Database Queries - Logic and Complexity
How Multivariate Complexity Theory Began

Moshe Y. Vardi

Rice University

First-Order Logic

A formalism for specifying properties of mathematical structures, such as *graphs*, *partial orders*, *groups*, *rings*, *fields*, . . .

Mathematical Structure:

$$\mathbf{A} = (D, R_1, \ldots, R_k, f_1, \ldots, f_l),$$

- D is a non-empty set – *universe*, or *domain*
- R_i is an m-ary *relation* on D, for some m (that is, $R_i \subseteq D^m$)
- f_j is an n-ary *function* on D, for some n (that is, $f_i : D^n \to D$)

Examples

Graph $\mathbf{G} = (V, E)$

- V: nodes
- $E \subseteq V^2$: edges

Groups $\mathbf{G} = (V, \cdot)$

- V: elements
- $\cdot : V^2 \to V$: product

First-Order Logic on Graphs

Syntax:

- First-order variables: x, y, z, . . . (range over nodes)
- Atomic formulas: $E(x, y)$, $x = y$
- Formulas: Atomic Formulas + Boolean Connectives ($\vee$, $\wedge$, $\neg$) + First-Order Quantifiers ($\exists x$, $\forall x$)

Examples:

- "node x has at least two distinct neighbors"

$$(\exists y)(\exists z)(\neg(y = z) \wedge E(x, y) \wedge E(x, z))$$

Concept: x is *free* in the above formula, which expresses a property of nodes.

- "each node has at least two distinct neighbors"

$$(\forall x)(\exists y)(\exists z)(\neg(y = z) \wedge E(x, y) \wedge E(x, z))$$

Concept: The above is a *sentence*, that is, a formula with no free variables; it expresses a property of graphs.

Semantics of First-Order Logic

Semantics:

- First-order variables range over elements of the universes of structures
- To evaluate a formula φ, we need a graph $\mathbf{G}$ and a *binding* α that maps the free variables of φ to nodes of $\mathbf{G}$

Notation: $\mathbf{G} \models_\alpha \varphi(x_1, \ldots, x_k)$

Fundamental Distinction: Syntax vs. semantics (Tarski, 1930)

From Model Theory to Relational Databases

- A sentence ψ is either true or false on a given graph $\mathbf{G}$. In particular, sentences specify classes of graphs: $\mathsf{models}(\psi) = \{\mathbf{G} : \mathbf{G} \models \psi\}$

  Model Theory: Logic provides a metatheory for mathematical modeling.

- E.F. Codd, 1970: Formulas $\varphi(x_1, \ldots, x_k)$ define *queries*:
  $\varphi(\mathbf{G}) = \{\langle \alpha(x_1), \ldots, \alpha(x_k)\rangle : \mathbf{G} \models_\alpha \varphi(x_1, \ldots, x_k)\}$

  Example: $(\exists y)(\exists z)(\neg(y = z) \wedge E(x, y) \wedge E(x, z))$ – "List nodes that have at least two distinct neighbors"

  Relational Databases: \$30B+ industry

Relational Databases

Codd's Two Fundamental Ideas:

- *Tables are relations*: a *row* in a table is just a *tuple* in a relation; order of rows/tuples does not matter!

- *Formulas are queries*: they specified the *what* rather then the *how* – declarative programming!

Algorithmic Problems in First-Order Logic

Truth-Evaluation Problem (Model Checking): Given a first-order formula $\varphi(x_1, \ldots, x_k)$, a graph $\mathbf{G}$, and a binding α, does $\mathbf{G} \models_\alpha \varphi(x_1, \ldots, x_k)$?

Satisfiability Problem: Given a first-order formula ψ, is there a graph $\mathbf{G}$ and binding α, such that $\mathbf{G} \models_\alpha \psi$?

Facts:

- Satisfiability is *undecidable* [Church, Turing, 1936]
- Truth evaluation, which is query evaluation, is *decidable*.

Beyond First-Order Logic

Fagin, 1976: graph connectivity is *not expressible* in first-order logic!

• There *is no* first-order formula $\varphi(x, y)$ that says there is a path in graph G from node x to node y.

