A Call to Regularity

Moshe Y. Vardi*

Rice University

*Joint work with D. Calvanese, G. De Giacomo, and M. Lenzerini
Database Query Languages

• Standard database query languages (e.g., SQL 2.0) are essentially 1st-order.

• Aho and Ullman, 1978: 1st-order languages are weak; add recursion

• Gallaire and Minker, 1978: add recursion via logic programs

• SQL 3.0, 1999: recursion added

Expressiveness costs money!!!

• 1st-order queries: LOGSPACE

• Recursive queries: PTIME
Datalog

**Datalog:**

- Function-free logic programs
- Existential, positive fixpoint logic
- Select-project-join-union-recur queries

**Example:** *Transitive Closure*

\[
Path(x, y) : - Edge(x, y)
\]

\[
Path(x, y) : - Path(x, x), Path(x, y)
\]

Query Containment

Query Optimization: Given $Q$, find $Q'$ such that:

- $Q \equiv Q'$
- $Q'$ is “easier” than $Q$

Query Containment: $Q_1 \sqsubseteq Q_2$ if $Q_1(B) \subseteq Q_2(B)$ for all databases $B$.

Fact: $Q \equiv Q'$ iff $Q \sqsubseteq Q'$ and $Q' \sqsubseteq Q$

Consequence: Query containment is a key database problem.
Query Containment

Other applications:

- query reuse
- query reformulation
- information integration
- cooperative query answering
- integrity checking
- ... 

**Consequence:** Query containment is a fundamental database problem.
Decidability of Query Containment

- **SQL**: undecidable
  - Folk Theorem
  - Poor theory and practice of optimization

- **SPJU**: decidable
  - Rich theory and practice of optimization

- **Datalog**: undecidable
  - Shmueli–1977
  - Difficult theory and practice of optimization

**Unfortunately**, most decision problems involving Datalog are undecidable - almost no interesting, well-behaved fragments.
1990s: Back to Binary Relations

WWW:

- Nodes
- Edges
- Labels

*Semistructured Data:* WWW, SGML documents, library catalogs, XML documents, Meta data, . . . .

*Formally:* \((D, E, \lambda)\)

- \(D\) - nodes
- \(E \subseteq D^2\) - edge
- \(\lambda: E \rightarrow \Lambda^+\) – labels (alt., also node labels)
Path Queries

Active Research Topic: What is the right query language for semistructured data?

Basic Element of all proposals: path queries

- $Q(x, y) : = x \, L \, y$
- $L$: formal language over labels
- $a \cdot \underline{l_1} \cdots \underline{l_k} \cdot b$
- $Q(a, b)$ holds if $l_1 \cdots l_k \in L$

Example: Regular Path Query

$Q(x, y) : = x \, (\text{Wing} \cdot \text{Part}^+ \cdot \text{Nut}) \, y$
Path-Query Containment

\[ Q_1(x, y) : - x \; L_1 \; y \]
\[ Q_2(x, y) : - x \; L_2 \; y \]

**Language-Theoretic Lemma 1:**

\[ Q_1 \subseteq Q_2 \iff L_1 \subseteq L_2 \]

**Proof:** Consider a database

\[ a \cdot l_1 \cdots l_k \cdot b \text{ with } l_1 \cdots l_k \in L_1 \]

**Corollary:** Path-Query Containment is

- undecidable for context-free path queries
- decidable for regular path queries.
Regular Path Queries

Observations:

- A fragment of Transitive-Closure Logic

- A fragment of binary Datalog
  - Concatenation: \[ E(x, y) : - E_1(x, z), E_2(z, y) \]
  - Union: \[ E(x, y) : - E_1(x, y) \]
    \[ E(x, y) : - E_1(x, y) \]
  - Transitive Closure: \[ P(x, y) : - E(x, z) \]
    \[ P(x, y) : - E(x, z), E(z, y) \]

Consequence:

- Data complexity: \textbf{NLOGSPACE}

- Expression complexity: \textbf{PTIME}

Containment: PSPACE-complete, via nondeterministic automata (Stockmeyer, 1973).
Two-Way RPQs

**Extended Alphabet:** \( \Lambda^- = \{ a^- : a \in \Lambda^+ \} \)
\[ \Lambda = \Lambda^+ \cup \Lambda^- \]

**Inverse Roles:**

\( Part(x, y) \): \( y \) part of \( x \)
\( Part^-(x, y) \): \( x \) part of \( y \)

