The paper which started it all


Recent overview given by one of the founders at the gems of PODS session 2019


The core ideas in a small textbook

Slides that are adapted and extended in the following


Another nice overview on recent developments (also used here)


Will occasionally draw connections to belief revision

Own more extensive additions prefixed by “Ö.:”
Basic Notions
Integrity constraints (ICs)

- For many (DB) purposes sufficient
  - tuple-generating dependencies (TGDs)
  - equality generating dependencies (EGDs) over the same schema (Compare lectures on data integration and data exchange)

- Special cases of EGDs
  - Functional dependency (FD) $R : X \rightarrow Y$
    - If two tuples in $R$ agree on $X$ then they agree on $Y$.
  - Key Constraint
    - $R : X \rightarrow Y$ and $Y = Attributes(R) \setminus X$.

- There are further ICs ...
Common Classes of ICs (Wijsen 19))

\[ \forall \vec{x} \phi(\vec{x}) \rightarrow \bigvee_{i=1}^{n} \exists \vec{y}_i \psi_i(\vec{x}, \vec{y}_i) \]

- tgd
  - LAV tgd
    - IND
      - (no cycles w.r.t. \( \exists \) positions)
  - weakly acyclic tgd

- FOL
  - \( \forall \vec{x} (\phi(\vec{x}) \land \text{boole}(x_k = x_l)) \rightarrow \bigvee_{i=1}^{n} \psi_i(\vec{x}) \)

- full \( \forall \)-tg
  - full \( \forall \)-tg
  - \( \forall \vec{x} \phi(\vec{x}) \rightarrow \bigvee_{i=1}^{n} \psi_i(\vec{x}, \vec{y}_i) \)

- UC
  - denial/DC
    - egd \( \forall \vec{x} \phi(\vec{x}) \rightarrow x_l = x_k \)
  - full tgd
    - FD \( R : \vec{x} \rightarrow \vec{y} \)
      - key \( R : \vec{x} \rightarrow \text{attr}(R) \setminus \vec{x} \)

Convention: \( \phi(\vec{x}) \) contains exactly variables in \( \vec{x} \); \( \psi(\vec{x}), \psi_i(\vec{x}) \) may contain subvector of \( \vec{x} \)
Inconsistent Databases

- $\Sigma$: Set of integrity constraints (ICs)
Inconsistent Databases

- $\Sigma$: Set of integrity constraints (ICs)

- Inconsistent DB $\mathcal{A}$: Does not satisfy $\Sigma$, for short: $\mathcal{A} \not\models \Sigma$
Inconsistent Databases

- $\Sigma$: Set of integrity constraints (ICs)

- Inconsistent DB $\mathcal{A}$: Does not satisfy $\Sigma$, for short: $\mathcal{A} \not\models \Sigma$

- Context of and reasons for inconsistent DBs
  - Lacking support of (some) ICs
  - Heterogeneous sources with different ICs in data integration
  - Data warehouse/ETL: data to be cleaned up beforehand
Coping with Inconsistent DBs

- **Data cleaning**: Make DB consistent
  - By say adding, deleting, updating rows
  - Chase-procedure can be understood as systematic cleaning (compare notion of Null-based repairs)
  - In industrial-strength practice: ad-hoc, based on heuristics, for specific domains only
  - Main approach in industry (e.g., IBM InfoSphere Quality Stage, Microsoft DQS)

- **Database repairs**: repair (only) virtually & provide DB services
  - In particular: Enable consistent query answering (cqa) over inconsistent DBs
  - Parameters
    - Kinds of allowed repair operations (not discussed in detail)
    - Minimality notion for repairs (see next slides)
  - Compare also: Paraconsistent logics
See discussion on database update in previous lectures
Can consider DB as a model (corresponding to a complete theory)
Model describable as belief base $B$
Idea: Use $\Sigma$ as trigger info (multiple revision):
Data cleaning becomes $B \ast \Sigma$

