REPLACE: A Logical Framework for Combining Collective Entity Resolution and Repairing

Meghyn Bienvenu[1] , **Gianluca Cima**[2] and **Víctor Gutiérrez-Basulto**[3]
[1]Univ. Bordeaux, CNRS, Bordeaux INP, LaBRI, UMR 5800
[2]Department of Computer, Control and Management Engineering, Sapienza University of Rome
[3]School of Computer Science & Informatics, Cardiff University
meghyn.bienvenu@labri.fr, cima@diag.uniroma1.it, gutierrezbasultov@cardiff.ac.uk

Abstract

This paper considers the problem of querying dirty databases, which may contain both erroneous facts and multiple names for the same entity. While both of these data quality issues have been widely studied in isolation, our contribution is a holistic framework for jointly deduplicating and repairing data. Our REPLACE framework follows a declarative approach, utilizing logical rules to specify under which conditions a pair of entity references can or must be merged and logical constraints to specify consistency requirements. The semantics defines a space of solutions, each consisting of a set of merges to perform and a set of facts to delete, which can be further refined by applying optimality criteria. As there may be multiple optimal solutions, we use classical notions of possible and certain query answers to reason over the alternative solutions, and introduce a novel notion of most informative answer to obtain a more compact presentation of query results. We perform a detailed analysis of the data complexity of the central reasoning tasks of recognizing optimal solutions and (most informative) possible and certain answers, for each of the three notions of optimal solution and for both general and restricted specifications.

1 Introduction

Data quality is one of the most fundamental problems in data management as dirty data can lead to incorrect decisions based on faulty retrieved answers from information systems, and unreliable data analysis, engendering huge costs to the private and public sectors [Fan and Geerts, 2012; Ilyas and Chu, 2019]. It is also a multi-faceted problem, encompassing several distinct issues: multiple representations of the same entity (deduplication), conflicting and/or erroneous information (consistency / accuracy), missing information (completeness), and outdated information (currency) [Fan and Geerts, 2012]. While far from solved, each facet of the data quality problem has given rise to a sizeable literature and increasingly sophisticated methods. We give a brief (and necessarily incomplete) introduction to the issues central to our work: deduplication and consistency.

Deduplication, also called record linkage or entity resolution (ER), was originally formulated as the task of identifying duplicate records in a table, and traditionally handled by comparing attribute values using similarity measures [Newcombe *et al.*, 1959]. Over time, however, variants of the problem have been explored, in which we may identify (match, merge) pairs of entity references or values, rather than whole tuples, and treat multiple tables and/or entity types together (so-called collective entity resolution [Bhattacharya and Getoor, 2007]), e.g. using a match of two authors infer that a pair of paper ids should be merged. Moreover, some recent approaches to collective ER [Deng *et al.*, 2022; Bienvenu *et al.*, 2022] are able to exploit recursive dependencies, e.g. a merge of authors may trigger merges of papers which in turn may trigger new author merges. Diverse techniques have been applied to (collective) ER, including similarity measures, deep learning, probabilistic formalisms, and declarative frameworks based upon logical rules and constraints, see [Christophides *et al.*, 2021] for a recent survey.

There is likewise a vast body of work aimed at identifying and removing conflicting facts to restore consistency. Declarative constraints (such as functional dependencies, or the broader class of denial constraints) are often employed to specify consistency requirements [Chu *et al.*, 2013], and the goal is to produce a consistent version of the data (called a repair) through deletion or modification of facts. However, due to lack of information, one typically cannot definitively identify the 'true' repair, so data cleaning often relies upon heuristics to produce a unique result [Ilyas and Chu, 2019]. By contrast, the well-known consistent query answering (CQA) paradigm [Arenas *et al.*, 1999; Bertossi, 2011] allows for meaningful answers to be obtained without committing to a single repair, by reasoning over the space of all (preferred) repairs and returning those answers which hold w.r.t. every such repair. The approach follows the skeptical mode of reasoning often considered in knowledge representation and reasoning (KR), and it has in turn inspired a line of KR research on inconsistency-tolerant ontology-based data access [Bienvenu, 2020; Lukasiewicz *et al.*, 2022]. While CQA has higher complexity than data cleaning methods, SAT-based implementations show promising results [Dixit and Kolaitis, 2022].

The different facets of the data quality problem have mostly been considered in isolation, whereas in practice, datasets can be expected to suffer from multiple data quality issues. A pipeline approach, which applies different methods in sequence, has the disadvantage that useful synergies may be missed, as noted in [Chu *et al.*, 2013; Fan *et al.*, 2014]. For example, by merging two constants, we may resolve a violation of a functional dependency (FD) without the need to delete facts, while conversely, by deleting incorrect facts, we may enable some desirable merges. The interest of combining ER with repairs has been advocated in [Fan *et al.*, 2014]: "When taken together, record matching and data repairing perform much better than being treated as separate processes". They propose to interleave repair operations (value updates) with merges of values inferred using matching dependencies, and study when this combined process terminates and how to generate a single repair of optimal cost.

We believe that the development of holistic approaches for jointly tackling the ER and repairing tasks, pioneered in [Fan *et al.*, 2014], merits further investigation. Indeed, given the vast number of different ER and repair methods, there are many options for which methods to use and how to integrate them. Moreover, to the best of our knowledge, no work has explored how to reason over a space of alternative solutions (in the spirit of CQA) for the combined task. These considerations motivate us to introduce REPLACE, a logic-based framework for collective entity resolution and repairing.

The REPLACE framework adopts the expressive class of denial constraints (which generalize the conditional FDs considered in [Fan *et al.*, 2014]) and subset repairs (obtained via deletion of facts, rather than updates), the most commonly considered repair notion in the CQA literature. The ER mechanism in REPLACE is based on the recently proposed LACE framework [Bienvenu *et al.*, 2022], which employs hard and soft rules to define mandatory and possible merges of constants. Differently from [Fan *et al.*, 2014] and other works using matching dependencies [Fan *et al.*, 2009; Bertossi *et al.*, 2013], the semantics is global in the sense that we merge *all* occurrences of the matched constants [Arasu *et al.*, 2009; Burdick *et al.*, 2016], rather than only those constant occurrences used in deriving the match. Such a semantics is geared towards merging of constants that are entity references (e.g. authors, publications), whereas the local one is more appropriate for merging attribute values (e.g. titles and addresses) (see [Bienvenu *et al.*, 2022] for a detailed discussion). REPLACE's semantics further follows the standard desiderata of maximizing merges and minimizing deletions. As these two criteria may conflict, REPLACE implements three natural ways to compare solutions: give priority to the maximization of merges (MER), give priority to the minimization of deletions (DEL), or adopt the Pareto principle (PAR).

Aside from introducing the new framework, our main contribution is the investigation of the data complexity of the main reasoning tasks associated with REPLACE. First, we show that the problem of recognizing optimal solutions is coNP-complete, for all three optimality notions. Next, we consider how to query the space of optimal solutions and determine the complexity of recognizing *certain* and *possible* query answers, i.e. those answers which hold in all or some optimal solution, respectively. The certain answer tasks are Π_2^p-complete for the three optimality notions, while for possible answers, the recognition problem Σ_2^p-complete for MER and DEL, but NP-complete for PAR. We further consider a restricted setting in which inequality atoms are disallowed in denial constraints. This restriction does not yield better complexity for these problems, if we consider the MER preorder, whereas for DEL and PAR, the complexity improves in almost all cases. As a further contribution, we introduce a novel notion of most informative answer to obtain a more compact presentation of query results and show that the improved format leads to a slight increase in the complexity of certain and possible answer recognition tasks. We conclude the paper with some directions for future work.

