Foundations of Data and Knowledge Bases

Joachim Niehren

Links: Linking Dynamic Data
Inria Lille

November 10, 2020

Outline

Relational Databases

Applications

Information systems (SAP)	Online shops
Banking systems	Travel booking
Music store	Hotel booking
...	...

SQL

programming language: relations as values
programming systems: Postgres, Oracle

References

1. Database Systems: The Complete Book. By H. Garcia-Molina, J. Ullman, and J. Widom. 2001 (1000 pages)
2. Foundation of Database: S. Abiteboul, R. Hull, V. Vianu. 1994 (400 pages)

Example: Movie Database

Movies	*Title*	*Director*	*Actor*
	The Trouble with Harry	Hitchcock	Gwenn
	The Trouble with Harry	Hitchcock	Forsythe
	The Trouble with Harry	Hitchcock	MacLaine
	The Trouble with Harry	Hitchcock	Hitchcock
	...	...	...
	Cries and Whispers	Bergman	Andersson
	Cries and Whispers	Bergman	Sylwan
	Cries and Whispers	Bergman	Thulin
	Cries and Whispers	Bergman	Ullman

Location	*Theater*	*Address*	*Phone Number*
	Gaumont Opéra	31 bd. des Italiens	47 42 60 33
	Saint André des Arts	30 rue Saint André des Arts	43 26 48 18
	Le Champo	51 rue des Ecoles	43 54 51 60
	...	...	...
	Georges V	144 av. des Champs-Elysées	45 62 41 46
	Les 7 Montparnassiens	98 bd. du Montparnasse	43 20 32 20

Pariscope	*Theater*	*Title*	*Schedule*
	Gaumont Opéra	Cries and Whispers	20:30
	Saint André des Arts	The Trouble with Harry	20:15
	Georges V	Cries and Whispers	22:15
	...	...	...

SQL Aggregate Query

```
select
        Movies.title ,  Location.address
from
        Movies ,  Location ,  Pariscope
where
        Movies.director    =  ''Bergman''
  and   Movies.title       =  Pariscope.title
  and   Pariscope.theatre  =  Location.theater
```

SQL Aggregate Query

```
select
        Movies.title, COUNT(Location.address)
from
        Movies, Location, Pariscope
where
        Movies.director    = ''Bergman''
  and   Movies.title       = Pariscope.title
  and   Pariscope.theatre = Location.theater
GROUPED BY Movies.title
```

Database Schema Σ

Relational Vocabulary

$$\Sigma = \{R_1[A_1], \ldots, R_n[A_n]\} \cup \mathrm{Consts}$$

where

$R_1, \ldots, R_n$ relation symbols
$A_1, \ldots, A_n$ finite subsets of attributes
Consts set of constants

Movie Example

$$\begin{aligned} \Sigma \quad &= \{ \quad Movies[Title, Director, Actor] \\ & \qquad Location[Theatre, Address, 'Phone\ Number'] \\ & \qquad Pariscope[Theatre, Title, Schedule]\} \uplus \mathrm{Consts} \\ \mathrm{Consts} \quad &= \quad UTF8^* \end{aligned}$$

Database (Instance)

Tuples and Relations with Attributes A

$\tau : A \to D$	tuples with values in D
$r \subseteq D^A$	relation with values in D

Finite Σ-Structure

$$\mathbf{D} = (D, I)$$

where

D domain is a finite set of data values (strings, integers, floats)
$I(R[A]) \subseteq D^A$ interpretation as relation for all $R[A] \in \Sigma$
$I(c) \in D$ interpreation in domain

Relational Algebra on D

Operators on relations $r \subseteq D^A$ and $r' \subseteq D^{A'}$:

Join (like Cartesian product)

$$r \bowtie r' = \{\tau \cup \tau' \mid \tau \in r, \tau' \in r'\} \qquad \text{if } A \cap A' = \emptyset$$

Union and Differences of Relations

$$r \cup r' \qquad \text{and} \qquad r \setminus r' \qquad \text{if } A = A'$$

Relational Algebra on D

Operators on relations $r \subseteq D^A$ and $r' \subseteq D^{A'}$:

Projection

$$\pi_{A'}(r) = \{\tau_{|A'} \mid \tau \in r\} \qquad \text{if } A' \subseteq A$$