Hence there is actually a theoretically well investigated theory (belief revision) that could be used for data cleaning
Why not used?
  Mainly due to worst-case complexity
  The same holds, imho, also for data repair (only prototypes available, not industrial-strength software)
Definition (Database Repair, (Arenas/Bertossi/Chomicki 99))

DB $\mathcal{B}$ is a repair of inconsistent DB $\mathcal{A}$ w.r.t. ICs $\Sigma$ iff

1. $\mathcal{B} \models \Sigma$
2. $\mathcal{B}$ is minimally different from $\mathcal{A}$
Definition (Database Repair, (Arenas/Bertossi/Chomicki 99))

DB $\mathcal{B}$ is a repair of inconsistent DB $\mathcal{A}$ w.r.t. ICs $\Sigma$ iff

1. $\mathcal{B} \models \Sigma$
2. $\mathcal{B}$ is minimally different from $\mathcal{A}$

- No unique definition for being minimally different
**Definition (Database Repair, (Arenas/Bertossi/Chomicki 99))**

DB $\mathcal{B}$ is a **repair** of inconsistent DB $\mathcal{A}$ w.r.t. ICs $\Sigma$ iff

1. $\mathcal{B} \models \Sigma$

2. $\mathcal{B}$ is **minimally different** from $\mathcal{A}$

- No unique definition for being minimally different
- In DB community the following instances were investigated
  - Classical set-based repair
  - Cardinality-based repairs
  - Attribute-based repairs (in particular: null-based repairs)
  - Preferred repairs
The problem of minimality in revision

- The idea of minimal repair does not work for belief sets
- Symmetric difference $X \oplus Y = X \setminus Y \cup Y \setminus X$
- Assume a propositional logic with AGM-constraints

**Proposition (Rott 2000)**

Assume that

- $X$ is a consistent belief set with $\neg \alpha \in X$
- $Y_1 \neq Y_2$ are belief sets with $\alpha \in Y_1 \cap Y_2$.

Then $X \oplus Y_1$ und $X \oplus Y_2$ are not comparable w.r.t. $\subseteq$, i.e.

$X \oplus Y_1 \not\subseteq X \oplus Y_2$ and $X \oplus Y_2 \not\subseteq X \oplus Y_1$

- Hence all repairs used for revision $X \ast \alpha$ would be $\oplus$-minimal!

DBs are rather belief bases (not deductively closed)

Hence Rott’s observation not applicable

Under what kinds of closures non-trivial minimality notion ensured?

Disjunctive closure allows minimality considerations


Definition (Classical Subset Repair)

For a set of integrity constraints $\Sigma$ and an inconsistent database $\mathcal{A}$ we say that $\mathcal{B}$ is a classical subset-repair of $\mathcal{A}$ w.r.t. $\Sigma$ iff

1. $\mathcal{B} \subseteq \mathcal{A}$
2. $\mathcal{B} \models \Sigma$
3. and there is no $\mathcal{B}'$ with properties 1. and 2. and $\mathcal{B} \subset \mathcal{B}'$
Example (Classical subset repairs)

- ICs and database

$$\Sigma = \{ \forall x \forall y \forall z ( R(x, y) \land R(x, z) ) \rightarrow y = z \}$$

$$\mathcal{A} = \{ R(a_1, b_1), R(a_1, b_2), R(a_2, b_1), R(a_2, b_2) \}$$

- Classical subset repairs

$$\mathcal{B}_1 = \{ R(a_1, b_1), R(a_2, b_1) \}$$

$$\mathcal{B}_2 = \{ R(a_1, b_1), R(a_2, b_2) \}$$

$$\mathcal{B}_3 = \{ R(a_1, b_2), R(a_2, b_1) \}$$

$$\mathcal{B}_4 = \{ R(a_1, b_2), R(a_2, b_2) \}$$

- Exponentially many repairs in general
Minimal repairs (Wijsen 19)