2 Preliminaries

A *(relational) schema* $\mathcal{S}$ is a finite set of relation symbols, with each $R \in \mathcal{S}$ having an associated arity and list of attributes. As is standard, we use R/k and $R(A_1, \ldots, A_k)$ to indicate, respectively, that R has arity k and that its attributes are $A_1, \ldots, A_k$. A *database instance over a schema* $\mathcal{S}$ (or *($\mathcal{S}$-)database* for short) assigns to each k-ary relation symbol $R \in \mathcal{S}$ a finite k-ary relation over a fixed, denumerable set of constants. Equivalently, we view an $\mathcal{S}$-database D as a finite set of *facts* of the form $R(c_1, \ldots, c_k)$, where $(c_1, \ldots, c_k)$ is a tuple of constants of the same arity as R. We use the notations $R(c_1, \ldots, c_k) \in D$ and $D \subseteq D'$ with their obvious meanings. The *active domain* of a database D, denoted by $\mathsf{dom}(D)$, is the set of constants occurring in D.

When we speak of queries in this paper, unless otherwise stated, we mean a *conjunctive query (CQ)*. Recall that a *CQ over a schema* $\mathcal{S}$ takes the form $q(\mathbf{x}) = \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ and $\mathbf{y}$ are disjoint lists of variables, and φ is a finite conjunction of relational atoms over $\mathcal{S}$, i.e. atoms of the form $R(t_1, \ldots, t_k)$ with $R \in \mathcal{S}$ and each t_i is either a constant or a variable from $\mathbf{x} \cup \mathbf{y}$. The *arity* of a query $q(\mathbf{x})$ is the arity of $\mathbf{x}$, and a query with arity 0 is called *Boolean*. Given an n-ary query $q(x_1, \ldots, x_n)$ and n-tuple of constants $\mathbf{c} = (c_1, \ldots, c_n)$, we denote by $q[\mathbf{c}]$ the Boolean query obtained by replacing each x_i by c_i. The *answers to an n-ary query $q(\mathbf{x})$ over a database D* is defined as the set of n-tuples of constants $\mathbf{c}$ from $\mathsf{dom}(D)$ such that the Boolean CQ $q[\mathbf{c}]$ holds in D. We use $q(D)$ to denote the answers to q over D.

When formulating entity resolution rules, we will consider queries that may also contain atoms built from a set of externally defined binary *similarity predicates*. The preceding definitions and notations extend to such queries, the only difference being that similarity predicates have a fixed meaning (typically defined by applying a similarity metric, e.g. edit distance, and keeping those pairs of values whose score exceeds a given threshold).

Our framework will also make use of *denial constraints* [Bertossi, 2011; Fan and Geerts, 2012]. Recall that a *denial constraint over a schema* $\mathcal{S}$ takes the form $\forall \mathbf{x}. \neg(\phi(\mathbf{x}))$, where $\phi(\mathbf{x})$ is a finite conjunction of relational atoms over $\mathcal{S}$ and inequality atoms $t_1 \neq t_2$.

3 Existing LACE Framework

In this section, we recall the salient features and definitions of the LACE framework [Bienvenu *et al.*, 2022] for collective entity resolution, as it will form the basis for our new REPLACE framework, presented in Section 4.

Entity resolution consists in determining pairs of database constants that refer to the same entity and can thus be identified. We will use the term *merge* to speak about such pairs. The LACE framework employs hard and soft rules to indicate, respectively, required or potential merges. A *hard rule (w.r.t. a schema $\mathcal{S}$)* takes the form $q(x, y) \Rightarrow \mathsf{EQ}(x, y)$, where $q(x, y)$ is a CQ, whose atoms may use relation symbols in $\mathcal{S}$ as well as similarity predicates, and EQ is a special relation symbol (not in $\mathcal{S}$) used to store merges. Intuitively, such a rule states that (c_1, c_2) being an answer to q is sufficient to conclude that c_1 and c_2 refer to the same entity. A *soft rule* has a similar form: $q(x, y) \dashrightarrow \mathsf{EQ}(x, y)$, but states instead that (c_1, c_2) being an answer to q provides reasonable evidence for c_1 and c_2 denoting the same entity. Soft rules suggest potential (but not mandatory) merges of constants. In what follows, we use the notation $q(x, y) \rightarrow \mathsf{EQ}(x, y)$ for a generic (hard or soft) rule, and shall omit quantifiers in rule bodies for brevity.

In addition to rules for generating merges, the LACE framework employs denial constraints to define consistency requirements. Together they form a specification:

Definition 1 ([Bienvenu *et al.*, 2022])**.** *A* data quality (DQ) specification Σ *over a schema $\mathcal{S}$ takes the form $\Sigma = \langle \Gamma, \Delta \rangle$, where $\Gamma = \Gamma_h \cup \Gamma_s$ is a finite set of hard and soft rules over $\mathcal{S}$, and Δ is a finite set of denial constraints over $\mathcal{S}$.*

Example 1. *Figure 1 introduces the schema $\mathcal{S}_{\mathsf{ex}}$, database D_{ex}, and DQ specification $\Sigma_{\mathsf{ex}} = \langle \Gamma_{\mathsf{ex}}, \Delta_{\mathsf{ex}} \rangle$ of our running example. Informally, the hard rule ρ_1 states that paper identifiers with similar titles, same year, same venue, same first author, and same conference chair must refer to the same paper. The soft rule σ_1 states that author ids associated with similar emails and the same institution likely refer to the same person. Finally, the denial constraint δ_1 enforces that there is a single chair for a given venue and year, while δ_2 states that the first author of a paper cannot be the same as the chair of the event where the paper was published.*

The semantics of LACE is based upon solutions, which take the form of equivalence relations over the constants, with the meaning that all constants from the same equivalence class are deemed to be references to the same entity. Solutions equate constants, rather than occurrences of constants, because LACE focuses specifically on merging constants that are entity references (e.g. paper and author ids). Intuitively, each solution is obtained by 'deriving' new merges via rule applications and closure operations. Importantly, rule bodies are evaluated on the database induced by previously derived merges, which makes it possible for new rules to become applicable, i.e. merges can enable additional merges.