Selection by Constraints

$$\sigma_{a=d}(r) = \{\tau \mid \tau(a) = d, \tau \in r\} \quad \text{where } d \in D$$
$$\sigma_{a=a'}(r) = \{\tau \mid \tau(a) = \tau(a'), \tau \in r\}$$

where

$a, a' \in A$ and $c \in \mathrm{Consts}$

Renaming of Attributes

$$\rho_\theta(r) = \{\tau \circ \theta \mid \tau \in r\} \text{ where } \theta : A \to A' \text{ bijection}$$

Database Queries

Expressions Q build from Σ and the operators of relational algebra:

Q, Q'	$::=$	$R[A]$	where $R[A] \in \Sigma$
	$\mid$	$all[A]$	for finite subsets A of attributes
	$\mid$	$Q \bowtie Q'$	join
	$\mid$	$Q \setminus Q'$	difference
	$\mid$	$Q \cup Q'$	union
	$\mid$	$\pi_A(Q)$	for finite subsets A of attributes
	$\mid$	$\sigma_C(Q)$	for constraints C of form $a = a'$ or $a = c$ where $c \in$
	$\mid$	$\rho_\theta(Q)$	for bijections $\theta : A \to A'$

Well-Typed Queries

The type of a relations is the set of its attributes.

$$\frac{R[A] \in \Sigma}{R[A] : A} \qquad \frac{Q : A \qquad Q' : A' \qquad A \cap A' = \emptyset}{Q \bowtie Q' : A \cup A'}$$

$$\frac{\mathit{true}}{\mathit{all}[A] : A} \qquad \frac{Q : A \qquad Q' : A}{Q \cup Q' : A} \qquad \ldots$$

Only well-typed queries are permitted.

Answer Sets

Any query Q defines set of tuples for each database $\mathbf{D} = (D, I)$ with schema Σ:

$$[\![Q]\!]^{\mathbf{D}} \subseteq D^A \text{ if } Q : A$$

By homorphic interpretation in the relational algebra over D:

$$\begin{array}{l}
[\![R[A]]\!]^{\mathbf{D}} = I(R[A]) \\
[\![all[A]]\!]^{\mathbf{D}} = D^A = \{\tau \mid \tau : A \to D\} \\
[\![Q \bowtie Q']\!]^{\mathbf{D}} = [\![Q]\!]^{\mathbf{D}} \bowtie [\![Q']\!]^{\mathbf{D}} \\
[\![Q \setminus Q']\!]^{\mathbf{D}} = [\![Q]\!]^{\mathbf{D}} \setminus [\![Q']\!]^{\mathbf{D}} \\
[\![\pi_{A'}(Q)]\!]^{\mathbf{D}} = \pi_{A'}([\![Q]\!]^{\mathbf{D}}) \\
[\![\sigma_{a=a'}(Q)]\!]^{\mathbf{D}} = \sigma_{a=a'}([\![Q]\!]^{\mathbf{D}}) \\
[\![\sigma_{a=c}(Q)]\!]^{\mathbf{D}} = \sigma_{a=I(c)}([\![Q]\!]^{\mathbf{D}}) \\
[\![\rho_{\theta}(Q)]\!]^{\mathbf{D}} = \rho_{\theta}([\![Q]\!]^{\mathbf{D}})
\end{array}$$

SQL Queries to Relational Algebra Queries

Reconsider Example Query

```
select  Movies.title ,  Location.address
from    Movies,   Location ,  Pariscope
where   Movies.director    =  ''Bergman''
  and   Movies.title       =  Pariscope.title
  and   Pariscope.theatre  =  Location.theater
```

Corresponding Relational Algebra Query

$$
\begin{array}{l}
\pi_{\{MTitle, Address\}}(\\
\quad \sigma_{Director='Bergman'}(\\
\quad\quad \sigma_{MTitle=PTitle}(\\
\quad\quad\quad \sigma_{PTheatre=LTheater}(\\
\quad\quad\quad\quad \rho_{[Title/MTitle]}(Movies[Title, Director, Actor]) \bowtie \\
\quad\quad\quad\quad \rho_{[Theatre/LTheater]}(Location[Theatre, Address, 'Phone\ Number'] \bowtie \\
\quad\quad\quad\quad \rho_{[Theatre/PTheater, Title/PTitle]}(Pariscope[Theatre, Title, Schedule]]) \\
\quad\quad\quad\quad))))
\end{array}
$$

Exercises

Define $Q \cap Q'$ where $Q : A$ and $Q' : A$ in the relational algebra.

