Constraint Satisfaction: An Introduction - Part II

Moshe Y. Vardi

Rice University

Existential Second-Order Logic

ESO: FO + existentially quantified relations

Example: *Disconnectivity*

$$
\begin{array}{ll}
(\exists S) & ((\exists x)S(x) \wedge (\exists x)(\neg S(x)) \wedge \\
 & (\forall x)(\forall y)((S(x) \wedge (\neg S(y))) \rightarrow \neg E(x,y)))
\end{array}
$$

Data Complexity of ESO: NP-complete

- *In NP*: Guess quantified relations and check FO in PTIME.
- *NP-hard*: 3-COLOR is expressible in ESO.

Fagin's Theorem: ESO=NP

- *Descriptive Complexity Theory*: machine-independent characterization of complexity class

ESO and CSP

Obviously: CSP$\subseteq$ESO

Examples:

- *3-COLOR*:

$$\begin{array}{ll}(\exists C_1, C_2, C_3)(\forall x, y) & ((C_1(x) \vee C_2(x) \vee C_3(x))\wedge \\ & \neg(E(x,y) \wedge C_1(x) \wedge C_1(y))\wedge \\ & \neg(E(x,y) \wedge C_2(x) \wedge C_2(y))\wedge \\ & \neg(E(x,y) \wedge C_3(x) \wedge C_3(y))).\end{array}$$

- *3-SAT*: T_0, T_1, T_2, T_3 – clause types

$$\begin{array}{l}(\exists T)(\forall x_1, x_2, x_3) \\ ((C_0(x_1,x_2,x_3) \rightarrow T(x_1) \vee T(x_2) \vee T(x_3))\wedge \\ (C_1(x_1,x_2,x_3) \rightarrow \neg T(x_1) \vee T(x_2) \vee T(x_3))\wedge \\ (C_2(x_1,x_2,x_3) \rightarrow \neg T(x_1) \vee \neg T(x_2) \vee T(x_3))\wedge \\ (C_3(x_1,x_2,x_3) \rightarrow \neg T(x_1) \vee \neg T(x_2) \vee \neg T(x_3))).\end{array}$$

CSP and MMSNP

$$
\begin{array}{ll}
(\exists C_1, C_2, C_3)(\forall x, y) & ((C_1(x) \vee C_2(x) \vee C_3(x))\wedge \\
& \neg(E(x,y) \wedge C_1(x) \wedge C_1(y))\wedge \\
& \neg(E(x,y) \wedge C_2(x) \wedge C_2(y))\wedge \\
& \neg(E(x,y) \wedge C_3(x) \wedge C_3(y))).
\end{array}
$$

MMSNP Characteristics:

- *Strict*: universal first-order quantifiers only (SNP)
- *Monadic*: quantified sets only
- *Monotone*: all occurrences of E negated
- *Without inequality*: No $\neq$

Proposition: (Feder&V., 1993)
$CSP(\mathbf{B})$ is in MMSNP for all $\mathbf{B}$.

MMSNP vs CSP

Theorem 1: (Feder&V., 1993, Kun, 2006)
CSP and MMSNP are polynomial equivalent.

- Same dichotomy question

Theorem 2: (Feder&V., 1993) NP is polynomially equivalent to

- monotone, monadic SNP, with inequality
- monadic SNP without inequality
- monotone SNP without inequality

Corollary: Assuming $P \neq NP$, the Dichotomy Conjecture fails for these extensions of MMSNP.

Combinatorial Characterizations:

- MMSNP (Madelaine&Stewart, 2007)
- NP (Nešetřil&Kun, 2007)

Datalog and Non-Uniform CSP

Example: NON 2-COLORABILITY

$$
\begin{array}{rcl}
O(X,Y) & :- & E(X,Y) \\
O(X,Y) & :- & O(X,Z), E(Z,W), E(W,Y) \\
Q & :- & O(X,X)
\end{array}
$$

Recall: Datalog $\subseteq$ PTIME

Define: $\overline{\mathrm{CSP}(\mathbf{B})} = \{\mathbf{A} : \mathbf{A} \not\in \mathrm{CSP}(\mathbf{B})\}$.

