- Delete all vertices that only occurred in e

Theorem

The GYO-reduct is $\langle \emptyset, \emptyset \rangle$ if and only if the GYO'-reduct is $\langle \emptyset, \emptyset \rangle$

$\rightsquigarrow$ alternative characterization of acyclic hypergraphs

Exercise

Decide if the following conjunctive queries are tree queries by applying (one version of) the GYO algorithm.

1. $\exists x, y, z, v.\ r(x, y) \wedge r(y, z) \wedge r(z, v) \wedge s(x, y, z) \wedge s(y, z, v)$
2. $\exists x, y, z, u, v, w.\ r(x, y) \wedge s(x, z, v) \wedge r(u, z) \wedge t(x, v, u, w)$

Definition

Input: hypergraph $H = \langle V, E \rangle$
Output: GYO'-reduct of H

Apply the following simplification rule as long as possible:

- Select an ear e of H
- Delete e
- Delete all vertices that only occurred in e

Join Trees

Both GYO algorithms can be implemented in linear time

Open question: what benefit does BCQ acyclicity give us?

Fact: if a BCQ is acyclic, then it has a join tree

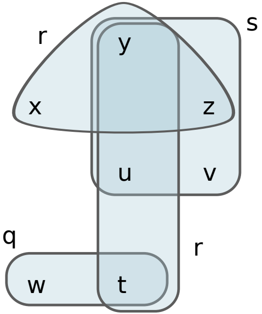
Definition

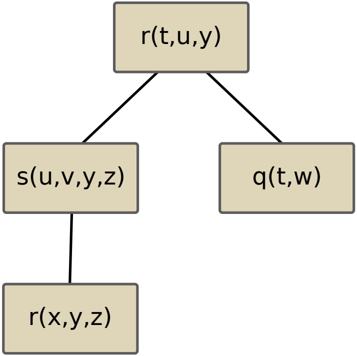
A join tree of a (B)CQ is an arrangement of its query atoms in a tree structure T, such that for each variable x, the atoms that refer to x are a connected subtree of T.

A (B)CQ that has a join tree is called a tree query.

Example: Join Tree

$$\exists x, y, z, t, u, v, w. \big(r(x, y, z) \wedge r(t, u, y) \wedge s(u, v, y, z) \wedge q(t, w) \big)$$





Processing Join Trees Efficiently

Join trees can be processed in polynomial time

Key ingredient: the semijoin operation

Definition

Given two relations $R[U]$ and $S[V]$, the semijoin $R^{\mathcal{I}} \ltimes S^{\mathcal{I}}$ is defined as $\pi_U(R^{\mathcal{I}} \bowtie S^{\mathcal{I}})$.

Join trees can be processed by computing semijoins bottom-up
$\rightsquigarrow$ Yannakakis' Algorithm

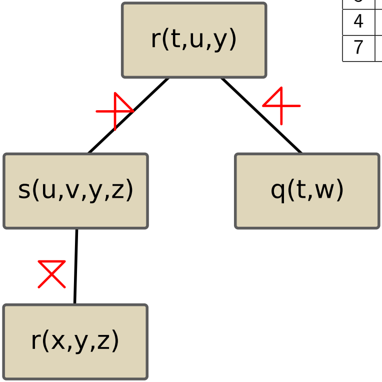
Yannakakis' Algorithm by Example

r:

~~1~~	~~2~~	~~3~~
~~3~~	~~3~~	~~5~~
4	7	3
7	9	7

s:

2	8	3	5
~~2~~	~~4~~	~~4~~	~~6~~
3	4	2	3
7	1	3	5
~~8~~	~~5~~	~~6~~	~~4~~
9	2	7	3



q:

2	3
4	5
4	7
6	5
7	2

r:

1	2	3
3	3	5
4	7	3
7	9	7

Yannakakis' Algorithm: Summary

Polynomial time procedure for answering BCQs

Does not immediately compute answers in the version given here
$\leadsto$ modifications needed

Even tree queries can have exponentially many results,
but each can be computed (not just checked) in P
$\leadsto$ output-polynomial computation of results

Exercise: Yannakakis' Algorithm

Solve the following combinatorial crossword puzzle using Yannakakis' algorithm (in spirit). Specify the join tree that you are using.

x_1	x_2	x_3	x_4	x_5	x_6	x_7
x_8		x_9				x_{10}
x_{11}		x_{12}		x_{13}	x_{14}	x_{15}
x_{16}		x_{17}				x_{18}
x_{19}		x_{20}		x_{21}	x_{22}	x_{23}

1 hor.:

B	R	I	S	T	O	L
C	A	R	A	M	E	L
P	H	A	R	A	O	H
S	P	I	N	A	C	H
T	S	U	N	A	M	I

1 vert.:

C	L	E	A	R
H	U	M	A	N
P	E	A	C	E
S	H	A	R	K
T	I	G	E	R

3 vert.:

H	A	P	P	Y
I	N	F	E	R
L	A	B	O	R
L	A	T	E	R
U	N	T	I	L

7 vert.:

H	E	A	R	T
H	O	N	E	Y
I	R	O	N	Y
L	O	G	I	C
M	A	G	I	C

13 hor.:

A	N	D
C	A	T
D	I	M
L	A	G
W	I	N

21 hor.:

A	R	C
F	E	E
L	O	W
T	W	O
W	A	Y

Exercise: Acyclicity and Constants

How can we deal with BCQs that feature constants? E.g.,

$$\exists x, y, z.\mathsf{mother}(x, y) \land \mathsf{father}(x, z) \land \mathsf{bornIn}(y, \texttt{"Montpellier"}) \land \\ \mathsf{bornIn}(z, \texttt{"Montpellier"})$$

Discussion

Is the above query acyclic? Can we solve it in polynomial time?

Summary and Outlook

We have covered the following topics:

- The relational calculus: RA and FO-queries are equivalent
- Acyclic Boolean Conjunctive Queries: Tractability

Future Content:

- Partial Exam