In order to formally define solutions, we must first introduce some preliminary notions. Given a set S of pairs of constants from a database D, we denote by $\mathsf{EqRel}(S, D)$ the least equivalence relation $E \supseteq S$ over $\mathsf{dom}(D)$, i.e. we close S under reflexivity, symmetry, and transitivity. We assume that each equivalence relation E is equipped with a function rp_E that maps each element to a representative of its equivalence class in E. Given a database D and an equivalence relation E over $\mathsf{dom}(D)$, the *database induced by D and E*, denoted by D_E, is the database obtained from D by replacing each constant c by $\mathsf{rp}_E(c)$. Moreover, for a tuple $\mathbf{c}$ of constants (resp. query q, denial constraint δ), we denote by $\mathbf{c}_E$ (resp. q_E, δ_E) the tuple of constants (resp. query, denial constraint) obtained by replacing each constant c mentioned also in an equivalence relation E by $\mathsf{rp}_E(c)$. We then define the set $q(D, E)$ of *answers to a query $q(\mathbf{x})$ w.r.t. D and E* as:

$$\mathbf{c} \in q(D, E) \text{ iff } \mathbf{c}_E \in q_E(D_E)$$

A set of denial constraints Δ is satisfied in (D, E), written $(D, E) \models \Delta$, if $D_E \models \delta_E$ for every $\delta \in \Delta$. A rule $\gamma = q(x, y) \rightarrow \mathsf{EQ}(x, y) \in \Gamma$ is satisfied in (D, E), written $(D, E) \models \gamma$, if $q(D, E) \subseteq E$, and $(D, E) \models \Gamma'$ if all rules in $\Gamma' \subseteq \Gamma$ are satisfied. We call a pair (c, c') of constants *active in (D, E) w.r.t.* Γ if there exists a rule $q(x, y) \rightarrow \mathsf{EQ}(x, y) \in \Gamma$ such that $(c, c') \in q(D, E)$.

Remark 1. *There is in fact an additional syntactic condition placed on LACE rulesets (and which we shall adopt also in this paper), namely, that attributes that are involved in merges cannot participate in similarity atoms. We refer to [Bienvenu* et al.*, 2022] for a formal definition and discussion, simply noting that this condition ensures an unambiguous evaluation of similarity atoms in induced databases.*

We can now give the formal definition of LACE solutions:

Definition 2 ([Bienvenu *et al.*, 2022])**.** *Given a DQ specification Σ over a schema $\mathcal{S}$ and an $\mathcal{S}$-database D, we say that an equivalence relation E over* $\mathsf{dom}(D)$ *is an* ER candidate solution for (D, Σ) *if it satisfies one of the two conditions:*

(i) $E = \mathsf{EqRel}(\emptyset, D)$*;*

(ii) $E = \mathsf{EqRel}(E' \cup \{\alpha\}, D)$*, where E' is a candidate solution for (D, Σ) and $\alpha = (c_1, c_2)$ is active in (D, E') w.r.t. Γ.*

An ER solution *for (D, Σ) is a candidate solution E that further satisfies (a) $(D, E) \models \Gamma_h$ and (b) $(D, E) \models \Delta$. We denote by $\mathsf{ERSol}(D, \Sigma)$ the set of ER solutions for (D, Σ).*

Notice that each pair of constants that is deemed equivalent by the ER solution is obtained by a sequence of rule applications and closure operations. Moreover, solutions must be coherent in the sense that all of the hard rules and denial constraints have to be satisfied w.r.t. the induced database.

Example 2. *Continuing our running example, let D'_{ex} be the $\mathcal{S}_{\mathsf{ex}}$-database obtained from D_{ex} by removing the tuples regarding papers p_1, p_2, p_3, and p_4. Due to the tuples involving p_6, p_7, and p_8, we have $(D'_{\mathsf{ex}}, E_0) \not\models \delta_1$, for the initial relation $E_0 = \mathsf{EqRel}(\emptyset, D'_{\mathsf{ex}})$. However, we can resolve this violation by merging authors a_4 and a_5. Indeed, one can verify that $\epsilon = (a_4, a_5)$ is active in (D'_{ex}, E_0) w.r.t. Γ_{ex} due to σ_1. Also $\alpha = (a_1, a_2)$ and $\beta = (a_2, a_3)$ are active due to σ_1.*

However, we cannot include both α and β, otherwise by transitivity we would have $a_1 = a_3$, implying that the first author of paper p_5 would be the same as the chair, in violation of δ_2. Now, consider $E_1 = \mathsf{EqRel}(\{\beta, \epsilon\}, D'_{\mathsf{ex}})$ and $E_2 = \mathsf{EqRel}(\{\alpha, \epsilon\}, D'_{\mathsf{ex}})$. While $E_1 \in \mathsf{ERSol}(D'_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$, we have $E_2 \notin \mathsf{ERSol}(D'_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$. This is because $(D'_{\mathsf{ex}}, E_2) \not\models \rho_1$ since $\zeta = (p_6, p_7)$ is now active in (D'_{ex}, E_2) w.r.t. Γ_{ex}. One can verify that E_1 and $E_3 = \mathsf{EqRel}(\{\alpha, \epsilon, \zeta\}, D'_{\mathsf{ex}})$ are the

Author(aid, email, inst)

aid	email	inst
a_1	wtaka@gm.com	Tokyo
a_2	wtaka@tku.jp	Tokyo
a_3	takaw@tku.jp	Tokyo
a_4	mnk@ox.uk	NYU
a_5	mnk@gm.com	NYU
a_6	ropaolo@sap.it	Sap

Paper(pid, title, fid, year, venue, cid)

pid	title	fid	year	venue	cid
p_1	Computational Complexity of CQA	a_6	2009	IJCAI	a_1
p_2	CQA: Computational Complexity	a_6	2009	IJCAI	a_2
p_3	A Framework for Collective ER	a_1	2010	PODS	a_2
p_4	A Logical Framework for Collective ER	a_1	2010	PODS	a_2
p_5	Answering CQs over DL Ontologies	a_1	2012	KR	a_3
p_6	AI Techniques for ER	a_2	2023	AAAI	a_5
p_7	AI Techniques for Collective ER	a_1	2023	AAAI	a_4
p_8	Logical Techniques for Collective ER	a_3	2023	AAAI	a_5

$$\delta_1 = \neg(\exists p, t, f, y, v, c, p', t', f', c'.\text{Paper}(p, t, f, y, v, c) \wedge \text{Paper}(p', t', f', y, v, c') \wedge c \neq c')$$
$$\delta_2 = \neg(\exists p, t, a, y, v.\text{Paper}(p, t, a, y, v, a))$$
$$\rho_1 = \text{Paper}(x, t, f, y, v, c) \wedge \text{Paper}(y, t', f, y, v, c) \wedge t \approx_1 t' \Rightarrow \mathsf{EQ}(x, y)$$
$$\sigma_1 = \text{Author}(x, e, i) \wedge \text{Author}(y, e', i) \wedge e \approx_2 e' \dashrightarrow \mathsf{EQ}(x, y)$$

Figure 1: A schema $\mathcal{S}_{\mathsf{ex}}$, $\mathcal{S}_{\mathsf{ex}}$-database D_{ex}, and DQ specification $\Sigma_{\mathsf{ex}} = \langle \Gamma_{\mathsf{ex}}, \Delta_{\mathsf{ex}} \rangle$ over $\mathcal{S}_{\mathsf{ex}}$ with $\Gamma_{\mathsf{ex}} = \{\rho_1, \sigma_1\}$ and $\Delta_{\mathsf{ex}} = \{\delta_1, \delta_2\}$. The extension of the similarity predicates $\approx_1$ and $\approx_2$ (both restricted to $\mathsf{dom}(D_{\mathsf{ex}})$) are the symmetric and reflexive closures of $\{(e_1, e_2), (e_2, e_3), (e_4, e_5)\}$ and $\{(t_1, t_2), (t_3, t_4), (t_6, t_7), (t_7, t_8)\}$, respectively, where e_i and t_i are the email of author a_i and title of paper p_i, respectively.