**Example:** Step Siblings

\( Q(x, y) : - \)
\[ x \quad [(father^- \cdot father) + (mother^- \cdot mother)]^+ \quad y \]

**Containment:** Two-way nondeterministic automata

- Hopcroft and Ullman: 2DFA
- Hopcroft, Motwani and Ullman: ???
Language Containment – Upper Bound

**Lemma:** \( L(E_1) \subseteq L(E_2) \) iff \( L(E_1) - L(E_2) \) = \( \emptyset \)

Algorithm for checking whether \( L(E_1) \subseteq L(E_2) \):

1. Construct NFAs \( A_i \) such that \( L(A_i) = L(E_i) \) – *linear blow-up*.

2. Construct \( \overline{A_2} \) such that \( L(\overline{A_2}) = \Sigma^* - L(A_2) \) – *exponential blow-up*.

3. Construct \( A = A_1 \times \overline{A_2} \) such that \( L(A) = L(E_1) - L(E_2) \) – *quadratic blow-up*.

4. Check if there is a path from start state to final state in \( A \) – *NLOGSPACE*.

**Bottom Line:** \( \text{PSPACE} \)
2NFA

\[ A = (\Sigma, S, S_0, \rho, F) \]

- \( \Sigma \) – finite alphabet
- \( S \) – finite state set
- \( S_0 \subseteq S \) – initial states
- \( F \subseteq S \) – final states
- \( \rho : S \times \Sigma \rightarrow 2^{S \times \{-1,0,+1\}} \) – transition function

**Theorem:** Rabin&Scott, Shepherdson, 1959

2NFA \( \equiv \) 1NFA
2RPQ Containment

Difficulties:

• 2NFA $\rightarrow$ 1NFA: exponential blow-up
  
  – **Consequence:** Doubly exponential complementation

• Difference between query and language containment
  
  – $Q_1(x, y) : - x \text{ Parent } y$
  – $Q_2(x, y) : - x \text{ Parent } \cdot \text{ Parent}^- \cdot \text{ Parent } y$

  – $Q_1 \subseteq Q_2$ but
  
  $L(\text{Parent}) \not\subseteq L(\text{Parent} \cdot \text{Parent}^- \cdot \text{Parent})$
Back to Basics: 2NFA→1NFA

**Theorem:** Vardi, 1988

Let $A = (\Sigma, S, S_0, \rho, F)$ be a 2NFA. There is a 1NFA $A^c$ such that

- $L(A^c) = \Sigma^* - L(A)$
- $\|A^c\| \in 2^O(\|A\|)$

**Proof:** Guess a subset-sequence counterexample

$a_0 \cdots a_{k-1} \notin L(A)$ iff there is a sequence $T_0, T_1, \cdots, T_k$ of subsets of $S$ such that

1. $S_0 \subseteq T_0$ and $T_k \cap F = \emptyset$.
2. If $s \in T_i$ and $(t,+1) \in \rho(s,a_i)$, then $t \in T_{i+1}$, for $0 \leq i < k$.
3. If $s \in T_i$ and $(t,0) \in \rho(s,a_i)$, then $t \in T_i$, for $0 \leq i < k$.
4. If $s \in T_i$ and $(t,-1) \in \rho(s,a_i)$, then $t \in T_{i-1}$, for $0 < i \leq k$. 
Foldings

**Definition:** Let \( u, v \in \Lambda^* \). We say that \( v \) **folds** onto \( u \), denoted \( v \leadsto u \), if \( v \) can be “folded” on \( u \), e.g.,

\[
abb^{−}bc \leadsto abc.
\]

Pictorially,

\[
\xrightarrow{a} \cdot \xrightarrow{b} \cdot \xleftarrow{b} \cdot \xrightarrow{b} \cdot \xrightarrow{c} \leadsto \xrightarrow{a} \cdot \xrightarrow{b} \cdot \xrightarrow{c}.
\]

**Definition:** Let \( E \) be an RE over \( \Lambda \). Then \( \text{fold}(E) = \{v : v \leadsto u, u \in L(E)\} \).

**Language-Theoretic Lemma 2:**

Let \( Q_1(x,y) : = x \ E_1 \ y \)
\[
Q_2(x,y) : = x \ E_2 \ y
\]
be 2RPQs. Then \( Q_1 \sqsubseteq Q_2 \) iff \( L(E_1) \subseteq \text{fold}(E_2) \).
2RPQ containment