Exercises

Define $Q \cap Q'$ where $Q : A$ and $Q' : A$ in the relational algebra.

$$Q \cap Q' = \overline{\overline{Q} \cup \overline{Q'}}$$

where

$$\overline{Q''} = all[A] \setminus Q'' \text{ for all } Q'' : A$$

Exercises

Define $Q \cap Q'$ where $Q : A$ and $Q' : A$ in the relational algebra.

$$Q \cap Q' = \overline{\overline{Q} \cup \overline{Q'}}$$

where

$$\overline{Q''} = all[A] \setminus Q'' \text{ for all } Q'' : A$$

Can you define it without using union and difference?

Exercises

Define $Q \cap Q'$ where $Q : A$ and $Q' : A$ in the relational algebra.

$$Q \cap Q' = \overline{\overline{Q} \cup \overline{Q'}}$$

where

$$\overline{Q''} = all[A] \setminus Q'' \text{ for all } Q'' : A$$

Can you define it without using union and difference?
for an arbitrary bijection $\theta : \{a_1, \ldots, a_n\} \to A$:

$$\pi_A(\sigma_{a_1=\theta(a_1)} \ldots (\sigma_{a_n=\theta(a_n)}(Q \bowtie \rho_\theta(Q')) \ldots))$$

Exercises

Define $Q \cap Q'$ where $Q : A$ and $Q' : A$ in the relational algebra.

$$Q \cap Q' = \overline{\overline{Q} \cup \overline{Q'}}$$

where

$$\overline{Q''} = all[A] \setminus Q'' \text{ for all } Q'' : A$$

Can you define it without using union and difference?
for an arbitrary bijection $\theta : \{a_1, \ldots, a_n\} \to A$:

$$\pi_A(\sigma_{a_1=\theta(a_1)} \ldots (\sigma_{a_n=\theta(a_n)}(Q \bowtie \rho_\theta(Q')) \ldots))$$

So, union is redundant in the relational algebra!

$$Q \cup Q' = \overline{\overline{Q} \cap \overline{Q'}}$$

Outline

Syntax

FO Σ-Formulas=FO Queries

We use attributes as variables, so let $\mathrm{EleVars}$ be the set of attributes.

$$\phi, \phi' ::= R(a_1{:}x_1, \ldots, a_n{:}x_n) \mid x = y \mid x = c \mid \phi \wedge \phi' \mid \neg\phi \mid \exists x.\phi$$

where $R[a_1, \ldots, a_n] \in \Sigma$ relation symbols, $c \in \Sigma$ constants, and $x, y, x_1, \ldots, x_n \in \mathrm{EleVars}$ variables.

We will sometimes write $R(x_1, \ldots, x_n)$ instead of $R(a_1{:}x_1, \ldots, a_n{:}x_n)$ if attributes are totally ordered and $a_1 < \ldots < a_n$.

Conjunctive Queries

FO Σ-formulas with conjunction and existential quantification, but without negation.

Semantics

Compile FO queries to relational algebra queries.

$$
\begin{aligned}
raq(R(a_1{:}x_1, \dots, a_n{:}x_n)) &= \rho_{[a_1/x_1 \dots a_n/x_n]}(R[A]) \\
raq(\phi \wedge \phi') &= raq(\phi) \ltimes all[V(\phi') \setminus V(\phi)] \\
&\qquad \cap\ raq(\phi') \ltimes all[V(\phi) \setminus V(\phi')] \\
raq(x = y) &= \sigma_{x=y}(all[x, y]) \\
raq(x = c) &= \sigma_{x=c}(all[x]) \\
raq(\exists x.\phi) &= \pi_{V(\phi)\setminus\{x\}}(raq(\phi)) \\
raq(\neg\phi) &= all[V(\phi)] \setminus raq(\phi)
\end{aligned}
$$

FO Queries = Relational Algebra Queries

Theorem

Any conjunctive query can be converted into a relational algebra query without difference in polynomial time, and vice versa.