Datalog vs. Non-Uniform CSP: Explanation for many tractability results

- $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in Datalog

Note: $\overline{\mathrm{CSP}(\mathbf{B})}$ is positively monotone, like Datalog.

k-Datalog

Definition:

- k-Datalog: Datalog with at most k variables per rule (Non 2-Colorability is in 3-Datalog)

- $\exists\mathrm{IL}^k$: k-variable existential positive infinitary logic
  - variables: $x_1, \ldots, x_k$
  - no universal quantifiers
  - no negations
  - infinitary conjunctions and disjunctions

Facts: Fix $k \geq 1$

- k-Datalog $\subset$ $\exists\mathrm{IL}^k$

- $\exists\mathrm{IL}^k$ can be characterized in terms of *existential k-pebble games* between the *Spoiler* and the *Duplicator*.

- There is a PTIME algorithm to decide whether the *Spoiler* or the *Duplicator* wins the existential k-pebble game.

Existential k-Pebble Games

$\mathbf{A}, \mathbf{B}$: structures

- **Spoiler**: *places* on or *removes* a pebble from an element of $\mathbf{A}$.
- **Duplicator**: tries to duplicate move on $\mathbf{B}$.

$\mathbf{A}$: $a_1, a_2, \ldots, a_l$ $\qquad\qquad$ $l \leq k$
$\mathbf{B}$: $b_1, b_2, \ldots, b_l$

- *Spoiler wins*: $h(a_i) = b_i$, $1 \leq i \leq l$ is not a homomorphism.
- *Duplicator wins*: otherwise.

Fact: (Kolaitis&V., 1995)

$\mathbf{B}$ satisfies the same $\exists \mathrm{IL}^k$ sentences as $\mathbf{A}$ iff the Duplicator wins the existential k-pebble game on $\mathbf{A}, \mathbf{B}$.

k-Datalog and CSP

Theorem: (Kolaitis&V., 1998): TFAE for $k \geq 1$ and a structure $\mathbf{B}$:

- $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in k-Datalog
- $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in $\exists \mathrm{IL}^k$
- $\mathrm{CSP}(\mathbf{B}) = \{\mathbf{A}$: *Duplicator* wins the existential k-pebble game on $\mathbf{A}$ and $\mathbf{B}\}$.

Intuition: $\overline{\mathrm{CSP}(\mathbf{B})} \in k$-Datalog implies that existence of homomorphism is equivalent to the Duplicator winning the existential k-pebble game.

Winning Configurations

Let $\mathbf{A} = (A, R_1^{\mathbf{A}}, \ldots, R_m^{\mathbf{A}})$ and $\mathbf{B} = (B, R_1^{\mathbf{B}}, \ldots, R_m^{\mathbf{B}})$ be two relational structures. Let $\mathbf{a} \in A^k$ and $\mathbf{a} \in B^k$.

Definition:

- $(\mathbf{a}, \mathbf{b})$ is a *winning configuration* for the Duplicator if the Duplicators wins the existential k-pebble game starting from the position $(\mathbf{a}, \mathbf{b})$.

- $W^k(\mathbf{A}, \mathbf{B})$ is the set of winning configurations.

Theorem: (Kolaitis&V., 1998)

- $W^k(\mathbf{A}, \mathbf{B})$ is definable in least-fixpoint logic over the combined structure $\mathbf{A} + \mathbf{B}$.

- For a fixed structure $\mathbf{B}$, the complement of $W^k(\mathbf{A}, \mathbf{B})$ is definable in k-Datalog over the structure $\mathbf{A}$.