- Capture notions of minimal repairs with relation $\leq_{\Sigma}$

**Definition (Binary repair relation $\leq_{\Sigma}$ (informal))**

$\mathcal{B}_1 \leq_{\Sigma} \mathcal{B}_2$ iff repair of $\mathcal{A}$ into $\mathcal{B}_1$ requires no more effort than repair of $\mathcal{A}$ into $\mathcal{B}_2$

- Minimal repairs $\text{Min}_{\leq_{\Sigma}} (\mathcal{A})$ of $\mathcal{A}$:

\[
\{ \mathcal{B} \mid \mathcal{B} \models \Sigma \text{ and there is no } \mathcal{B}' \text{ s.t.: } \mathcal{B}' \models \Sigma \text{ and } \mathcal{B}' <_{\Sigma} \mathcal{B} \}
\]

- What properties to require of $<_{\Sigma}$?
  - Acyclicity
Minimal repairs w.r.t. some order

- Symmetric-difference order

\[ \mathcal{B}_1 \leq_{\mathcal{A},\oplus} \mathcal{B}_2 \text{ iff } \mathcal{B}_1 \oplus \mathcal{A} \subseteq \mathcal{B}_2 \oplus \mathcal{A} \]

- Transitive, reflexive, antisymmetric (partial order)

- Note: \( \mathcal{B}_1, \mathcal{B}_2 \) not necessarily subsets of \( \mathcal{A} \)

- Cardinality order

\[ \mathcal{B}_1 \leq_{\mathcal{A},c} \mathcal{B}_2 \text{ iff } |\mathcal{B}_1 \oplus \mathcal{A}| \leq |\mathcal{B}_2 \oplus \mathcal{A}| \]

- Transitive and reflexive (pre-order)

**Definition (General Subset [Superset] Repair)**

\( \mathcal{B} \) is a general subset-repair [superset-repair] of \( \mathcal{A} \) w.r.t. \( \Sigma \) iff it is a \( \leq_{\mathcal{A},\Sigma} \)-minimal repair (for some relation \( \leq_{\mathcal{A},\Sigma} \)) and \( \mathcal{B} \subseteq \mathcal{A} [\mathcal{B} \supseteq \mathcal{A}] \).
Preferred/Prioritized repairs

- Assume $\Sigma$ consists of FDs
- Inconsistent prioritizing db $(\mathcal{A}, \succ)$
  - Intuition: $f \succ g$ iff $f$ is fact prioritized over fact $g$
  - $\succ$ acyclic with: $f \succ g$ entails $\{f, g\} \not\models \Sigma$.

Definition (Prioritization-order and $g$-repair)

- For $\mathcal{B}_1, \mathcal{B}_2 \subseteq \mathcal{A}$, $\mathcal{B}_1 \models \Sigma$, and $\mathcal{B}_2 \models \Sigma$:
  $\mathcal{B}_1 \preceq_{\mathcal{A}, \succ} \mathcal{B}_2$ iff
  for every $g \in \mathcal{B}_2 \setminus \mathcal{B}_1$ there is $f \in \mathcal{B}_1 \setminus \mathcal{B}_2$ s.t. $f \succ g$.

- $\mathcal{B}$ is a globally optimal repair/$g$-repair iff $\mathcal{B} \in Min_{\preceq_{\mathcal{A}, \succ}}(\mathcal{A})$.

Example

- DB instance over $R(X, Y)$: $\{R(a, b), R(c, b), R(c, d)\}$
- Constraints $\Sigma$: $\{R : X \rightarrow Y, R : Y \rightarrow X\}$
Example

- DB instance over $R(X, Y)$: \{$R(a, b), R(c, b), R(c, d)$\}
- Constraints $\Sigma$: \{$R : X \rightarrow Y, R : Y \rightarrow X$\}