Aho&Ullman, 1980: Augment FO with *fixpoints*.

Aho&Ullman, 1980: FO$<$FP.

Standard Complexity Theory

Standard Complexity Analysis – *Scaling Behavior*

- Focus on *decision* (yes/no) problems to eliminate dependence on output size.

- Measure how run time/memory usage *grows* as function of input size.

Database Context:

- Focus on Boolean (yes/no) queries to eliminate dependence on output size.

- Input size: database size *plus* query size.

Failure of Standard Complexity Theory

Difficulty:

- Typical input size is $10^9 + 100$
- Which size is more challenging? $2 \cdot 10^9 + 100$ or $10^9 + 200$?

Intuition: Database size and query size play *very different* roles! This is *not reflected* in standard complexity theory.

Relational Complexity Theory – V. 1982

Basic Principle: Separate the influences of data and query on complexity

- *Influence of Query*: Fix data
- *Influence of Data*: Fix query

Real-Life Motivation:

- *Census Data Analysis*: data fixed for 10 years, multiple queries
- *Technical Trading*: price-arbitrage fixed query, data changes momentarily

Separate Influence of Data and Query

A Tale of Two Complexities:

- *Query Complexity* of query language L: Fix $\mathbf{B}$, study

$$\{Q \in L : Q(\mathbf{B}) \text{ is nonempty}\}$$

- *Data Complexity* of query language L: Fix $Q \in L$, study

$$\{\mathbf{B} : Q(\mathbf{B}) \text{ is nonempty}\}$$

From Query Complexity to Expession Complexity

Observation:

- Data complexity is insensitive to syntax of queries, as queries are fixed.

- Query complexity is highly sensitive to syntax of queries, e.g.,

– $R \times R \times R \times R \times R \times R \times R \times R \times R \times R$

– R^{10}

Conclusion: Change "*Query Complexity*" to "*Expression Complexity*".

Data vs Expression Complexity

Basic phenomenon: exponential gap!

Query Lang.	Data Comp.	Expression Comp.
FO	LOGSPACE	PSPACE
FP	PTIME	EXPTIME
$\exists$SO	NP	NEXPTIME
PFP	PSPACE	EXPSPACE

Recall: Exponential gaps are strict!

Theory justifies intuition: Characteristics of queries matter much more than size of data!

Relational Complexity Theory – V. 1995

Question: Why is expression complexity so high? How do databases evaluate queries in practice?

Intuitive Answer: Large intermediate results!

- How much is $1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 8 \times 9 \times 0$?

- *Example*: $R_1 \bowtie R_2 \bowtie R_3 \bowtie R_4 \bowtie R_5$ can be empty, even when $R_1 \bowtie R_2 \bowtie R_3$ is very large.

Question: Can we formalize this intuition?

Answer: *Variable-confined queries*

Variable-Confined Queries

Definition: L^k consists of formulas of logic L with at most k variables (Barwise, 1977)

Example: "There exists a path of length 2"

- FO^3: $(\exists x)((\exists y)(\exists z)(R(x,y) \wedge R(y,z))$
- FO^2: $(\exists x)((\exists y)(R(x,y) \wedge (\exists x)R(y,x))$

Key Result: Variable-confined queries have lower expression complexity!

Query Lang.	Data Compl.	Expression Comp.	VC Expr. Comp.
FO	LOGSPACE	PSPACE	PTIME
FP	PTIME	EXPTIME	PTIME
∃SO	NP	NEXPTIME	NP
PFP	PSPACE	EXPSPACE	PSPACE

Variable-Confined Queries Are Easier

Conclusion: Exponential gap between data complexity and expression complexity *shrinks or vanishes* for variable-confined queries.

Optimization Problem: Find smallest k such that given FO query Q in is FO^k.

Answer: Undecidable!