only maximal ER solutions *for* $\mathsf{ERSol}(D'_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$*, i.e. they belong to* $\mathsf{ERSol}(D'_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *and there is no other solution in* $\mathsf{ERSol}(D'_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *containing strictly more merges.*

Now reconsider the original database D_{ex}*. One can verify that* $\mathsf{Sol}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}}) = \emptyset$*. This is because the tuples with* p_1 *and* p_2 *violate* δ_1*, and, if* a_1 *and* a_2 *are merged to solve this violation, then* δ_2 *is violated due to the tuples with* p_3 *and* p_4*.*

4 REPLACE: Adding Delete Operations

In practice, a given database may suffer from multiple data quality issues. Some constraint violations may result from the use of different constants for the same entity, and thus may be resolved through merging constants. However, other constraint violations stem from the presence of erroneous facts and can only be resolved by removing information. In this section, we introduce a holistic approach to data quality that allows for both merge and fact deletion operations. Our new REPLACE framework can be viewed as the marriage of LACE with the well-known consistent query answering approach.

Extending LACE with fact deletions allows us to obtain meaningful solutions when $\mathsf{ERSol}(D, \Sigma) = \emptyset$, but also to discover merges that were blocked due to constraint violations:

Example 3. *Recall the database* D'_{ex} *from the previous example and observe that by removing the fact with pid* p_5*, we can now include both* $\alpha = (a_1, a_2)$ *and* $\beta = (a_2, a_3)$ *in the set of merges, which will lead to* $\eta = (p_7, p_8)$ *being active.*

The REPLACE framework adopts the DQ specifications from LACE, but redefines what constitutes a solution to a database-specification (D, Σ) pair. In addition to an equivalence relation E that specifies merges, solutions will additionally contain a set R of facts to delete from D. We shall require that (*i*) $E \in \mathsf{ERSol}(D \setminus R, \Sigma)$, i.e. E is an ER solution for $(D \setminus R, \Sigma)$ and (*ii*) if a fact $\varphi \in R$ is equivalent to a fact $\psi \in D$ w.r.t. E, then $\psi \in R$. A fact $\varphi = P(\mathbf{c})$ is said to be *equivalent to* $\psi = P'(\mathbf{c}')$ *w.r.t.* E, denoted $\varphi \equiv_E \psi$, if $P = P'$ and $\mathbf{c}_E = \mathbf{c}'_E$.

We are now ready to formally define the new notion of solutions employed by REPLACE:

Definition 3. *Given a DQ specification* Σ *over a schema* $\mathcal{S}$ *and an* $\mathcal{S}$*-database* D*, we say that a pair* $W = (R, E)$ *is a* solution for (D, Σ) *if* (i) $R \subseteq D$*,* (ii) $E \in \mathsf{ERSol}(D \setminus R, \Sigma)$*, and* (iii) *for all* $\varphi, \psi \in D$ *with* $\varphi \equiv_E \psi$*,* $\varphi \in R$ *iff* $\psi \in R$*. We denote by* $\mathsf{Sol}(D, \Sigma)$ *the set of solutions for* (D, Σ)*.*

Example 4. *Let* $W_1 = (R_1, E_1)$ *and* $W_2 = (R_2, E_2)$ *be such that* R_1 *(resp.* R_2*) consists of the Paper fact with pid* p_1 *(resp.* p_2*),* $E_1 = \mathsf{EqRel}(\{\beta, \epsilon, \theta\}, (D_{\mathsf{ex}} \setminus R_1))$*, and* $E_2 = \mathsf{EqRel}(\{\beta, \epsilon, \theta\}, (D_{\mathsf{ex}} \setminus R_2)))$*, where* $\beta = (a_2, a_3)$*,* $\epsilon = (a_4, a_5)$*, and* $\theta = (p_3, p_4)$*. One can verify that* $W_1 \in \mathsf{Sol}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *and* $W_2 \in \mathsf{Sol}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$*.*

Rather than considering all solutions, it is natural to focus on the 'best' ones. But what makes a solution better than another? Similarly to LACE, we will prefer solutions that contain more merges, since we aim to tackle the ER problem. However, we also want to retain as much information as possible, hence should minimize fact deletions, as is done when defining repairs. These two criteria may conflict, as deleting more facts may enable more merges. This leads us to consider three natural ways to compare solutions: give priority to the maximization of merges (MER), give priority to the minimization of deletions (DEL), or adopt the Pareto principle and accord equal priority to both criteria (PAR). The following definition formalizes the three preorders for comparing solutions and the resulting notions of optimal solution, using set inclusion for comparing the sets of merges and deletions.

Definition 4. *Consider a DQ specification* Σ *over schema* $\mathcal{S}$ *and* $\mathcal{S}$*-database* D*. The preorders* $\prec_{\text{MER}}$*,* $\prec_{\text{DEL}}$*, and* $\prec_{\text{PAR}}$ *over* $\mathsf{Sol}(D, \Sigma)$ *are defined as follows:*

- $(R, E) \prec_{\text{MER}} (R', E')$ *iff either* (i) $E \subset E'$ *or* (ii) $E \subseteq E'$ *and* $R' \subset R$*;*
- $(R, E) \prec_{\text{DEL}} (R', E')$ *iff either* (i) $R' \subset R$ *or* (ii) $R' \subseteq R$ *and* $E \subset E'$*;*
- $(R, E) \prec_{\text{PAR}} (R', E')$ *iff either* (i) $E \subset E'$ *and* $R' \subseteq R$ *or* (ii) $R' \subset R$ *and* $E \subseteq E'$*.*

*For $X \in \{\textsc{Mer}, \textsc{Del}, \textsc{Par}\}$, we call a solution W for (D, Σ) an $\preceq_X$-*optimal solution *for (D, Σ) if there is no solution W' for (D, Σ) such that $W \prec_X W'$, and denote by $\mathsf{Sol}_X(D, \Sigma)$ the set of $\preceq_X$-optimal solutions for (D, Σ).*

It is easy to verify that both $\mathsf{Sol}_{\textsc{Mer}}(D, \Sigma) \subseteq \mathsf{Sol}_{\textsc{Par}}(D, \Sigma)$ and $\mathsf{Sol}_{\textsc{Del}}(D, \Sigma) \subseteq \mathsf{Sol}_{\textsc{Par}}(D, \Sigma)$ hold for any database-specification pair (D, Σ). The next example shows that the converse inclusions do not necessarily hold. Furthermore, using analogous arguments, it is not hard to construct a case where $W \in \mathsf{Sol}_{\textsc{Par}}(D, \Sigma)$ but neither $W \in \mathsf{Sol}_{\textsc{Mer}}(D, \Sigma)$ nor $W \in \mathsf{Sol}_{\textsc{Del}}(D, \Sigma)$.