- $\oplus$-repairs:
  - \{\$R(a, b), R(c, d)$\}
  - \{\$R(c, b)$\}
Example

- DB instance over $R(X, Y)$: \{ $R(a, b), R(c, b), R(c, d)$ \}

Constraints $\Sigma$: \{ $R : X \rightarrow Y$, $R : Y \rightarrow X$ \}

- $\oplus$-repairs:
  - \{ $R(a, b), R(c, d)$ \}
  - \{ $R(c, b)$ \}

- C-repair
  - \{ $R(a, b), R(c, d)$ \}
Example

- DB instance over $R(X, Y)$: $\{R(a, b), R(c, b), R(c, d)\}$
- Constraints $\Sigma$: $\{R : X \rightarrow Y, R : Y \rightarrow X\}$

- $\oplus$-repairs:
  - $\{R(a, b), R(c, d)\}$
  - $\{R(c, b)\}$

- C-repair
  - $\{R(a, b), R(c, d)\}$

- G-repair for $\succ$ with $R(c, b) \succ R(a, b)$ and $R(c, b) \succ R(c, d)$
  - $\{R(c, b)\}$
Null-based repairs: tuple level

- **Input DB $A$**
  
  \[
  \{S(c_1, r_1, i_1), S(c_2, r_2, i_2), S(c_3, r_3, i_3), \quad \text{Supply(Company,Receiver,Item)} \\
  A(i_1, 50), A(i_2, 30) \} \\
  \]

- **Constraints $\Sigma$:** \[
  \forall x \forall y \forall z [S(x, y, z) \rightarrow \exists v A(z, v)] \] (an IND)

- **Repairs**
  
  - $B_1$: delete $S(c_3, r_3, i_3)$ from $A$
  - $B_2$: Insert $(i_3, \text{NULL})$ into relation $A$.

Null-based repairs: attribute level

▶ Input DB $\mathcal{A}$
  $\{R(a_4, a_3), R(a_2, a_1), R(a_3, a_3), $
  S(a_4), S(a_2), S(a_3)\}$

▶ Constraints $\Sigma$: $\{\neg \exists x \exists y (S(x) \land R(x, y) \land S(y))\}$ (a DC)

▶ Repairs
  $\mathcal{B}_1$: $\{R(a_4, a_3), R(a_2, a_1), R(a_3, a_3),$
  $S(a_4), S(a_2), S(\text{NULL})\}$
  $\mathcal{B}_2$: $\{R(a_4, \text{NULL}), R(a_2, a_1), R(a_3, \text{NULL}),$
  $S(a_4), S(a_2), S(a_3)\}$

▶ Note: Null prevents a join

Consistent Query Answering (CQA)

Definition (Arenas, Bertossi, Chomicki 99)

The consistent answers of $Q$ on $\mathcal{A}$ w.r.t. $\Sigma$ is the set

$$cqa_r(Q, \mathcal{A}, \Sigma) = \bigcap \{Q(\mathcal{B}) | \mathcal{B} \text{ is an } r\text{-repair of } \mathcal{A} \text{ w.r.t. } \Sigma\}$$

- Follows the usual pattern for dealing with incomplete information
- Compare this with certain answers in data exchange and OBDA
- When clear from context repair type $r$ not mentioned.
Example (Consistent query answering)

ICs, database, and classical subset-repairs

\[ \Sigma = \{ \forall x \forall y \forall z (R(x, y) \land R(x, z)) \rightarrow y = z \} \]

\[ \mathcal{A} = \{ R(a_1, b_1), R(a_1, b_2), R(a_2, b_1), R(a_2, b_2) \} \]

\[ \mathcal{B}_1 = \{ R(a_1, b_1), R(a_2, b_1) \} \]
\[ \mathcal{B}_2 = \{ R(a_1, b_1), R(a_2, b_2) \} \]
\[ \mathcal{B}_3 = \{ R(a_1, b_2), R(a_2, b_1) \} \]
\[ \mathcal{B}_4 = \{ R(a_1, b_2), R(a_2, b_2) \} \]

Queries and answers

\[ Q_1(x) = \exists y R(x, y) \]
\[ cqa(Q_1, \mathcal{A}, \Sigma) = \{ a_1, a_2 \} \]
\[ Q_2(x) = \exists z R(z, x) \]
\[ cqa(Q_1, \mathcal{A}, \Sigma) = \emptyset \]
Complexity and Dichotomies
Two main decision problems

Definition (Decision problem $CERTAINTY_r(Q, \Sigma)$)

- Boolean query $Q$ and $\Sigma$ fixed.
  - Input: database $\mathcal{A}$
  - Output: $cqa_r(Q, \Sigma, \mathcal{A})$

Complexity ranges from polynomial time computability to undecidability
Two main decision problems

**Definition (Decision problem \( CERTAINTY_r(Q, \Sigma) \))**

Boolean query \( Q \) and \( \Sigma \) fixed.