Conjunctive Queries

Conjunctive Query: First-order logic without $\forall, \vee, \neg$; written as a rule
$Q(X_1, \ldots, X_n) \ :- \ R(X_3, Y_2, X_4), \ldots, S(X_2, Y_3)$

Significance: most *common* SQL queries (*Select-Project-Join*)

Example: $GrandParent(X, Y) \ :- \ Parent(X, Z), Parent(Z, Y)$

Equivalently: $(\exists Z)(Parent(X, Z) \wedge Parent(Z, Y))$

Complexity of Conjunctive Queries

Chandra&Merlin, 1977: Expression complexity of CQ is NP-complete.

Precise Complexity Analysis: $||B|^{||Q||}$, for evaluating query Q, over database B.

Yannakakis, 1995: $||B||^{||Q||}$ is much worse than $c^{||Q||} \cdot ||B|^d$ for fixed c, d, which is *fixed-parameter tractable* (FPT) – **parameterized complexity analysis**

Papadimitriou&Yannakakis, 1997: CQ evaluation is *W[1]-complete* – unlikely to be FPT.

Variable-Confined CQ

V., 1995: CQ^k – CQ using at most k variables.

- If Q is in CQ^k, then query can be evaluated over database B in time $||Q|| \cdot ||B||^k$ - FPT!

Example: Contrast

$$(\exists x, y, z)(R(x, y) \wedge R(y, z))$$

and

$$(\exists x)((\exists y)(R(x, y) \wedge (\exists x)R(y, x))$$

Hardness of CQs

• **Observation**: The critical parameter is *number of variables*, *not size of query*!

• **Question**: Characterize smallest k such that a given conjunctive query Q is in CQ^k.

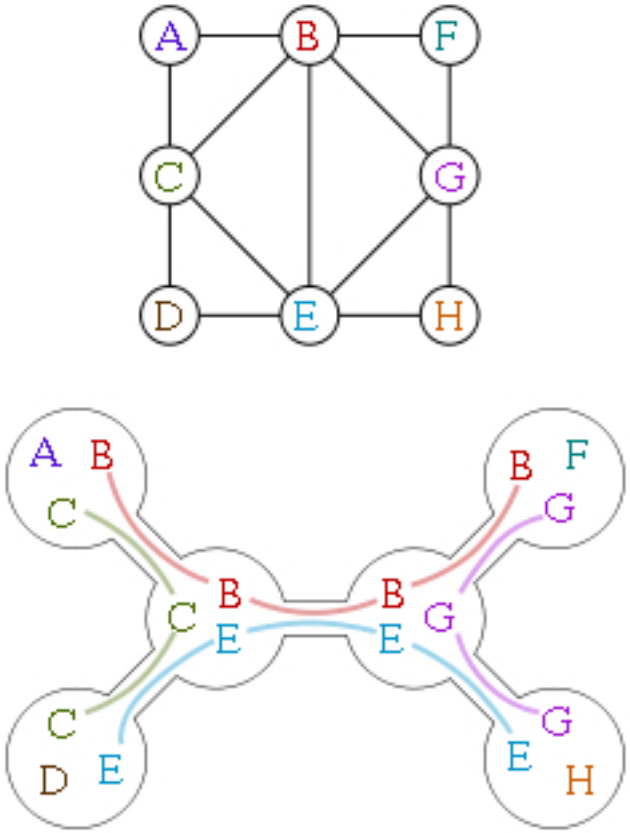

Figure 1: Tree Decomposition of Width 2

Treewidth

Treewidth: "width" of best tree decomposition – measures "tree-likeness" of graphs

- A tree has treewidth 1.
- A cycle has treewith 2.
- An $m \times m$ grid has treewidth m.

CQs Treewidth

Query Graph: graph of a conjunctive query

- *Nodes*: variables
- *Edges*: connect nodes that co-occur in an atom

Definition: $treewidth(Q)$ is $treewidth(graph(Q))$.

Kolaitis&V., 1998: Q is in CQ^k iff $treewidth(Q) < k$.

Corollary: Bounded treewidth CQs are fixed-parameter tractable.

Theory and Practice

Question: Can the theory be used to optimize CQs?

Partial Answer: Not easily!

- Finding treewidth of a graph is NP-hard!

