

# Towards Optimising Query Evaluation with Quotient Databases

José María Turull Torres and Sebastian Link

*Information Science Research Centre,  
Massey University, New Zealand*

1. Introduction
2. Notation
3. The Concept of Type
4. Problems when Evaluating Queries on Quotient Databases
5. Suitable Classes of Databases
6. Suitable Classes of Computable Queries

## 1.1 General Introduction

- relational databases from a **logical point of view**
- representation independence: dbs modeling same state are same dbs
- **queries**: recursive functions preserving isomorphism
- computation models: machines, programming languages, logics etc.
- single db: preserving isomorphism means preserving automorphism
- two elements with the same structural properties are indistinguishable
- structural property: relation to all other elements in database
- **$\mathcal{L}$ -type**: set of satisfied  $\mathcal{L}$ -formulae up to one free variable
- finite: tuples of same  $FO$ -type iff commutable by automorphism
- unary queries: answer to every query on given db is union of equivalence classes in equality relation of  $FO$ -types defined in given db

## 1.2 Introduction: Suitable Classes of Databases

- two general problems:
  - **infinitely many  $FO$ -types** in class of dbs over given schema
  - **building isolating formula** for  $FO$ -type is **exponential**
- first approach: consider classes of dbs for which
  - number of  $FO$ -types for whole class is finite
  - isolating formula for those  $FO$ -types can be built in PTIME
- **census dbs**: answers to fixed set of questions for any population
- evaluating c. query reduces time from  $\mathcal{O}(f(n))$  to  $\mathcal{O}(n + f(\log n))$

## 1.3 Introduction: Suitable Query Classes

- consider arbitrary dbs, but query classes:
  - **computable on quotient** of given db's domain and
  - quotients **efficiently computable**
- preserving equality of  $FO^k$ -theories and  $C^k$ -theories
- answer is union of equivalence classes in equality relation of types
- Otto: **equality** of these types **decidable in PTIME**
- quotient of any database's domain can be efficiently pre-computed

## 2.1 Notation—DBs and Queries

- $\sigma = \langle R_1, \dots, R_s \rangle$  db schema with arities  $r_1, \dots, r_s$
- **instance** over  $\sigma$ : structure  $I = \langle \text{dom}(I), R_1^I, \dots, R_s^I \rangle$
- **size** of the db  $I$  is  $|\text{dom}(I)|$
- **k-tuple**  $\bar{a}_k$ : tuple of length  $k$  formed by elements from  $\text{dom}(I)$
- $\mathcal{B}_\sigma$ : class of all dbs of schema  $\sigma$
- **computable query of arity  $r \geq 1$  and schema  $\sigma$** :
  - total recursive function  $q^r : \mathcal{B}_\sigma \rightarrow \mathcal{B}_{\langle R \rangle}$
  - isomorphism preserving
  - $\text{dom}(q(I)) \subseteq \text{dom}(I)$  for every  $I$  over  $\sigma$
- **Boolean query** is a 0-ary query
- class of computable queries of schema  $\sigma$  is  $\mathcal{CQ}_\sigma$ , and  $\mathcal{CQ} = \bigcup_\sigma \mathcal{CQ}_\sigma$

## 2.2 Notation—Logic

- purely relational signatures with equality, finite structures only
- **satisfaction:**  $\models_{\mathcal{L}}$ , **equivalence** between dbs:  $\equiv_{\mathcal{L}}$
- $Th_{\mathcal{L}}(I) = \{\varphi \in \mathcal{L}_{\sigma} : I \models_{\mathcal{L}} \varphi\}$
- $\varphi(x_1, \dots, x_r)$  whose free variables in  $\{x_1, \dots, x_r\}$
- $I \models \varphi(x_1, \dots, x_k)[a_1, \dots, a_k]$
- $FO^k$ : fragment of  $FO$  where formulae have variables in  $\{x_1, \dots, x_k\}$
- $C^k$ : adding **counting quantifiers** to  $FO^k$  ( $\exists^{\geq m} x, m \geq 1$ )
- $\exists^{\geq m} x. \varphi(x)$ : at least  $m$  *different* elements in db satisfying  $\varphi$

### 3 The Concept of Type

- **all** properties of  $\bar{a}_k$  in db  $I$  including properties of all subtuples
- $tp_I^{\mathcal{L}}(\bar{a}_k) = \{\varphi \in \mathcal{L}_\sigma : \text{free}(\varphi) \subseteq \{x_1, \dots, x_k\} \wedge I \models \varphi[a_1, \dots, a_k]\}$
- $Tp^{\mathcal{L}}(\sigma, k) = \{tp_I^{\mathcal{L}}(\bar{a}_k) : I \in \mathcal{B}_\sigma \wedge \bar{a}_k \in (\text{dom}(I))^k\}$
- $I$  **realizes** type  $\alpha$  iff  $tp_I^{\mathcal{L}}(\bar{a}_k) = \alpha$  for some  $k$ -tuple  $\bar{a}_k$  over  $I$
- $Tp^{\mathcal{L}}(I, k) = \{tp_I^{\mathcal{L}}(\bar{a}_k) : \bar{a}_k \in (\text{dom}(I))^k\}$
- $I \equiv_{FO} J \iff I \simeq J$  for every (finite)  $I, J$  over  $\sigma$
- **isolating formula**  $(FO^k, C^k)$ :
  - **single** formula equivalent to type of tuple over given db
  - can be built inductively for given db