Proof.

"$\Rightarrow$" By above compiler, and the fact that intersection can be expressed without difference.
"$\Leftarrow$": An exercise

$\square$

FO Queries = Relational Algebra Queries

Theorem

Any conjunctive query can be converted into a relational algebra query without difference in polynomial time, and vice versa.

Proof.

"$\Rightarrow$" By above compiler, and the fact that intersection can be expressed without difference.

"$\Leftarrow$": An exercise

$$
\begin{aligned}
&foq(R[A]) = R(a_1{:}a_1, \ldots, a_n{:}a_n)\\
&foq(Q \bowtie Q') = foq(Q) \wedge foq(Q')\\
&foq(\sigma_{a=c}(Q)) = foq(Q) \wedge a = c\\
&foq(\sigma_{a=a'}(Q)) = foq(Q) \wedge a = a'\\
&foq(\pi_{A'}(Q)) = \exists A''.foq(Q) \text{ where } Q : A \text{ and } A'' = A \setminus A'\\
&foq(\rho_\theta(Q)) = foq(Q)\theta
\end{aligned}
$$

$\square$

Outline

1. Relational Databases
   - SQL
   - Relational Algebra
   - FO Logic
   - Hardness of Database Problems
   - Acyclic Conjunctive Queries
2. Yannakakis' Algorithm
   - Generalization to Counting
   - Boolean Circuits: dDNNFs
   - Quantifier-free ACQs to dDNNFs
3. Dependency Weighted Aggregation
   - Motivation in Data Mining
   - Dependency Weighted Aggregation for Database Queries
   - Computation for dDNNFs

Query Answering

Input

database + query

Output

1. existence of answers
2. the answer set
3. the number of answers

Hardness

Theorem

Existence of answers for conjunctive queries is NP-complete

Proof.

Consider the signature $\Sigma = \{and[1,2,3], or[1,2,3], not[1,2], 0, 1\}$ and consider the Σ-structure with the truth tables of the Boolean functions as database. Any SAT formula can the be encoded into a conjunctive Σ-formula in polynomial time.

$$(x \vee y) \wedge z \quad \text{becomes} \quad \exists o.\ and(o, z, 1) \wedge or(x, y, o)$$

$\square$

Corollary

Existence of answers for select-from-where SQL queries is NP-complete.

Ahhrrrg, our most basic SQL example is this class!

Outline

1. Relational Databases
   - SQL
   - Relational Algebra
   - FO Logic
   - Hardness of Database Problems
   - Acyclic Conjunctive Queries
2. Yannakakis' Algorithm
   - Generalization to Counting
   - Boolean Circuits: dDNNFs
   - Quantifier-free ACQs to dDNNFs
3. Dependency Weighted Aggregation
   - Motivation in Data Mining
   - Dependency Weighted Aggregation for Database Queries
   - Computation for dDNNFs

ACQs

For Binary Signatures

The undirect graph G of the conjunctive query ϕ is acyclic.

G has edge $\{x, y\}$ iff $R(x, y)$ is in ϕ for some binary R

Examples

$R(x, y) \wedge R(y, z)$ is an ACQ but not $R(x, y) \wedge R(y, z) \wedge R(z, x)$.

ACQs

For General Signatures

More complicated ...: α-acyclicity, β-acyclicity, γ-acyclicity ... not today.

Example

The SQL select-from-where example query corresponds to the following conjunctive query, which is α-acyclic :

$$\exists x_{theater}. \quad Movies(x_{title}, 'Bergman', \_) \wedge Pariscope(x_{theater}, x_{title}, \_) \wedge Location(x_{theater}, x_{address}, \_)$$

ACQ's are Feasible (Mostly)

Theorem (Yannakakis Generalization (Capelli, Pichler 2016))

The number of answers of an ACQ Q without quantifiers on a database $\mathbf{D}$ can be computed in polynomial time.

Corollary (Yannakakis 1981)

The existence of answers for an ACQ Q on a database $\mathbf{D}$ can be decided in polynomial time.

Theorem (Mengel/Durand 2015)

Computing the number of answers of a general ACQ Q on a database $\mathbf{D}$ is $\sharp P$-hard.