Example 5. *Returning to our running example, it can be verified that W_1 and W_2 from Example 4 both belong to $\mathsf{Sol}_X(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ for each $X \in \{\textsc{Mer}, \textsc{Del}, \textsc{Par}\}$.*

Next consider $W_3 = (R_3, E_3)$, in which R_3 consists of the tuples with pids p_3 and p_4 and $E_3 = \mathsf{EqRel}(\{\alpha, \mu, \epsilon, \zeta, \}, (D_{\mathsf{ex}} \setminus R_3))$, where $\alpha = (a_1, a_2)$, $\mu = (p_1, p_2)$, $\epsilon = (a_4, a_5)$, and $\zeta = (p_6, p_7)$. One can show that $W_3 \in \mathsf{Sol}_{\textsc{Del}}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ (hence, $W_3 \in \mathsf{Sol}_{\textsc{Par}}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$) because the violation of δ_1 involving the p_1 and p_2 tuples is resolved by merging a_1 and a_2, rather than via deletion. We claim however that $W_3 \notin \mathsf{Sol}_{\textsc{Mer}}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$. To see why, let $W_4 = (R_4, E_4)$ be such that R_4 contains the tuples with pids p_3, p_4, and p_5 and $E_4 = \mathsf{EqRel}(\{\alpha, \mu, \beta, \epsilon, \zeta, \eta\}, (D_{\mathsf{ex}} \setminus R_4))$, where $\beta = (a_2, a_3)$ and $\eta = (p_7, p_8)$. One can verify that $W_4 \in \mathsf{Sol}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ and $W_3 \prec_{\textsc{Mer}} W_4$ (while $W_4 \prec_{\textsc{Del}} W_3$ and W_3 and W_4 are incomparable w.r.t. the $\preceq_{\textsc{Par}}$ preorder).

Overall, we obtain the following: $\mathsf{Sol}_{\textsc{Par}}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}}) = \{W_1, W_2, W_3, W_4\}$, $\mathsf{Sol}_{\textsc{Del}}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}}) = \{W_1, W_2, W_3\}$, and $\mathsf{Sol}_{\textsc{Mer}}(D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}}) = \{W_1, W_2, W_4\}$.

We conclude this section by situating REPLACE w.r.t. existing frameworks. First, observe that for any database-specification pair (D, Σ), we have $(\emptyset, E) \in \mathsf{Sol}_{\textsc{Del}}(D, \Sigma)$ iff $(\emptyset, E) \in \mathsf{Sol}_{\textsc{Par}}(D, \Sigma)$ iff E is a *maximal ER solution* in the sense of [Bienvenu *et al.*, 2022, Definition 3]. Thus, the maximal solutions considered in LACE can be seen as special case of $\preceq_{\textsc{Del}}$- and $\preceq_{\textsc{Par}}$-optimal solutions. It is not hard to see that an analogous property does not hold for $\preceq_{\textsc{Mer}}$ preorder.

Next we relate REPLACE solutions with the subset repairs employed in consistent query answering. Consider any database-specification pair (D, Σ) such that $\Sigma = \langle \emptyset, \Delta \rangle$. Then, $\mathsf{Sol}_{\textsc{Mer}}(D, \Sigma)$, $\mathsf{Sol}_{\textsc{Del}}(D, \Sigma)$, and $\mathsf{Sol}_{\textsc{Par}}(D, \Sigma)$ all coincide and contain only solutions of the form (R, trivE), where $\mathsf{trivE} = \{(c, c) \mid c \in \mathsf{dom}(D \setminus R)\}$. It is readily verified that $(R, \mathsf{trivE}) \in \mathsf{Sol}_{\textsc{Mer}}(D, \Sigma) = \mathsf{Sol}_{\textsc{Del}}(D, \Sigma) = \mathsf{Sol}_{\textsc{Par}}(D, \Sigma)$ iff $D \setminus R$ is a repair in the sense of [Chomicki and Marcinkowski, 2005, Definition 2.2].

5 Reasoning about Solutions

In this section, we analyze the computational complexity of the central decision problems associated with the REPLACE framework, namely, checking whether a given set of merges and deletions is an (optimal) solution, and whether a candidate tuple is a certain or possible answer w.r.t. the space of optimal solutions. As is common when considering data-centric tasks, we employ the *data complexity* measure [Vardi, 1982], i.e. complexity is measured w.r.t. the size of the database D (and also the pair $W = (R, E)$ when it is part of the input).

Our results, summarized in Table 1, consider the three notions of optimality, as well as the impact of adopting a syntactically restricted form of specification (defined further).

5.1 Solution Recognition

We first consider the solution recognition problem (REC): given Σ, D, and W, decide whether $W \in \mathsf{Sol}(D, \Sigma)$. Tractability easily follows from the P-completeness of the analogous problem for ERSol [Bienvenu *et al.*, 2022]:

Theorem 1. REC *is* P-*complete.*

Next we determine the complexity of the problem X-OPTREC of deciding whether $W \in \mathsf{Sol}_X(D, \Sigma)$, where $X \in \{\textsc{Mer}, \textsc{Del}, \textsc{Par}\}$ is the chosen optimality notion.

Theorem 2. *X-OPTREC is coNP-complete for any $X \in \{\textsc{Mer}, \textsc{Del}, \textsc{Par}\}$.*

The upper bounds employ a guess-and-check approach, exploiting Theorem 1. We transferred an existing coNP lower bound for maximal ER solutions to X-OPTREC when $X \in \{\textsc{Del}, \textsc{Par}\}$, while MER-OPTREC required a new proof.

5.2 Query Answering

In an ideal world, we would determine which solution corresponds to the true data, and query the resulting clean instance. When this is infeasible, due to lack of time or knowledge, a reasonable approach is to query the space of optimal solutions to identify those tuples that are answers w.r.t. every solution (in line with CQA semantics and the *skeptical* mode of inference employed in non-monotonic reasoning) or at least one solution (a form of *credulous / brave* reasoning).

This leads us to define the following notions of certain and possible answers. Note that given a solution $W = (R, E)$ to (D, Σ), we shall use $q(D, W)$ to refer to $q(D \setminus R, E)$.

Definition 5. *Given a DQ specification Σ, database D, and query q, all over schema $\mathcal{S}$, and $X \in \{\textsc{Mer}, \textsc{Del}, \textsc{Par}\}$, we say that a tuple $\mathbf{c}$ of constants is an X-*certain *(resp. X-*possible*) answer to q on D w.r.t. Σ if $\mathbf{c} \in q(D, W)$ for every (resp. some) $W \in \mathsf{Sol}_X(D, \Sigma)$. We use X-$\mathsf{certAns}(q, D, \Sigma)$ and X-$\mathsf{possAns}(q, D, \Sigma)$ to denote, respectively, the set of X-certain answers and X-possible answers to q on D w.r.t. Σ.*

Example 6. *First consider the query $q^1_{\mathsf{ex}}(x, y, z) = \exists t, v, e, m. Paper(x, t, y, 2023, v, e) \wedge Author(y, m, z)$, which returns the id of papers written in 2023 along with the institution and the id of its first author. For the tuple $\mathbf{t} = (p_6, a_3, Tokyo)$, we have the following:*

