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 \textbf{logical point of view}
- representation independence: dbs modeling same state are same dbs
- \textbf{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
- \textbf{\(L\)-type}: set of satisfied \(L\)-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 $O(f(n))$ to $O(n + f(\log n))$
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

- $\sigma = \langle R_1, \ldots, R_s \rangle$ db schema with arities $r_1, \ldots, r_s$
- **instance** over $\sigma$: structure $I = \langle dom(I), R^I_1, \ldots, R^I_s \rangle$
- **size** of the db $I$ is $|\ dom(I)|$
- **k-tuple** $\bar{a}_k$: tuple of length $k$ formed by elements from $dom(I)$
- $\mathcal{B}_\sigma$: 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
  - $dom(q(I)) \subseteq dom(I)$ 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: $\equiv_\mathcal{L}$

- $Th_\mathcal{L}(I) = \{ \varphi \in \mathcal{L}_\sigma : I \models_\mathcal{L} \varphi \}$

- $\varphi(x_1, \ldots, x_r)$ whose free variables in $\{x_1, \ldots, x_r\}$

- $I \models \varphi(x_1, \ldots, x_k)[a_1, \ldots, a_k]$

- $FO^k$: fragment of $FO$ where formulae have variables in $\{x_1, \ldots, x_k\}$

- $C^k$: 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

- $tp^\mathcal{L}_I(\bar{a}_k) = \{ \varphi \in \mathcal{L}_\sigma : \text{free}(\varphi) \subseteq \{x_1, \ldots, x_k\} \land I \models \varphi[a_1, \ldots, a_k] \}$

- $Tp^\mathcal{L}(\sigma, k) = \{ tp^\mathcal{L}_I(\bar{a}_k) : I \in \mathcal{B}_\sigma \land \bar{a}_k \in (\text{dom}(I))^k \}$

- $I$ **realizes** type $\alpha$ iff $tp^\mathcal{L}_I(\bar{a}_k) = \alpha$ for some $k$-tuple $\bar{a}_k$ over $I$

- $Tp^\mathcal{L}(I, k) = \{ tp^\mathcal{L}_I(\bar{a}_k) : \bar{a}_k \in (\text{dom}(I))^k \}$

- $I \equiv_{FO} J \iff I \simeq J$ 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

- $tp^F_O(a_k) = tp^F_O(b_k)$ implies $Th_{FO}(I) = Th_{FO}(J)$ ($I \sim J$)

- $a_k \in q(I)$ implies $b_k \in q(I)$ whenever $tp^\mathcal{L}_I(a_k) = tp^\mathcal{L}_J(b_k)$

- 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
  - $I = \langle \{a_1, \ldots, a_n\}, R^I \subseteq \{a_1, \ldots, a_n\}^{k+1}, 0^I, 1^I \rangle$,
  - $a \in \text{dom}(I) \setminus \{0^I, 1^I\}$: $(a, z_1, \ldots, z_k) \in R^I$ for some $z_i \in \{0^I, 1^I\}$
  - $(b_1, \ldots, b_{k+1}) \in R^I$: $(b_2, \ldots, b_{k+1}) \in \{0^I, 1^I\}^k$, $b_1 \notin \{0^I, 1^I\}$
5.2 Isolating Formulae

- $P \subseteq \{1, \ldots, k\}$:
  \[ \overline{b}_P = (b_{P,1}, \ldots, b_{P,k}) \in \{0^I, 1^I\}^k \text{ by } b_{P,i} = 1 \text{ iff } i \in P \]

- $\varphi_P(x) \equiv \exists z_1, \ldots, z_k (R(x, z_1, \ldots, z_k) \land z_1 = b_{P,1} \land \cdots \land z_k = b_{P,k})$

- $\Phi_C = \{ \varphi_P \mid P \subseteq \{1, \ldots, k\} \}$

- $I \models \varphi(x)$ for $\varphi \in \Phi_C$ and all $I \in C, a \in dom(I)$ with $a \notin \{0^I, 1^I\}$

- every $\varphi(x) \in \Phi_C$ is automorphism type for elements

\[ f(x) = \begin{cases} 
  b & \text{if } x = a, \\
  a & \text{if } x = b, \\
  x & \text{else}
\end{cases} \]
• \( \forall q \in \mathcal{C} Q. \forall I \in \mathcal{C} : q(I) \) 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 $C$

- 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

<table>
<thead>
<tr>
<th>id</th>
<th>$B_1$</th>
<th>$\cdots$</th>
<th>$B_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>$\cdots$</td>
<td>0</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\cdots$</td>
<td>$\vdots$</td>
</tr>
<tr>
<td>$2^k$</td>
<td>1</td>
<td>$\cdots$</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$B_1$</th>
<th>$\cdots$</th>
<th>$B_k$</th>
<th>$n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$\cdots$</td>
<td>0</td>
<td>$n_1$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
</tr>
<tr>
<td>1</td>
<td>$\vdots$</td>
<td>1</td>
<td>$n_{2^k}$</td>
</tr>
</tbody>
</table>

- $q \in CQ$ with time complexity $O(f(n))$ where $n$ is the size of the db
- size of quotient: $O(1) \cdot O(\log n) = O(\log n)$ 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

- $k \geq 1$, $k \geq r \geq 0$:
  - $QCQ^k = \{ f^r \in CQ_\sigma \mid \forall I, J \in B_\sigma : Tp^{FO^k}(I, k) = Tp^{FO^k}(J, k) \Rightarrow Tp^{FO^k}(\langle I, f(I) \rangle, k) = Tp^{FO^k}(\langle J, f(J) \rangle, k) \}$
  - $\langle I, f(I) \rangle$ and $\langle J, f(J) \rangle$ dbs over $\sigma \cup \{R\}$
  - $f^r$ union of complete $FO^k$ types for all databases $I$ in $B_\sigma$
  - $QCQ^k = \bigcup_\sigma QCQ^k_\sigma$ 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
- $k \geq 1, k \geq r \geq 0$:
  - $QCQC^k_\sigma = \{ f^r \in QC_\sigma \mid \forall I, J \in B_\sigma : Tp^{C^k}(I, k) = Tp^{C^k}(J, k) \Rightarrow Tp^{C^k}(\langle I, f(I) \rangle, k) = Tp^{C^k}(\langle J, f(J) \rangle, k) \}$
  - $\langle I, f(I) \rangle$ 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 $B_\sigma$
  - $QCQC^k = \bigcup_\sigma QCQC^k_\sigma$ and $QCQC^{\omega} = \bigcup_{k \geq 1} QCQC^k$

- syntactic characterization by means of reflective counting machines
6.4 Examples

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