**Theorem:** Let $E$ be an RE over $\Lambda$. There is a 2NFA $\tilde{A}_E$ such that

1. $L(\tilde{A}_E) = fold(E)$
2. $|\tilde{A}_E| \in O(|E|)$

**Containment**

$Q_1(x, y) : x \notin E_1 y$

$Q_2(x, y) : x \notin E_2 y$

**TFAE**

1. $Q_1 \sqsubseteq Q_2$
2. $L(E_1) \subseteq fold(E_2)$.
3. $L(E_1) \subseteq L(\tilde{A}_E)$.
4. $L(E_1) \cap L(\tilde{A}_E^c) = \emptyset$
5. $L(A_{E_1} \times \tilde{A}_E^c) = \emptyset$

**Bottom-line:** 2RPQ containment is PSPACE-complete.
View-Based Query Processing 2RPQs

- **Global database**: $B$ over $\Lambda^+$
- **Views**: $\{V_1, \ldots, V_n\}$, $V_i$ is a query
- **View extensions**: $\{E_1, \ldots, E_n\}$, $E_i \subseteq V_i(B)$
- **Global query** $Q$ over $\Lambda$
- **Local query** over $V_1, \ldots, V_n$

**Query Processing**

1. **View-based query answering**: approximate $Q(B)$ using view-extension information.

2. **View-based query rewriting**: approximate global query by a local query based on view definitions

3. **View-based query losslessness**: Compare global query with its view-based approximation.

4. **View-based query containment**: Compare view-based approximations of two global queries.
 Conjunctive Queries

**Conjunctive Query**: Existential, conjunctive, positive first-order logic, i.e., first-order logic without $\forall, \vee, \neg$; written as a rule

$Q(x_1, \ldots, x_n) : \equiv R(x_3, y_2, x_4), \ldots, S(x_2, y_3)$

**Significance**:

- Most common SQL queries (*Select-Project-Join*)
- Core of Datalog

**Example**:

$Triangle(x, y, z) : \equiv Edge(x, y), Edge(y, z), Edge(z, x)$
Conjunctive Query Containment

**Canonical Database** $B^Q$:

- Each variable in $Q$ is a distinct element
- Each subgoal $R(x_3, y_2, x_4)$ of $Q$ gives rise to a tuple $R(x_3, y_2, x_4)$ in $B^Q$

**Fact:** (Chandra and Merlin, 1977)

For conjunctive queries $Q_1$ and $Q_2$, TFAE:

- The containment $Q_1 \subseteq Q_2$ holds
- There is a homomorphism $h : B^{Q_2} \rightarrow B^{Q_1}$ that is the identity on distinguished variables.
Conjunctive 2RPQ

\textbf{C2RPQ}: Core of all semistructured query languages

\[ Q(x_1, \ldots, x_n) : = y_1 E_1 z_1, \ldots, y_m E_m z_m \]

\begin{itemize}
  \item \( E_i \) – 2RPQ
\end{itemize}

\textbf{Intuition:}

\begin{itemize}
  \item C2RPQs are obtained from CQ by replacing atoms with REs over \( \Lambda \).
  \item C2RPQs are Select-Project- “Regular Join” queries.
\end{itemize}

\textbf{Example:}

\[ Q(x, y) : = z (Wing \cdot Part^+ \cdot Nut) x, \]
\[ z (Wing \cdot Part^+ \cdot Nut) y \]
C2RPQ Containment

**Difficulty**: Earlier techniques do not apply

- No canonical database
- No language-theoretic lemma

**Solution**: Combine and extend earlier ideas

- Infinite family of canonical databases
  - Each variable in $Q$ is a distinct element
  - Each subgoal $y_iE_i\tilde{z}_i$ of $Q$ is replaced by a simple path labeled by a word in $L(E_i)$.

- Represent canonical databases as words over a larger alphabet

- Develop automata-theoretic characterization of C2RPQ containment.

**Bottom-line**: C2RPQ containment is EXPSPACE-complete.
In Conclusion

Regular queries:

- A rich but well-behaved fragment of Datalog
- Of special interest for semistructured data
- Beautiful application of classical formal-language theory
- Novel theory of regular paths in labeled graphs

Research Question: What is the ultimate class of regular queries?

- $RPQs$
- $2RPQs$
- $C2RPQs$
- $UC2RPQs$
- ...