Bounded Hypertree Width

Theorem

For any class of conjunctive querys of bounded hypertree width C, *there exists polynomial time compiler mapping any pair* $(Q, \mathbf{D})$ *with* $Q \in C$ *to a pair* $(Q', \mathbf{D}')$ *such that* $Q' \in ACQ$ *and* $[\![Q]\!]^{\mathbf{D}} = [\![Q']\!]^{\mathbf{D}'}$.

For instance, $R(x, y) \wedge R(y, z) \wedge R(z, x)$ has tree width 2. Given a database $\mathbf{D}$ it can be rewritten to the ACQ $R(x, y) \wedge T(y, z, x)$ where T is interpeted in $\mathbf{D}'$ as $[\![R(y, z) \wedge R(z, x)]\!]^{\mathbf{D}}$.

Outline

Counting

ACQ

$$R(x,y) \wedge S(x,z) \wedge T(y,t)$$

Database

R	x	y	#
	1	0	n_4
	1	1	n_5

S	x	z	#
	1	0	n_0
	1	1	n_1

T	y	t	#
	0	0	n_2
	1	1	n_3

Computation

$n_0 = n_1 = n_2 = n_3 = 1$

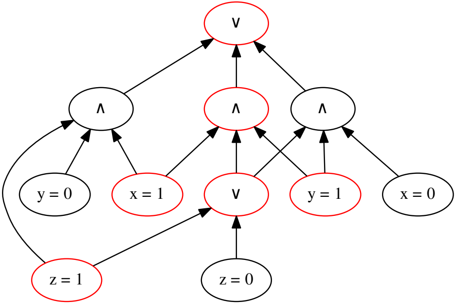
$n_4 = (n_0 + n_1) \cdot n_2 = 2$

$n_5 = (n_0 + n_1) \cdot n_3 = 2$

$count = n_4 + n_5 = 4$

Outline

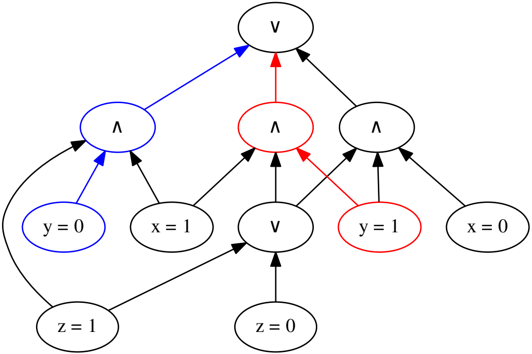
Boolean Circuits



x	y	z
0	1	0
0	1	1
1	0	1
1	1	0
1	1	1

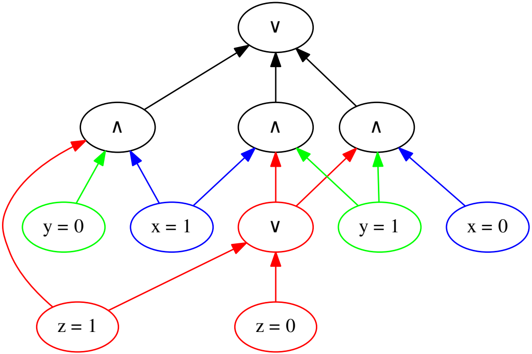
Deterministic Decomposable Boolean Circuits (dDNNFs)

- **deterministic disjunction**



Deterministic Decomposable Boolean Circuits (dDNNFs)

- deterministic disjunction
- **decomposable conjunction**



Outline

1. Relational Databases
   - SQL
   - Relational Algebra
   - FO Logic
   - Hardness of Database Problems
   - Acyclic Conjunctive Queries
2. Yannakakis' Algorithm
   - Generalization to Counting
   - Boolean Circuits: dDNNFs
   - Quantifier-free ACQs to dDNNFs
3. Dependency Weighted Aggregation
   - Motivation in Data Mining
   - Dependency Weighted Aggregation for Database Queries
   - Computation for dDNNFs