Corollary: For a fixed structure $\mathbf{B}$, there is a k-Datalog program $\rho_{\mathbf{B}}$ such that $\rho_{\mathbf{B}}(\mathbf{A})$ is nonempty iff the Spoiler wins the existential k-pebble game on $\mathbf{A}, \mathbf{B}$.

k-Datalog and CSP

$\rho_{\mathbf{B}}$:

- If $\rho_{\mathbf{B}}(\mathbf{A})$ is nonempty, then $\mathbf{A} \not\in \mathrm{CSP}(\mathbf{B})$.
- If $\overline{\mathrm{CSP}(\mathbf{B})}$ is definable in k-Datalog, then it is definable by $\rho_{\mathbf{B}}$.
- *Problem 1*: Decide for a given $\mathbf{B}$ whether $\overline{\mathrm{CSP}(\mathbf{B})}$ is definable by $\rho_{\mathbf{B}}$, that is, is $\overline{\mathrm{CSP}(\mathbf{B})}$ in k-Datalog?
- *Problem 2*: Is $\overline{\mathrm{CSP}(\mathbf{B})}$ in k-Datalog for *some* k?

Conjecture:

- Problem 1 is decidable.
- Problem 2 is undecidable.

Theorem: (Feder&V., 1993): Problem 2 is decidable for monadic Datalog.

Database Perspective

Database Perspective of k-Datalog: Fix $\mathbf{B}$

$$\mathrm{CSP}(\mathbf{B}) = \{\mathbf{A} : \rho_{\mathbf{B}}(\mathbf{A}) \text{ is empty}\}$$

Complexity of Non-Uniform CSP: data complexity of Datalog

Earlier Database Perspective: Fix a structure $\mathbf{B}$

$$\mathrm{CSP}(\mathbf{B}) = \{\mathbf{A} : Q_{\mathbf{A}}(\mathbf{B}) \text{ is nonempty}\}$$

Complexity of Non-Uniform CSP: expression complexity of conjunctive queries

Winning Strategies

$\mathbf{A}, \mathbf{B}$: structures

Definition: A *winning strategy* for the Duplicator in the existential k-pebble game on $\mathbf{A}, \mathbf{B}$ is a nonempty family $\mathcal{F}$ of k-partial homomorphisms from $\mathbf{A}$ to $\mathbf{B}$ that has the k-*forth property*:

- for every $f \in \mathcal{F}$ with $|f| < k$ and every $a \in A$ on which f is undefined, there is a $g \in \mathcal{F}$ that extends f and is defined on a.

Intuition:

- Think of f as a position in the game.
- If Spoiler places pebble on a, Duplicators responds with a pebble on $g(a)$.

CSP and Consistency

Key Idea in AI Research: *local consistency*

- extensibility of partial solutions
- backtrack-free search

Definition: A homomorphism instance $\mathbf{A}, \mathbf{B}$ is

- i-*consistent*: For every consistent assignment α to $i-1$ variables $v_1, \ldots, v_{i-1}$ and a variable v_i, there is a consistent extension of α to v_i.
- *Strongly* k-*consistent*: i-consistency holds for all $i \leq k$.

Game-Theoretic Formulation: (Kolaitis&V., 2000)

$\mathbf{A}, \mathbf{B}$ is strongly k-consistent iff the family of *all* k-partial homomorphism from $\mathbf{A}$ to $\mathbf{B}$ is a winning strategy for the Duplicator in the existential k-pebble game on $\mathbf{A}, \mathbf{B}$.

From Local to Global Consistency

Condition for Tractability: (Dechter, 1992)

Establishing strong k-consistency implies existence of solution.

- Start with $\mathbf{A}, \mathbf{B}$.
- Generate a strongly k-consistent instance $\mathbf{A}', \mathbf{B}'$ over the same domains.
- If h is a k-partial homomorphism from $\mathbf{A}'$ to $\mathbf{B}'$, then h is a k-partial homomorphism from $\mathbf{A}$ to $\mathbf{B}$.
- A function h is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ iff it is a homomorphism from $\mathbf{A}'$ to $\mathbf{B}'$.