- $\mathbf{t} \in \textsc{Mer}\text{-}\mathsf{certAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$*;*
- $\mathbf{t} \notin \textsc{Del}\text{-}\mathsf{certAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$*, as $\mathbf{t} \notin q^1_{\mathsf{ex}}(D_{\mathsf{ex}}, W_3)$, hence also $\mathbf{t} \notin \textsc{Par}\text{-}\mathsf{certAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$;*
- $\mathbf{t} \in X\text{-}\mathsf{possAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *($X \in \{\textsc{Mer}, \textsc{Del}, \textsc{Par}\}$).*

Next let $q^2_{\mathsf{ex}}(x, y) = \exists t, f, v, m, i. Paper(x, t, f, 2012, v, y) \wedge Author(y, m, i)$ be the query that returns the ids of papers written in 2012 and the venue chair. Observe that X-$\mathsf{certAns}(q^2_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}}) = \emptyset$ for $X \in \{\textsc{Mer}, \textsc{Par}\}$, while $\textsc{Del}$-$\mathsf{certAns}(q^2_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}}) = \{(p_5, a_3)\}$. Notice moreover that X-$\mathsf{possAns}(q^2_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}}) = \{(p_5, a_2), (p_5, a_3)\}$ for each $X \in \{\textsc{Mer}, \textsc{Del}, \textsc{Par}\}$.

Specifications	X	X-OptRec	X-CertAns	X-PossAns	X-MIcertAns	X-MIpossAns
General	Mer/Del	coNP-c	Π_2^p-c	Σ_2^p-c	DP$_2$-c	DP$_2$-c
	Par	coNP-c	Π_2^p-c	NP-c	DP$_2$-c	DP-c
Restricted	Mer	coNP-c	Π_2^p-c	Σ_2^p-c	DP$_2$-c	DP$_2$-c
	Del/Par	P-c	coNP-c	NP-c	DP-c	DP-c

Table 1: Data complexity of the decision problems, parameterized by $X \in \{\text{Mer}, \text{Del}, \text{Par}\}$. We use '-c' as an abbreviation for '-complete'.

Our next theorem provides the complexity of the decision problems X-CertAns and X-PossAns of checking whether a given tuple of constants belongs to the set of X-certain answers and X-possible answers, respectively. We remind the reader that whenever we speak of queries we refer to CQs.

Theorem 3. *X-CertAns is Π_2^p-complete for any $X \in \{\text{Mer}, \text{Del}, \text{Par}\}$, Par-PossAns is NP-complete, and X-PossAns is Σ_2^p-complete for $X \in \{\text{Mer}, \text{Del}\}$.*

The Π_2^p and Σ_2^p membership proofs involve guessing a potential solution W that contains / omit the query tuple and calling an NP oracle to check that W is indeed an optimal solution. The NP upper bound for Par-PossAns relies upon showing that it is sufficient to check that W is a solution, rather than a Par-optimal solution. This is because $c \in q(D, W)$ implies $c \in q(D, W')$ for any W' such that $W \prec_{\text{Par}} W'$ (no such property holds for $\prec_{\text{Mer}}$ and $\prec_{\text{Del}}$). While some lower bounds were adapted from analogous results for Lace, others require new ingredients.

5.3 Restricted Specifications

The preceding results show that it is computationally challenging to reason about optimal solutions. Faced with a similar situation, Bienvenu *et al.* (2022) explored *restricted DQ specifications*, in which inequality atoms are disallowed in the denial constraints. While such specifications cannot capture keys and functional dependencies, they do allow for other meaningful forms of constraints, e.g. class and property disjointness statements commonly used for Semantic Web data.

Do restricted DQ specifications yield better complexity in our setting? For Rec, Mer-OptRec, Mer-CertAns, Mer-PossAns, and Par-PossAns, the answer is no, as the lower bound proofs employ restricted DQ specifications. However, for the remaining decision problems, we do find a drop in complexity (under the usual complexity assumptions).

Theorem 4. *For restricted DQ specifications, we have that:*

- *Del-OptRec and Par-OptRec are P-complete;*
- *X-CertAns is coNP-complete for $X \in \{\text{Del}, \text{Par}\}$ and Del-PossAns is NP-complete;*

Intuitively, this lower complexity is due to constraint violations being preserved under improvements, i.e. if δ is a denial constraint without $\neq$-atoms and both $W = (R, E)$ and $W' = (R', E')$ belong to $\mathsf{Sol}(D, E)$, then $(D \setminus R)_E \not\models \delta$ implies $(D \setminus R')_{E'} \not\models \delta$ whenever $E \subseteq E'$ and $R' \subseteq R$.

5.4 Comparison with Lace and CQA

Comparing with Lace, we note that in almost all cases, the addition of delete operations does not affect the complexity of recognizing (maximal / optimal) solutions or certain and possible answers. The main exception is if we consider $\prec_{\text{Mer}}$-optimal solutions coupled with restricted specifications, where all problems are one level higher in the polynomial hierarchy than the corresponding problems in Lace.

Adding merges to CQA brings a notable increase in complexity. Indeed, the certain query answering and optimal solution recognition tasks are one level higher than the corresponding CQA and repair checking tasks, if one considers general specifications or restricted specifications with the $\prec_{\text{Mer}}$ preorder. An even larger complexity jump is observed for possible query answering, as the analogous task w.r.t. repairs is easily seen to have polynomial data complexity.

6 Most Informative Answers

While our notions of certain and possible answers (and the corresponding notions in [Bienvenu *et al.*, 2022]) provide a natural way of querying the space of optimal solutions, they present one major drawback from an end user's perspective: the query results may contain multiple distinct tuples that are *equivalent w.r.t. the considered solutions*, as illustrated next.

Example 7. *Consider a scenario in which we have the database-specification pair (D, Σ), the database D contains facts $P(c_1, c_2)$ and $P(c_3, c_4)$, and $W = (\emptyset, E)$ with $E = \mathsf{EqRel}(\{(c_1, c_3), (c_2, c_4)\}, D)$ is the only $\preceq_X$-optimal solution for (D, Σ), for every $X \in \{\text{Mer}, \text{Del}, \text{Par}\}$. Then, for the query $q(x_1, x_2) = P(x_1, x_2)$ and for any $X \in \{\text{Mer}, \text{Del}, \text{Par}\}$, we have four X-certain/possible answers to q on D w.r.t. Σ, namely: (c_1, c_2), (c_1, c_4), (c_3, c_2), and (c_3, c_4). These tuples could be more concisely presented as a a single tuple of* sets of constants $(\{c_1, c_3\}, \{c_2, c_4\})$.

To address this issue and present query results with as much information (and as little repetition) as possible, we introduce the new notions of *most informative (certain / possible) answers*. The main idea, evoked in the example, that answers to queries now consist of tuples of *sets* of constants, each set comprising constants in the same equivalence relation w.r.t. the solution(s) under consideration.