Yannakakis Fully Generalized

ACQ

$$Q(x, y, z, t) = R(x, y) \wedge S(x, z) \wedge T(y, t)$$

Database

R	x	y	C
	1	0	g_4
	1	1	g_5

S	x	z	C
	1	0	g_0
	1	1	g_1

T	y	t	C
	0	0	g_2
	1	1	g_3

Computation

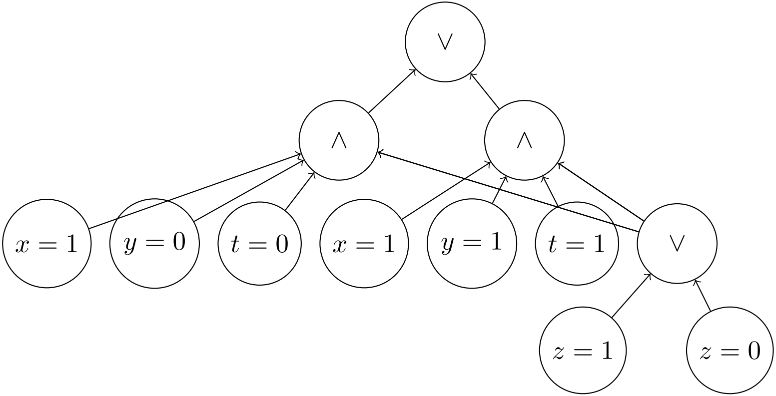
$g_0 = (z = 0) \qquad g_1 = (z = 1) \qquad g_2 = (t = 0) \qquad g_3 = (t = 1)$

$g_4 = (g_0 \vee g_1) \wedge g_2 \wedge (x = 1) \wedge (y = 0)$

$g_5 = (g_0 \vee g_1) \wedge g_3 \wedge (x = 1) \wedge (y = 0)$

$\text{dDNNF} = g_4 \vee g_5$

The Result dDNNF



Counting based on the dDNNF

Evaluate circuit with interpretation in semiring:

leafs	1
$\wedge$	$\cdot$
$\vee$	$+$

Outline

Outline

1. Relational Databases
   - SQL
   - Relational Algebra
   - FO Logic
   - Hardness of Database Problems
   - Acyclic Conjunctive Queries
2. Yannakakis' Algorithm
   - Generalization to Counting
   - Boolean Circuits: dDNNFs
   - Quantifier-free ACQs to dDNNFs
3. Dependency Weighted Aggregation
   - Motivation in Data Mining
   - Dependency Weighted Aggregation for Database Queries
   - Computation for dDNNFs

Averages Approximate Random Values

Id	**Spectator**	**Movie**	**Rating**
0	Argan	Ça commence aujourd'hui	3
1	Argan	Le monocle rit jaune	4
2	Argan	Les barbouzes	2
3	Argan	People : Jet Set 2	1
4	Cyrano	Les barbouzes	8
5	Cyrano	People : Jet Set 2	3
6	Frida	Le monocle rit jaune	9
7	Frida	People : Jet Set 2	2

On the available sample the average rating is:

$$\frac{3+4+2+1+8+3+9+2}{8} = 4.$$

For samples of independent observations, probability theory says that the avarage rating of the sample converges to the estimated average rating of the random variable.

Dependent Observations

However, the ratings in the sample are rarely independent:

Id	Spectator	Movie	Rating
0	Argan	Ça commence aujourd'hui	3
1	Argan	Le monocle rit jaune	4
2	Argan	Les barbouzes	2
3	Argan	People : Jet Set 2	1
4	Cyrano	Les barbouzes	8
5	Cyrano	People : Jet Set 2	3
6	Frida	Le monocle rit jaune	9
7	Frida	People : Jet Set 2	2

How can we deal with this?

Compensating Dependencies for One Attribute

Id	**Spectator**	**Movie**	**Rating**
0	Argan	Ça commence aujourd'hui	3
1	Argan	Le monocle rit jaune	4
2	Argan	Les barbouzes	2
3	Argan	People : Jet Set 2	1
4	Cyrano	Les barbouzes	8
5	Cyrano	People : Jet Set 2	3
6	Frida	Le monocle rit jaune	9
7	Frida	People : Jet Set 2	2