- Input: database \( \mathcal{A} \)
- Output: \( cqa_r(Q, \Sigma, \mathcal{A}) \)

Complexity ranges from polynomial time computability to undecidability

**Definition (Decision problem \( REPAIR_r(\Sigma) \))**

\( \Sigma \) fixed.

- Input: databases \( \mathcal{A}, \mathcal{B} \)
- Output: Is \( \mathcal{B} \) an \( r \)-repair of \( \mathcal{A} \) w.r.t. \( \Sigma \)?

Complexity ranges from polynomial time computability to coNP-completeness
Definition

- Consistent query answering for class of queries $\mathcal{Q}$ and class of integrity constraints $\mathcal{IC}$ is in complexity class $C$ iff $\text{CERTAINTY}(Q, \Sigma)$ is in $C$ for all $Q \in \mathcal{Q}$ and all $\Sigma \subseteq \mathcal{IC}$.

- Consistent query answering for $\mathcal{Q}$ and $\mathcal{IC}$ is $C$-complete iff it is in $C$ and $\text{CERTAINTY}(Q, \Sigma)$ is $C$-complete for some $Q \in \mathcal{Q}$ and some $\Sigma \subseteq \mathcal{IC}$. 
Complexities for $CERTAINTY_{\oplus}(Q, \Sigma)$ for $Q \in CQs$

What do the complexity results tell us?

- Even for very common queries and ICs untractable problems (coNP-complete problems)

- But: By definition this means only that there are some intractable (coNP-problems); does not say anything about tractable-untractable boundary

- Tackle this with dichotomy/trichotomy theorems
Complexities for $\text{REPAIR}_\oplus(\Sigma)$ for $Q \in CQs$

Complexity of CQA: An Illustration

- $R, S$: binary relations with first argument as keys:
  
  $$\Sigma = \{ R(u, v) \land R(u, w) \rightarrow v = w, S(u, v) \land S(u, w) \rightarrow v = w \}$$
Complexity of CQA: An Illustration

- \( R, S \): binary relations with first argument as keys:
  \[ \Sigma = \{ R(u, v) \land R(u, w) \rightarrow v = w, S(u, v) \land S(u, w) \rightarrow v = w \} \]

- \( Q_1 = \exists x, y, z (R(x, y) \land S(y, z)) \)
  - \( CERTAINTY(Q_1, \Sigma) \in P \)
  - Even FOL-rewritable
    \[ \exists x, y, z (R(x, y) \land S(y, z) \land \forall y'[R(x, y') \rightarrow \exists z' S(y', z')]) \]
Complexity of CQA: An Illustration

- **$R, S$:** Binary relations with first argument as keys:
  \[ \Sigma = \{ R(u, v) \land R(u, w) \rightarrow v = w, \ S(u, v) \land S(u, w) \rightarrow v = w \} \]

- **$Q_1 = \exists x, y, z (R(x, y) \land S(y, z))$:**
  - **CERTAINTY**($Q_1, \Sigma$) ∈ $P$
  - Even FOL-rewritable
    \[ \exists x, y, z (R(x, y) \land S(y, z) \land \forall y'[R(x, y') \rightarrow \exists z' S(y', z')] \]

- **$Q_2 = \exists x, y (R(x, y) \land S(y, x))$:**
  - **CERTAINTY**($Q_2, \Sigma$) ∈ $P$
  - But not FOL-rewritable
Complexity of CQA: An Illustration