Intuition: $\mathbf{A}', \mathbf{B}'$ is obtained by inferring from $\mathbf{A}, \mathbf{B}$ all constraints on k-variables – *constraint propagation* using the *join* and project rules.

Consistency and k-Datalog

Strong k-Consistency vs. Games: (Kolaitis&V., 2000)

The strongly k-consistent instance $\mathbf{A}', \mathbf{B}'$ can be obtained by "re-formatting" $W^k(\mathbf{A}, \mathbf{B})$

Intuition: Computing winning strategies can be viewed as constraint propagation.

Corollary: Let $\mathbf{B}$ a relational structure. $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in k-Datalog iff for every structure $\mathbf{A}$, establishing strong k-consistency for $\mathbf{A}, \mathbf{B}$ implies that there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$.

Theorem: (Kolaitis&Panttaja, 2003) The following problem is $\mathrm{EXPTIME}$-complete: given a positive integer k and two structures $\mathbf{A}$ and $\mathbf{B}$, can strong k-consistency be established?

Surprise: Establishing local consistency is harder than solving the problem!

- (Hertel&Urquhart, 2006): The following problem is $\mathrm{EXPTIME}$-complete: given $k > 1$ and a CNF formula φ, is φ refutable within *width* k?

Databases vs AI

Fundamentally different approaches:

- *AI*: search one answer
- *Databases*: generate all answers

Fundamental Question:: Is query evaluation effective for solving CSP?

Recall: $\mathbf{A} \mapsto Q_{\mathbf{A}}$

- $h : \mathbf{A} \to \mathbf{B}$ iff $Q_{\mathbf{A}}(\mathbf{B})$ is nonempty

Experiment: (with A. San Miguel Aguirre)

- Express 3-SAT as a CSP.
- Translate to conjunctive-query evaluation.
- Run on DB2.

Outcome: Terrible performance!!!

- DB2 optimizer ill suited.