- But heuristics help – exponential improvement for large CQs. [McMahan&V., 2004]

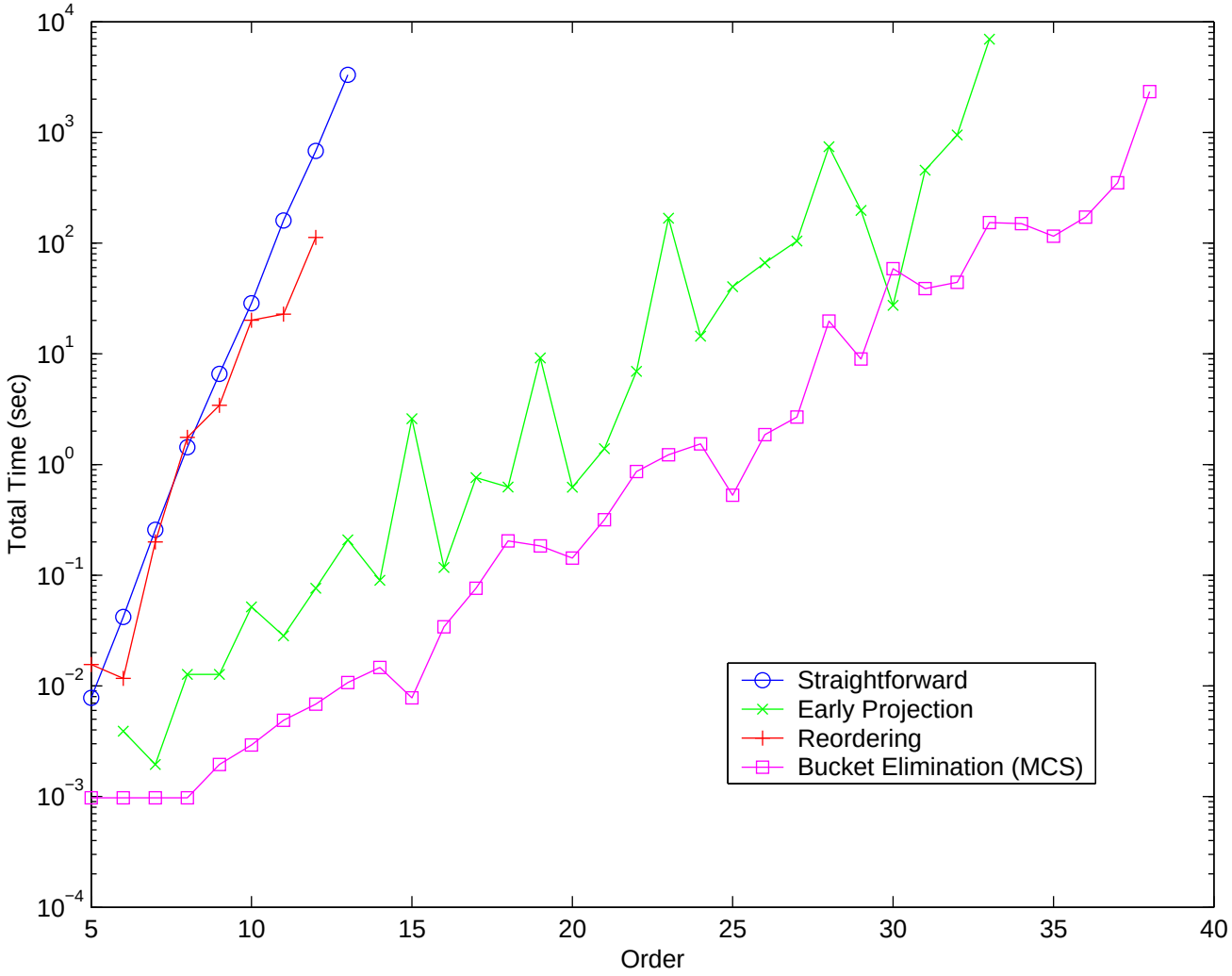

Figure 2: Experimental Results

In Conclusion

Role of Theory:

- Clarify conceptual framework
- Suggest experimental possibilties

Paradigmatic Example: Codd's Relational model

This Talk:

- *Conceptual Framework*: data and expression complexity
- *Optimization Heuristics*: treewidth minimization