- **R, S**: binary relations with first argument as keys:
  \[ \Sigma = \{ R(u, v) \land R(u, w) \rightarrow v = w, S(u, v) \land S(u, w) \rightarrow v = w \} \]

- **Q₁** = \( \exists x, y, z (R(x, y) \land S(y, z)) \)
  - **CERTAINTY** \((Q₁, \Sigma) \in P\)
  - Even FOL-rewritable
    \( \exists x, y, z (R(x, y) \land S(y, z) \land \forall y'[R(x, y') \rightarrow \exists z' S(y', z')]) \)

- **Q₂** = \( \exists x, y (R(x, y) \land S(y, x)) \)
  - **CERTAINTY** \((Q₂, \Sigma) \in P\)
  - but not FOL-rewritable

- **Q₃** = \( \exists x, y, z (R(x, y) \land S(z, y)) \)
  - **CERTAINTY** \((Q₂, \Sigma) \text{ coNP-complete}\)
Complexity of CQA: An Illustration

- \( R, S \): binary relations with first argument as keys:
  \[ \Sigma = \{ R(u, v) \land R(u, w) \rightarrow v = w, S(u, v) \land S(u, w) \rightarrow v = w \} \]

- \( Q_1 = \exists x, y, z (R(x, y) \land S(y, z)) \)
  - \( CERTAINTY(Q_1, \Sigma) \in P \)
  - Even FOL-rewritable
    \[ \exists x, y, z (R(x, y) \land S(y, z) \land \forall y'[R(x, y') \rightarrow \exists z'S(y', z')]) \]

- \( Q_2 = \exists x, y (R(x, y) \land S(y, x)) \)
  - \( CERTAINTY(Q_2, \Sigma) \in P \)
  - but not FOL-rewritable

- \( Q_3 = \exists x, y, z (R(x, y) \land S(z, y)) \)
  - \( CERTAINTY(Q_2, \Sigma) \text{ coNP-complete} \)

- Note: All queries are CQs but of different types
  \[ \rightarrow \] Classification with di-/trichotomies
Descriptive Complexity and Rewriting

- Instead of computational complexities, we can also use descriptive complexity.

- Remember the notion of logic $\mathcal{L}$ capturing a complexity class and the notion of rewriting.

- $CERTAINTY(Q, \Sigma)$ is expressible in $\mathcal{L}$, alias is $\mathcal{L}$-rewritable, iff there is $Q_{rew} \in \mathcal{L}$ s.t.

  $$cqa(Q, \mathcal{A}) = true \text{ iff } \mathcal{A} \models Q_{rew}$$

- Most attractive: $\mathcal{L} = FOL$
Famous open question

Dichotomy-Conjecture

For every set $\Sigma$ of primary keys, for every query $Q$ that is a disjunction of Boolean CQs, $\text{CERTAINTY}(Q, \Sigma)$ is either in $P$ or $\text{coNP}$-complete.
Reminder: Dichotomies

**Theorem (Ladner 1975)**

If $P \neq NP$ then there is a decision problem $Q$ s.t.
- $Q$ is in $NP$ but not in $P$
- $Q$ is not $NP$-complete

(Similar results for $coNP$ obtainable.)

The fine structure of $coNP$

| $coNP$-complete | not $coNP$-complete, not in $P$ | $P$ |
Reminder: Dichotomies

Theorem (Ladner 1975)

If $P \neq NP$ then there is a decision problem $Q$ s.t.
- $Q$ is in $NP$ but not in $P$
- $Q$ is not $NP$-complete

(Similar results for $coNP$ obtainable.)

The fine structure of $coNP$

<table>
<thead>
<tr>
<th>$coNP$-complete</th>
<th>not $coNP$-complete, not in $P$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$CERTAINTY(Q, \Sigma)$</td>
</tr>
</tbody>
</table>

Dichotomy conjecture
Theorem (Trichotomy (Koutris/Wijsen 19))

For every set $\Sigma$ of primary keys and self-join-free Boolean CQs $Q$, $\text{CERTAINTY}(Q, \Sigma)$ is either in $\text{FOL}$ or $L$-complete or $\text{coNP}$-complete.