AI vs Databases

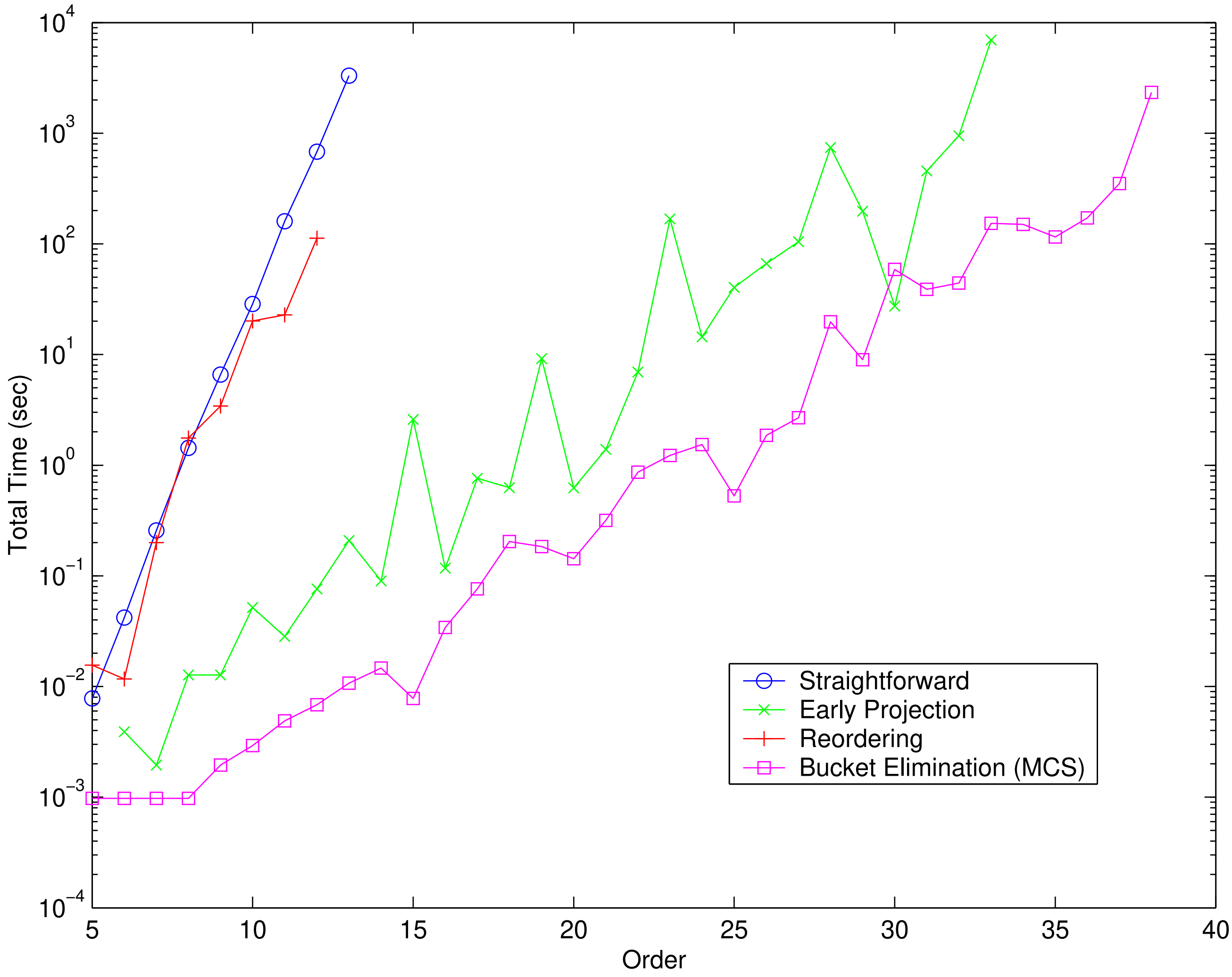
Hard problem for databases: evaluation of large conjunctive queries

- Corresponds to evaluating a sequence of joins and projections.
- Many possible evaluation orders possible
- Query optimizer has to search a very large space

An Alternative Approach: (McMahan&V., 2004)

- Consider the problem as a CSP problem.
- Apply CSP heuristics for constraint propagation.
- Focus on minizing the size of intermediate relations.

Experimental Results



Database Integration

Old Problem: An enterprise has many databases. How do we integrate all these databases to give us a unified picture?

New Problem: The Web has many, many databses. How do we integrate all these databases to give us a unified picture?

LaV: *Local as View*
Assume a virtual global database. Consider every local database as a *view* of the global database.

Views: *View=Query*

- Local database $L = V(D))$

Question: How do we query the global database?

View-Based Query Processing

Basic Idea: Computing the answer to a query based on a set of views, rather than on the data in a global database.

Significance: Relevant problem in query optimization, query answering with incomplete information, data warehousing, data integration, etc.

Two approaches:

- View-based query rewriting
- View-based query answering

View-Based Query Rewriting

Basic Idea: Given a query Q and a set of view definitions, reformulate Q into a new query expressed in a given language on the alphabet of the view names.

Many previous results:

- conjunctive queries
- queries with aggregates
- recursive queries
- Description Logics queries
- queries for semi-structured data

But: no general definition of query rewriting!!!

View-Based Query Answering

Basic Idea: Given a query Q, a set of view definitions, and a set of associated view extensions, compute the set of tuples t that are in the answer to Q in all the databases consistent with the views (*certain* tuples).