## 4 Problems when Evaluating Queries on Quotient DBs

- $tp_I^{FO}(\bar{a}_k) = tp_J^{FO}(\bar{b}_k)$  implies  $Th_{FO}(I) = Th_{FO}(J)$  ( $I \simeq J$ )
- $\bar{a}_k \in q(I)$  implies  $\bar{b}_k \in q(I)$  whenever  $tp_I^{\mathcal{L}}(\bar{a}_k) = tp_J^{\mathcal{L}}(\bar{b}_k)$
- two major problems:
  - $Tp^{\mathcal{L}}(\sigma, k)$  infinite (infinitely many isolating formulae)
  - isolating formula to be built for every  $I \in \mathcal{B}_\sigma$  (size does matter)

## 5.1 The Class $\mathcal{C}$ of Census DBs

- number of realised types will be finite for every db in  $\mathcal{C}$
- $k \in \mathbb{N}$ ,  $\sigma = \langle R, 0, 1 \rangle$ ,  $R$  is  $(k + 1)$ -ary, two constant symbols 0 and 1
- $I \in \mathcal{C} \subseteq \mathcal{B}_\sigma$  iff
  - $I = \langle \{a_1, \dots, a_n\}, R^I \subseteq \{a_1, \dots, a_n\}^{k+1}, 0^I, 1^I \rangle$ ,
  - $a \in \text{dom}(I) \setminus \{0^I, 1^I\}$ :  $(a, z_1, \dots, z_k) \in R^I$  for some  $z_i \in \{0^I, 1^I\}$
  - $(b_1, \dots, b_{k+1}) \in R^I$ :  $(b_2, \dots, b_{k+1}) \in \{0^I, 1^I\}^k$ ,  $b_1 \notin \{0^I, 1^I\}$

## 5.2 Isolating Formulae

- $P \subseteq \{1, \dots, k\}$ :

$$\bar{b}_P = (b_{P,1}, \dots, b_{P,k}) \in \{0^I, 1^I\}^k \text{ by } b_{P,i} = 1 \text{ iff } i \in P$$

- $\varphi_P(x) \equiv \exists z_1, \dots, z_k (R(x, z_1, \dots, z_k) \wedge z_1 = b_{P,1} \wedge \dots \wedge z_k = b_{P,k})$
- $\Phi_{\mathcal{C}} = \{\varphi_P \mid P \subseteq \{1, \dots, k\}\}$
- $I \models \varphi(x)$  for  $\varphi \in \Phi_{\mathcal{C}}$  and all  $I \in \mathcal{C}$ ,  $a \in \text{dom}(I)$  with  $a \notin \{0^I, 1^I\}$
- every  $\varphi(x) \in \Phi_{\mathcal{C}}$  is automorphism type for elements

$$f(x) = \begin{cases} b & , \text{ if } x = a, \\ a & , \text{ if } x = b, \\ x & , \text{ else} \end{cases}$$

- $\forall q \in \mathcal{CQ}. \forall I \in \mathcal{C}: q(I)$  is equivalent to

$$\alpha_{q,I} \equiv \bigvee_{\beta \in \Gamma} \beta(x)$$

for some  $\Gamma \subseteq \Phi_{\mathcal{C}} \cup \{x = 0, x = 1\}$

- $q(I)$  is *FO* definable since  $q$  preserves isomorphism, i.e.,  $I \equiv_{FO} J$
- quotient is  $\{0^I\}, \{1^I\}, A_{\varphi} = \{a \in \text{dom}(I) \mid I \models \varphi(x)[a]\} \forall \varphi \in \Phi_{\mathcal{C}}$
- $q(I)$  empty or union of some  $\{\{0^I\}, \{1^I\}\} \cup \{A_{\varphi} \mid \varphi \in \Phi_{\mathcal{C}}\}$
- are defined by  $x = 0, x = 1$  and  $\varphi$  with  $\varphi \in \Phi_{\mathcal{C}}$

## 5.3 Representation of Quotient DBs in $\mathcal{C}$

- store data into two separate tables
- first table: one representative from every class for all realised types
- second table: values for  $B_1$  to  $B_k$  plus cardinality of classes