The proof moreover reveals:

1. Membership in $L$ shown by rewriting into symmetric stratified Datalog with aggregation.
2. Membership in tractable classes ($\text{FOL} \cup L$) iff joins are foreign-key-primary-key joins

$\implies$ most SPJ queries are tractable

Connections to CSPs

- Unexpected connection between Consistent Query Answering and Constraint Satisfaction Problems (CSPs).
- Shows that the dichotomy conjecture for $CERTAINTY$ likely not trivial (as proof of CSP dichotomy highly non-trivial)

**Theorem (Fontaine 2015 (informal))**

$CERTAINTY(Q, \Sigma)$ dichotomy (Conjecture) $\models$ CSP dichotomy under specific conditions

Constrain Satisfaction Problems

- Traditionally (as used in AI research) considered as subclass of search problems with states and a goal test

**Definition**

- \( D = \) domain
- \( \Gamma = \) Constraint language = set of relations \( \{R_i\}_{i \in I} \) over \( D \)

The constraint satisfaction problem \( CSP(\Gamma) \) associated with \( \Gamma \) is the problem defined by instances of the form \((V, D, C)\) where

- \( V = \) set of variables
- \( C = \) set of constraints \((\vec{v}, R_i)\) (notated also as \( R_i(\vec{v}) \)) with
  - \( R_i \in \Gamma \) an \( n \)-ary relation
  - \( \vec{v} = (v_1, \ldots, v_n) \) with \( v_i \in V \) is the scope (state variables)

A solution to the problem (the goal) is a mapping \( \phi : V \rightarrow D \) fulfilling all constraints, i.e., \( \phi(\vec{v}) \in R_i \) for all \((\vec{v}, R_i) \in C\).
Example: Map Colouring

- \( D = \{ \text{red, green, blue} \} \)
- \( \Gamma = \{ R_1 \} = \{ \{ (x, y) \in D \times D \mid x \neq y \} \} \)
- \( V = \{ \text{WA, NT, Q, NSW, VI, SA, T} \} \)
- \( C = \{ \text{WA} \neq \text{NT}, \text{WA} \neq \text{SA}, \text{NA} \neq \text{SA}, \text{NA} \neq \text{Q}, \text{SA} \neq \text{Q}, \text{SA} \neq \text{NSW}, \text{SA} \neq \text{VI}, \text{Q} \neq \text{NSW}, \text{NSW} \neq \text{VI} \} \)

- A solution

\[ \phi : \quad \text{WA} \mapsto \text{red}, \quad \text{NT} \mapsto \text{green}, \quad \text{SA} \mapsto \text{blue}, \quad \text{Q} \mapsto \text{red}, \quad \text{NSW} \mapsto \text{green}, \quad \text{VI} \mapsto \text{red}, \quad \text{SA} \mapsto \text{blue}, \quad \text{T} \mapsto \text{green} \]
CSP = Finding a homomorphism

**Definition**

Given a CSP($\Gamma$) instance ($V$, $D$, $C$) the associated homomorphism instance $h : S \xrightarrow{\text{hom}} T$ is defined by

- **source structure** $S = (\text{variables, scopes})$
  $S = (V, R^S_i = \{\vec{v} \mid (\vec{c}, R) \in C\})_{i \in I}$

- **target structure** $T = (\text{values, constraint relations})$
  $T = \{D, R^T_i = R_i\}_{i \in I}$

- **homomorphism** $h = \text{solution } \phi$

Each homomorphism problem $\exists h : A \xrightarrow{\text{hom}} B$ gives rise to a CSP-instance: Generate constraint ($\vec{v}, R^B$) for each $\vec{v} \in R^A$.