Many previous results:

- complexity for several query and view languages [Abiteboul et al, 98], [Grahne et al, 99]
- Description-Logics queries [Calvanese et al, 00]
- Regular-path queries [Calvanese et al, 00]

Query Answring vs. Query Rewriting

Query Answering: $\mathrm{cert}(Q, \mathrm{def}(\mathcal{V}), \mathrm{ext}(\mathcal{V})$ computes the set of certain tuples for the query Q wrt to the view definition $\mathrm{def}(\mathcal{V})$ and the view extension $\mathrm{ext}(\mathcal{V})$.

Query Rewriting: (CDLV, 2000)

- Define $\mathrm{cert}_{Q,\mathrm{def}(\mathcal{V})}$ to be the Currying of cert wrt Q and $\mathrm{def}(\mathcal{V})$.
- $\mathrm{cert}_{Q,\mathrm{def}(\mathcal{V})}$ *is* the rewriting of Q wrt $\mathrm{def}(\mathcal{V})$.

Of interest: properties of $\mathrm{cert}_{Q,\mathrm{def}(\mathcal{V})}$

- *Traditional*: expressibility in certain query language
- *New*: computational complexity

Semistructured Data

WWW:

- Nodes
- Edges
- Labels

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

Formally: $(D, E, \Lambda_+, \lambda)$

- D - nodes
- $E \subseteq D^2$ - edges
- Λ_+ – labels
- $\lambda : E \rightarrow \Lambda_+$ – labeling (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
- $a \cdot \xrightarrow{l_1} \cdots \xrightarrow{l_k} \cdot b$
- $Q(a, b)$ holds if $l_1 \cdots l_k \in L$

Example: *Regular Path Query*

$$Q(x, y) \ :- \ x \ (Wing \cdot Part^{+} \cdot Nut) \ y$$

Complexity of Query Answering for RPQs

$\mathrm{cert}(Q, \mathrm{def}(\mathcal{V}), \mathrm{ext}(\mathcal{V}))$

Complexity: (CDLV, 2000)

- *Combined complexity*: in terms of Q, $\mathrm{def}(\mathcal{V})$, and $\mathrm{ext}(\mathcal{V})$ – *PSPACE-complete*.

- *Expression complexity*: in terms of Q, and $\mathrm{def}(\mathcal{V})$ – *PSPACE-complete*.

- *Data complexity*: in terms of $\mathrm{ext}(\mathcal{V})$ – *co-NP-complete*.

Note: Complexity of $\mathrm{cert}_{Q,\mathrm{def}(\mathcal{V})}$ is data complexity of cert – *co-NP-complete*.

Questions: When is $\mathrm{cert}_{Q,\mathrm{def}(\mathcal{V})}$ in PTIME? in Datalog?

Query Rewriting and CSP

Intimate Connection: between CSP and query rewriting.

Theorem: (CDLV,2000)

- Non-uniform CSP over directed graphs is polynomially reducible to the evaluation of query rewritings.
- Evaluation of query rewritings is polynomially reducible to non-uniform CSP.

Consequences:

- Classifying tractable query rewritings is closely related to classifying non-uniform CSP – *both hard*.
- Datalog programs for non-uniform CSP can be used as *approximate* query rewritings.

From Query Answering to CSP

Given: An RPQ Q and a set $\mathcal{V}$ of RPQ views.

Construct: A constraint template $T_{Q,\mathcal{V}}$ over a vocabulary of one binary and two unary relations.

Given: An extension $\mathcal{E}$ of $\mathcal{V}$ and a pair a, b of objects.

Construct: A constraint template $I_{\mathcal{E},a,b}$.

Theorem: (CDLV, 2000)
$(a, b) \notin \mathrm{cert}(Q, \mathcal{V}, \mathcal{E})$ iff there is a homomorphism from $I_{\mathcal{E},a,b}$ to $T_{Q,\mathcal{V}}$.

In Conclusion

CSP is a database problem!!!

- CSP vs. conjunctive queries
- CSP vs. Datalog
- CSP vs. query rewriting
- CSP vs. variable-confined queries

There is a real potential for AI-DB cross fertilization.