Definition 6. *Given a solution $W = (R, E)$ for (D, Σ), an n-ary query q, and an n-tuple $\mathbf{C} = (C_1, \ldots, C_n)$ of sets of constants from D, we call $\mathbf{C}$ a* set-answer *to q on D* w.r.t. *W if the following holds:* (i) *C_i contains constants in the same equivalence class in E, for $1 \le i \le n$, and* (ii) *there exists a tuple of constants $\mathbf{c} = (c_1, \ldots, c_n) \in q(D, W)$ such that $c_i \in C_i$ for every $1 \le i \le n$. We denote by $\overline{q}(D, W)$ the set of set-answers to q on D w.r.t. W.*

Definition 7. *For $X \in \{\text{Mer}, \text{Del}, \text{Par}\}$, we say that a tuple $\mathbf{C}$ of sets of constants is a X-*certain set-answer *(resp. X-*

possible set-answer*)* to q on D w.r.t. Σ *if* $\mathbf{C} \in \overline{q}(D, W)$ *for every (resp. some)* $W \in \mathsf{Sol}_X(D, W)$*. We use* X-$\mathsf{SetCert}(q, D, \Sigma)$ *(resp.* X-$\mathsf{SetPoss}(q, D, \Sigma)$*) for the set of* X*-certain (resp.* X*-possible) set-answers to* q *on* D *w.r.t.* Σ*.*

Example 8. *Recall the queries* q^1_{ex} *and* q^2_{ex} *from Example 6. The tuple* $\mathbf{T} = (\{p_6\}, \{a_2, a_3\}, \{\mathit{Tokyo}\}) \in$ MER-$\mathsf{SetCert}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *while* $\mathbf{T} \notin X$-$\mathsf{SetCert}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *for both* $X =$ DEL *and* $X =$ PAR*.*

As another example, we have that $(\{p_5\}, \{a_2, a_3\}) \in X$-$\mathsf{SetPoss}(q^2_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *for each* $X \in \{$MER, DEL, PAR$\}$*.*

Among the X-certain and X-possible set-answers, we are interested in presenting the *most informative* ones. More formally, for two n-tuples $\mathbf{C} = \langle C_1, \dots, C_n \rangle$ and $\mathbf{C}' = \langle C'_1, \dots, C'_n \rangle$ of sets of constants, we say that $\mathbf{C}'$ is *strictly more informative* than $\mathbf{C}$ if (*i*) $C_i \subseteq C'_i$ for every $1 \leq i \leq n$, and (*ii*) $C_i \subset C'_i$ for some $1 \leq i \leq n$. Given a set S of n-tuples of sets of constants, we say that $\mathbf{C} \in S$ is *most informative in* S if there is no $\mathbf{C}' \in S$ that is strictly more informative than $\mathbf{C}$. With these notions in hand, we can now formally define most informative certain and possible answers.

Definition 8. *Given a DQ specification* Σ*, database* D*, and query* q*, all over schema* $\mathcal{S}$*, and* $X \in \{$MER, DEL, PAR$\}$*, we say that a tuple* $\mathbf{C}$ *of sets of constants from* D *is a* most informative X-certain answer *(resp.* most informative X-possible answer*) to* q *on* D *w.r.t.* Σ *if* $\mathbf{C}$ *is most informative in* X-$\mathsf{SetCert}(q, D, \Sigma)$ *(resp. in* X-$\mathsf{SetPoss}(q, D, \Sigma)$*). We denote by* X-$\mathsf{MIcertAns}(q, D, \Sigma)$ *(resp.* X-$\mathsf{MIpossAns}(q, D, \Sigma)$*) the set of most informative* X*-certain (resp.* X*-possible) answers to* q *on* D *w.r.t.* Σ*.*

Example 9. *Observe that* $\mathbf{T_1} = (\{p_6\}, \{a_2\}, \{\mathit{Tokyo}\}) \in X$-$\mathsf{MIcertAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *for* $X \in \{$DEL, PAR$\}$*, while* $\mathbf{T_1} \notin$ MER-$\mathsf{MIcertAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *because the tuple* $\mathbf{T}_0 = (\{p_6\}, \{a_2, a_3\}, \{\mathit{Tokyo}\})$ *in* MER-$\mathsf{SetCert}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *is strictly more informative than* $\mathbf{T_1}$*. In fact,* $\mathbf{T}_0 \in$ MER-$\mathsf{MIcertAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$*. Analogously, one can see that* $\mathbf{T_2} = (\{p_6, p_7\}, \{a_1, a_2\}, \{\mathit{Tokyo}\})$ *is such that* $\mathbf{T_2} \in X$-$\mathsf{MIpossAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *for both* $X =$ DEL *and* $X =$ PAR*, while* $\mathbf{T_2} \notin$ MER-$\mathsf{MIcertAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *because* $\mathbf{T_3} = (\{p_6, p_7, p_8\}, \{a_1, a_2, a_3\}, \{\mathit{Tokyo}\})$ *occurs in* MER-$\mathsf{SetPoss}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$ *and is strictly more informative than* $\mathbf{T_2}$*. In fact,* $\mathbf{T_3} \in$ MER-$\mathsf{MIpossAns}(q^1_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}})$*.*

For query q^2_{ex}*,* X-$\mathsf{MIcertAns}(q^2_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}}) = \emptyset$ *for* $X \in \{$MER, PAR$\}$ *while* DEL-$\mathsf{MIcertAns}(q^2_{\mathsf{ex}}, D_{\mathsf{ex}}, \Sigma_{\mathsf{ex}}) = \{(\{p_5\}, \{a_3\})\}$*. As for possible answers,* X-$\mathsf{MIpossAns} = \{(\{p_5\}, \{a_2, a_3\})\}$ *for each* $X \in \{$MER, DEL, PAR$\}$*.*

While X-$\mathsf{certAns}(q, D, \Sigma) \subseteq X$-$\mathsf{possAns}(q, D, \Sigma)$, the inclusion X-$\mathsf{MIcertAns}(q, D, \Sigma) \subseteq X$-$\mathsf{MIpossAns}(q, D, \Sigma)$ does not hold in general. However, we have the following related property: if $\mathbf{C} \in X$-$\mathsf{MIcertAns}(q, D, \Sigma)$, then either $\mathbf{C} \in X$-$\mathsf{MIpossAns}(q, D, \Sigma)$ or there exists $\mathbf{C}' \in X$-$\mathsf{MIpossAns}(q, D, \Sigma)$ that is strictly more informative than $\mathbf{C}$.

We now consider the decision problems X-MIPOSSANS and X-MIPOSSANS of checking whether a given tuple of sets of constants is a most informative X-certain (respectively, X-possible) answer. We find that adopting the most informative notions of answers leads to higher complexity compared to the (plain) notions of certain and possible answers.

Theorem 5. X-MICERTANS *is* DP_2*-complete*[1] *for any* $X \in \{$MER, DEL, PAR$\}$*,* X-MIPOSSANS *is* DP_2*-complete for* $X \in \{$MER, DEL$\}$*, and* PAR-MIPOSSANS *is DP-complete.*

The upper bounds rely on the fact that the set of yes-instances for X-MICERTANS is precisely the intersection of the yes-instances of X-SETCERT (decide whether $\mathbf{C} \in X$-$\mathsf{SetCert}(q, D, \Sigma)$) and X-NOBETTERCERT (decides whether there is no $\mathbf{C}' \in \mathsf{SetCert}(q, D, \Sigma)$ strictly more informative than $\mathbf{C}$). The latter can be solved by guessing a solution W_i for each possible 'minimal improvement' $\mathbf{C}_i$ of $\mathbf{C}$ and verifying that $\mathbf{C}_i \notin q(D, W_i)$. Similar considerations apply for X-MIPOSSANS. Lower bounds rely on new reductions.