Compensating Dependencies for One Attribute

Id	Spectator	Movie	Rating
0	Argan	Ça commence aujourd'hui	3
1	Argan	Le monocle rit jaune	4
2	Argan	Les barbouzes	2
3	Argan	People : Jet Set 2	1
4	Cyrano	Les barbouzes	8
5	Cyrano	People : Jet Set 2	3
6	Frida	Le monocle rit jaune	9
7	Frida	People : Jet Set 2	2

$$\begin{cases} W_0 + W_1 + W_2 + W_3 & \leq 1 \qquad \text{(Argan)} \\ W_4 + W_5 & \leq 1 \qquad \text{(Cyrano)} \\ W_6 + W_7 & \leq 1 \qquad \text{(Frida)} \end{cases}$$

Compensating Dependencies for One Attribute

Id	**Spectator**	**Movie**	**Rating**	W
0	Argan	Ça commence aujourd'hui	3	1/4
1	Argan	Le monocle rit jaune	4	1/4
2	Argan	Les barbouzes	2	1/4
3	Argan	People : Jet Set 2	1	1/4
4	Cyrano	Les barbouzes	8	1/2
5	Cyrano	People : Jet Set 2	3	1/2
6	Frida	Le monocle rit jaune	9	1/2
7	Frida	People : Jet Set 2	2	1/2

$$\begin{cases} W_0 + W_1 + W_2 + W_3 & \leq 1 \qquad \text{(Argan)} \\ W_4 + W_5 & \leq 1 \qquad \text{(Cyrano)} \\ W_6 + W_7 & \leq 1 \qquad \text{(Frida)} \end{cases}$$

Compensating Dependencies for One Attribute

Id	**Spectator**	**Movie**	**Rating**	W
0	Argan	Ça commence aujourd'hui	3	1/4
1	Argan	Le monocle rit jaune	4	1/4
2	Argan	Les barbouzes	2	1/4
3	Argan	People : Jet Set 2	1	1/4
4	Cyrano	Les barbouzes	8	1/2
5	Cyrano	People : Jet Set 2	3	1/2
6	Frida	Le monocle rit jaune	9	1/2
7	Frida	People : Jet Set 2	2	1/2

This yields a weighted average: $\frac{0.25 \cdot (3+4+2+1) + 0.5 \cdot (8+3+9+2)}{3} = 4.5$.

Compensating Dependencies on Both Attributes

Id	**Spectator**	**Movie**	**Rating**
0	Argan	Ça commence aujourd'hui	3
1	Argan	Le monocle rit jaune	4
2	Argan	Les barbouzes	2
3	Argan	People : Jet Set 2	1
4	Cyrano	Les barbouzes	8
5	Cyrano	People : Jet Set 2	3
6	Frida	Le monocle rit jaune	9
7	Frida	People : Jet Set 2	2

Compensating Dependencies on Both Attributes

Id	Spectator	Movie	Rating
0	Argan	Ça commence aujourd'hui	3
1	Argan	Le monocle rit jaune	4
2	Argan	Les barbouzes	2
3	Argan	People : Jet Set 2	1
4	Cyrano	Les barbouzes	8
5	Cyrano	People : Jet Set 2	3
6	Frida	Le monocle rit jaune	9
7	Frida	People : Jet Set 2	2

$$\begin{cases} W_0 + W_1 + W_2 + W_3 & \leq 1 \qquad \text{(Argan)} \\ \ldots & \\ W_3 + W_5 + W_7 & \leq 1 \qquad \text{(People : Jet Set 2)} \end{cases}$$

Compensating Dependencies on Both Attributes

Id	**Spectator**	**Movie**	**Rating**	W
0	Argan	Ça commence aujourd'hui	3	1/2
1	Argan	Le monocle rit jaune	4	1/6
2	Argan	Les barbouzes	2	1/6
3	Argan	People : Jet Set 2	1	1/6
4	Cyrano	Les barbouzes	8	5/6
5	Cyrano	People : Jet Set 2	3	1/6
6	Frida	Le monocle rit jaune	9	5/6
7	Frida	People : Jet Set 2	2	1/6

$$\begin{cases} W_0 + W_1 + W_2 + W_3 & \leq 1 \qquad \text{(Argan)} \\ \ldots & \\ W_3 + W_5 + W_7 & \leq 1 \qquad \text{(People : Jet Set 2)} \end{cases}$$

Compensating Dependencies on Both Attributes

Id	**Spectator**	**Movie**	**Rating**	W
0	Argan	Ça commence aujourd'hui	3	1/2
1	Argan	Le monocle rit jaune	4	1/6
2	Argan	Les barbouzes	2	1/6
3	Argan	People : Jet Set 2	1	1/6
4	Cyrano	Les barbouzes	8	5/6
5	Cyrano	People : Jet Set 2	3	1/6
6	Frida	Le monocle rit jaune	9	5/6
7	Frida	People : Jet Set 2	2	1/6

This yields a weighted average: $\frac{1/2 \cdot 3 + 1/6 \cdot (4+2+1+3+2) + 5/6 \cdot (8+9)}{3} \simeq 5.9$.