$id$	$B_1$	$\cdots$	$B_k$
1	0	$\cdots$	0
$\vdots$	$\vdots$	$\cdots$	$\vdots$
$2^k$	1	$\cdots$	1

$B_1$	$\cdots$	$B_k$	$n$
0	$\cdots$	0	$n_1$
$\vdots$	$\cdots$	$\vdots$	$\vdots$
1	$\vdots$	1	$n_{2^k}$

- $q \in \mathcal{CQ}$  with time complexity  $\mathcal{O}(f(n))$  where  $n$  is the size of the db
- size of quotient:  $\mathcal{O}(1) \cdot \mathcal{O}(\log n) = \mathcal{O}(\log n)$  by pre-computation of equivalence classes

- relation on  $I$ : add cardinality to every class ( $\mathcal{O}(n)$ )
- $\mathcal{O}(n + f(\log n))$  considering sizes of equivalence classes matter
- $q \in \mathcal{CQ}$  with  $\mathcal{O}(f(n))$ : evaluating  $q$  on dbs in  $\mathcal{C}$  takes  $\mathcal{O}(n + f(\log n))$
- first table sufficient if query size-independent from equivalence classes
- size then  $\mathcal{O}(1)$  yielding time complexity of  $\mathcal{O}(n)$  for all those queries

## 6.1 Suitable Classes of Queries

- essential: pre-computing quotient tractable for all queries in class
- two classes preserving realisation of  $\mathcal{L}$ -types for some  $\mathcal{L}$
- pre-computation tractable since equivalence in  $\mathcal{L}$  decidable in PTIME
- quotient sufficient for queries in classes proposed

## 6.2 Preserving Realisation of $FO^k$ -Types

- $k \geq 1, k \geq r \geq 0$ :
  - $QCQ_\sigma^k = \{f^r \in \mathcal{C}Q_\sigma \mid \forall I, J \in \mathcal{B}_\sigma : Tp^{FO^k}(I, k) = Tp^{FO^k}(J, k) \Rightarrow Tp^{FO^k}(\langle I, f(I) \rangle, k) = Tp^{FO^k}(\langle J, f(J) \rangle, k)\}$
  - $\langle I, f(I) \rangle$  and  $\langle J, f(J) \rangle$  dbs over  $\sigma \cup \{R\}$
  - $f^r$  union of complete  $FO^k$  types for all databases  $I$  in  $\mathcal{B}_\sigma$
  - $QCQ^k = \bigcup_\sigma QCQ_\sigma^k$  and  $QCQ^\omega = \bigcup_{k \geq 1} QCQ^k$
- how much does db need to be explored for evaluating query on it
- some need properties up to  $FO$ , some need only up to  $FO^k$
- syntactic characterization by means of reflective relational machines

## 6.3 Preserving Realisation of $C^k$ -Types

- $FO^k$  unable to count beyond  $k$
- $C^k$ : 2 variables sufficient for expressing any output degree
- $k \geq 1, k \geq r \geq 0$ :
  - $QCQ_{\sigma}^{C^k} = \{f^r \in \mathcal{C}Q_{\sigma} \mid \forall I, J \in \mathcal{B}_{\sigma} : Tp^{C^k}(I, k) = Tp^{C^k}(J, k) \Rightarrow Tp^{C^k}(\langle I, f(I) \rangle, k) = Tp^{C^k}(\langle J, f(J) \rangle, k)\}$
  - $\langle I, f(I) \rangle$  and  $\langle J, f(J) \rangle$  dbs over  $\sigma \cup \{R\}$ ,  $R$  is  $r$ -ary
  - $f^r$  union of complete  $C^k$  types for all databases  $I$  in  $\mathcal{B}_{\sigma}$
  - $QCQ^{C^k} = \bigcup_{\sigma} QCQ_{\sigma}^{C^k}$  and  $QCQ^{C^{\omega}} = \bigcup_{k \geq 1} QCQ^{C^k}$
- syntactic characterization by means of reflective counting machines

## 6.4 Examples

- size of db is even  $\in \mathcal{QCQ}^{C^1}$  and  $\notin \mathcal{QCQ}^\omega$
- graph is regular  $\in \mathcal{QCQ}^{C^2}$  and  $\notin \mathcal{QCQ}^\omega$
- graph is Eulerian  $\in \mathcal{QCQ}^{C^2}$  and  $\notin \mathcal{QCQ}^\omega$
- graph disjoint union of even number of cliques  $\in \mathcal{QCQ}^{C^2}$  &  $\notin \mathcal{QCQ}^\omega$
- graph is connected  $\in \mathcal{QCQ}^3$ ,  $\notin \mathcal{QCQ}^2$  and  $\notin \mathcal{QCQ}^{C^2}$
- graph: even number of connected components  $\in \mathcal{QCQ}^{C^\omega}$  &  $\notin \mathcal{QCQ}^\omega$