We conclude this section by considering the impact of adopting restricted specifications. While MER-MICERTANS, PAR-MIPOSSANS, and MER-MICERTANS retain their original complexity, the other problems enjoy lower complexity.

Theorem 6. *For restricted DQ specifications, the decision problems* DEL-MICERTANS*,* PAR-MICERTANS*, and* DEL-MIPOSSANS *are DP-complete.*

7 Conclusion and Future Work

We presented REPLACE, a new holistic framework for (possibly recursive) collective entity resolution and repairing, which employs denial constraints coupled with (hard and soft) logical rules to infer merges. The semantics, based upon solutions that take the form of (coherent) sets of merges and deletions, generalizes both LACE and (subset) repairs. In the spirit of CQA, we studied how to query the space of (optimal) solutions. Our complexity analysis shows that while certain and possible answer recognition is harder than the analogous tasks for repairs, it is for the most part on par with existing results for LACE. We also explored an important question (not considered in LACE) of how to present the query results, which is non-trivial due to the merged constants, leading us to propose novel notions of most informative answers.

We view this work as a starting point, with many interesting questions left to explore. First, it could be natural to consider other reasoning tasks, such as identifying certain and possible merges and deletions, which could help guide users towards a unique solution. While some results can be transferred from query answering, other cases require further study. Next, we believe it would be interesting to explore extensions of REPLACE with quantitive information (weight or scores) associated to rules and facts, in particular, so that approximate weighted solutions could be generated. Finally, we would like to develop an efficient prototype based on logic-based technologies, such as answer set programming (ASP) [Gebser *et al.*, 2012]. To this end, we could use LACE's ASP encoding as a steppingstone, but new insights will be needed to handle most informative certain answers, whose DP_2 complexity goes beyond what is traditionally supported by ASP. It would also be interesting to use more informative similarity measures by adding ML predicates, in the style of [Deng *et al.*, 2022].

[1]We recall that the complexity classes DP (a.k.a. BH(2)) and DP_2 (a.k.a. $\mathrm{BH}_3(2)$) are the second level of the Boolean hierarchy of NP sets and of Σ^p_2 sets, respectively [Chang and Kadin, 1996].

Acknowledgements

This work has been supported by the ANR AI Chair INTENDED (ANR-19-CHIA-0014), by MUR under the PNRR project FAIR (PE0000013), and by the Royal Society (IES\R3\193236).

References

[Arasu *et al.*, 2009] Arvind Arasu, Christopher Ré, and Dan Suciu. Large-scale deduplication with constraints using dedupalog. In *Proceedings of the Twenty-Fifth International Conference on Data Engineering (ICDE 2009)*, pages 952–963, 2009.

[Arenas *et al.*, 1999] Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Consistent query answers in inconsistent databases. In *Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 1999)*, pages 68–79, 1999.

[Bertossi *et al.*, 2013] Leopoldo E. Bertossi, Solmaz Kolahi, and Laks V. S. Lakshmanan. Data cleaning and query answering with matching dependencies and matching functions. *Theory of Computing Systems*, 52(3):441–482, 2013.

[Bertossi, 2011] Leopoldo E. Bertossi. *Database Repairing and Consistent Query Answering*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.

[Bhattacharya and Getoor, 2007] Indrajit Bhattacharya and Lise Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data*, 1(1):5, 2007.

[Bienvenu *et al.*, 2022] Meghyn Bienvenu, Gianluca Cima, and Víctor Gutiérrez-Basulto. LACE: A logical approach to collective entity resolution. In *Proceedings of the Forty-First International Conference on Management of Data (PODS 2022)*, pages 379–391, 2022.

[Bienvenu, 2020] Meghyn Bienvenu. A short survey on inconsistency handling in ontology-mediated query answering. *Künstliche Intelligenz*, 34(4):443–451, 2020.

[Burdick *et al.*, 2016] Douglas Burdick, Ronald Fagin, Phokion G. Kolaitis, Lucian Popa, and Wang-Chiew Tan. A declarative framework for linking entities. *ACM Transactions on Database Systems*, 41(3):17:1–17:38, 2016.

[Chang and Kadin, 1996] Richard Chang and Jim Kadin. The boolean hierarchy and the polynomial hierarchy: A closer connection. *SIAM Journal on Computing*, 25(2):340–354, 1996.

[Chomicki and Marcinkowski, 2005] Jan Chomicki and Jerzy Marcinkowski. Minimal-change integrity maintenance using tuple deletions. *Information and Computation*, 197(1-2):90–121, 2005.

[Christophides *et al.*, 2021] Vassilis Christophides, Vasilis Efthymiou, Themis Palpanas, George Papadakis, and Kostas Stefanidis. An overview of end-to-end entity resolution for big data. *ACM Computing Surveys*, 53(6):127:1–127:42, 2021.

[Chu *et al.*, 2013] Xu Chu, Ihab F. Ilyas, and Paolo Papotti. Holistic data cleaning: Putting violations into context. In *Proceedings of the Twenty-Ninth IEEE International Conference on Data Engineering (ICDE 2013)*, pages 458–469, 2013.

[Deng *et al.*, 2022] Ting Deng, Wenfei Fan, Ping Lu, Xiaomeng Luo, Xiaoke Zhu, and Wanhe An. Deep and collective entity resolution in parallel. In *Proceedings of the Thirty-Eighth IEEE International Conference on Data Engineering (ICDE 2022)*, pages 2060–2072, 2022.

[Dixit and Kolaitis, 2022] Akhil A. Dixit and Phokion G. Kolaitis. Consistent answers of aggregation queries via SAT. In *Proceedings of the Thirty-Eighth IEEE International Conference on Data Engineering (ICDE 2022)*, pages 924–937, 2022.

[Fan and Geerts, 2012] Wenfei Fan and Floris Geerts. *Foundations of Data Quality Management*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2012.

[Fan *et al.*, 2009] Wenfei Fan, Xibei Jia, Jianzhong Li, and Shuai Ma. Reasoning about record matching rules. *Proceedings of the VLDB Endowment*, 2(1):407–418, 2009.

[Fan *et al.*, 2014] Wenfei Fan, Shuai Ma, Nan Tang, and Wenyuan Yu. Interaction between record matching and data repairing. *Journal of Data and Information Quality*, 4(4):16:1–16:38, 2014.

[Gebser *et al.*, 2012] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2012.

[Ilyas and Chu, 2019] Ihab F. Ilyas and Xu Chu. *Data Cleaning*. ACM, 2019.

[Lukasiewicz *et al.*, 2022] Thomas Lukasiewicz, Enrico Malizia, Maria Vanina Martinez, Cristian Molinaro, Andreas Pieris, and Gerardo I. Simari. Inconsistency-tolerant query answering for existential rules. *Artificial Intelligence*, 307:103685, 2022.

[Newcombe *et al.*, 1959] Howard B. Newcombe, James M. Kennedy, Samantha Axford, and Allen P. James. Automatic linkage of vital records. *Science*, 130(3381):954–959, 1959.

[Vardi, 1982] Moshe Y. Vardi. The complexity of relational query languages (extended abstract). In *Proceedings of the Fourteenth ACM Symposium on Theory of Computing (STOC 1982)*, pages 137–146, 1982.