Dependency Weighted Aggregates

Fractional Matching Number $fmn(S)$

Maximal value of $W_1 + \ldots + W_n$ subject to the linear constraints.

Dependency Weighted Aggregate $dwa(S)$

$$\sum_{i=1}^{7} W_i Rating_i$$

such that $W_1 + \ldots + W_n$ maximal value subject to the linear constraints.

Theorem (Ramon et. al 2013)

Dependency weighted aggegates of samples $dwa(S)$ converges to estimated rating when S grows.

Linear Programs

The fractional matching number is defined by a linear program of polynomial size in of sample S.

Lemma

$fmn(S)$ can be computed in time polynomial in the size S and also $dwa(S)$.

Question

What happens if we are given a query Q and a database D such that $S = Q(\mathbf{D})$?

Lemma

$fmn(S)$ has non-zero value $\iff S \neq \emptyset$

Proposition

For conjunctive queries Q on databases $\mathbf{D}$, $fmn(Q(\mathbf{D}))$ cannot be computed in polynomial time.

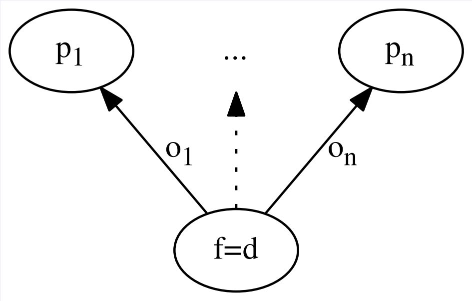
Theorem (Capelli, Crosetti, Niehren, Ramon 2019)

For ACQs Q on databases $\mathbf{D}$, $fmn(Q(\mathbf{D}))$ and $dwa(Q(\mathbf{D}))$ can be computed in polynomial time.

Use the dDNNF representing $Q(\mathbf{D})$ computed by the full generalization of Yannakakis' algorithm!

Linear Program on Circuit

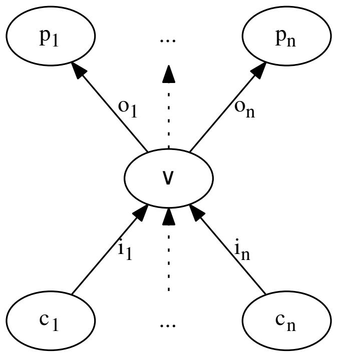
Input Leafs



$W_{o_1} + \ldots + W_{o_n} \leq 1$

Linear Program on Circuit

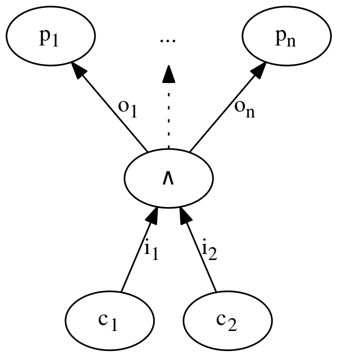
$\vee$-Gates



$W_{i_1} + ... + W_{i_n} = W_{o_1} + ... + W_{o_n}$

Linear Program on Circuit

$\wedge$-Gates



$W_{i_1} = W_{i_2} = W_{o_1} + ... + W_{o_n}$

Solutions Correspond

An edge e of the dDNNF C represent the set $C(e)$ of all tuples τ accepted by C via e.

$$W_e = \sum_{\tau \in C(e)} W_\tau$$

Conclusion

- Query answering for select-from-where SQL-queries is hard
- Yannakakis's algorithm is a powerful algorithm that compiles a quantifier-free ACQ and a database to a dDNNFs.
- Answer aggregation for dDNNFs can be computed in polynomial time.
- Dependency weighted aggregates for dDNNFs can be computed in polynomial time.