**Lit:** T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. SIAM J. Comput., 28:57–104, 199

Dichotomies for CSP

- Complexity of finding solutions depends on $\Gamma$
- Dichotomy theorem for subclass of conservative CSPs (c-CSP) which are CSPs with additionally:
  - For each variable $v \in V$ one has a unary relation $R_v \subseteq D$
  - Solution $\phi$ must fulfill $\phi(v) \in R_v$.

Theorem (Dichotomy for conservative CSPs, (Bulatov 11))

For each $\Gamma$ the problem $c-CSP(\Gamma)$ is either in $P$ or $NP$-complete.

- Proof relies on algebraic machinery based on polymorphisms (Barto et al., 17)


Connections to CSPs

Theorem (Fontaine 2015 formal)

▶ \textit{CERTAINTY}(Q, \Sigma) \textit{ dichotomy} (Conjecture) \models CSP dichotomy where

▶ \Sigma \text{ is a finite set of Horn constraints (full tgd with atomic head)}
▶ Q \text{ is a union of Boolean CQs}

▶ \textit{CERTAINTY}(Q, \Sigma) \textit{ dichotomy} (Conjecture) \models c\text{-CSP dichotomy}

Alternative route for proving the conjecture

- Breakthrough result in 2017 for wider class of CSP

**Theorem (Dichotomy for CSPs, (Bulatov 17))**

*For each $\Gamma$ the problem $\text{CSP}(\Gamma)$ is either in $P$ or $NP$-complete.*

- Proof strategy for conjecture: Show that it is entailed by Bulatov’s dichotomy for CSPs.

Systems
Theory and Practice

- Theory of database repairs a theoretical foundation for coping with inconsistent DBs

- Extensively studied in last 20 years

- Only marginally used in data cleaning (few examples given by (Bertossi 19)

- Industrial-strength cqa-systems have yet to be developed

Prototypes (optional slide)

- Hippo
  

- ConQuer
  

- ConsEx
  

- EQUIP
  
# Feature Overview (Optional Slide)

<table>
<thead>
<tr>
<th>System</th>
<th>Constraints</th>
<th>Queries</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hippo</td>
<td>UC</td>
<td>Projection-free with $\cup$ and $\setminus$</td>
<td>Direct Alg.</td>
</tr>
<tr>
<td>ConQuer</td>
<td>Keys</td>
<td>CQs</td>
<td>FO-rewriting</td>
</tr>
<tr>
<td>ConsEx</td>
<td>UC + IND with acyclicity</td>
<td>Datalog $\neg$</td>
<td>ASP</td>
</tr>
<tr>
<td>EQUIP</td>
<td>Keys IND</td>
<td>CQs</td>
<td>Reduction to ILP</td>
</tr>
</tbody>
</table>
Kolaitis’ vision for a comprehensive system (Optional Slide)

Module-based system depending on complexity of \( cqa(Q, \Sigma) \)

- **Preprocessing**: Determine evaluation strategy based on complexity classification for \( cqa(Q, \Sigma) \)

- **Processing**
  - Module A: FOL-rewriting + DB engine if \( cqa(Q, \Sigma) \) FOL rewritable
  - Module B: Direct Algorithm or reduction to LP if \( cqa(Q, \Sigma) \in P \setminus FOL \)
  - Module C: Reduction to ILP (oe SAT or QBF) if \( cqa(Q, \Sigma) \in \text{coNP} \).
Lifting to ontologies

- Whole idea can of course be lifted also to ontologies

- For a recent contribution for DL-Lite ontologies see (Bienvenu et al 2019)

Synopsis and Outlook (Kolaitis + Ö.)

▶ Database repair meeting point for database, logic, and complexity

▶ Further dichotomies; main conjecture open

▶ Much to be done for industrial-strength systems for different types of repairs and classes of constraints
  ▶ Promising approach: Combine database engines with SAT solvers and QBF solvers
  ▶ Ö: This fits to the general trend of “SQL-incorporates it all’
    ▶ SQL now supports arrays
    ▶ SQL is going to give support for streams ...

▶ Ö: Systematic study of connections to belief revision
▶ Ö: Are there dichotomies also for belief revision?