Querying Inconsistent Prioritized Data with ORBITS: Algorithms, Implementation, and Experiments

Meghyn Bienvenu, Camille Bourgaux

Querying Inconsistent Prioritized Data with ORBITS: Algorithms, Implementation, and Experiments

Meghyn Bienvenu[1] , **Camille Bourgaux**[2]
[1] CNRS & University of Bordeaux, France
[2] DI ENS, ENS, CNRS, PSL University & Inria, Paris, France
meghyn.bienvenu@labri.fr, camille.bourgaux@ens.fr

Abstract

We investigate practical algorithms for inconsistency-tolerant query answering over prioritized knowledge bases, which consist of a logical theory, a set of facts, and a priority relation between conflicting facts. We consider three well-known semantics (AR, IAR and brave) based upon two notions of optimal repairs (Pareto and completion). Deciding whether a query answer holds under these semantics is (co)NP-complete in data complexity for a large class of logical theories, and SAT-based procedures have been devised for repair-based semantics when there is no priority relation, or the relation has a special structure. The present paper introduces the first SAT encodings for Pareto- and completion-optimal repairs w.r.t. general priority relations and proposes several ways of employing existing and new encodings to compute answers under (optimal) repair-based semantics, by exploiting different reasoning modes of SAT solvers. The comprehensive experimental evaluation of our implementation compares both (i) the impact of adopting semantics based on different kinds of repairs, and (ii) the relative performances of alternative procedures for the same semantics.

1 Introduction

The question of how to handle data that is inconsistent w.r.t. expressed constraints, be they given by database dependencies or ontologies, has great practical relevance. Data cleaning addresses this problem by modifying datasets so that they satisfy the constraints, often using heuristics to decide how to resolve contradictions, which may result in wrong facts being kept, or true facts removed (as discussed e.g. in (Fan 2015)). An alternative, more principled, approach is to adopt inconsistency-tolerant semantics in order to extract meaningful information from the contradictory data.

In the database setting, such an approach goes by the name of *consistent query answering* (CQA) and has been extensively studied since the seminal work of Arenas, Bertossi, and Chomicki (1999), see e.g. the recent survey by Wijsen (2019). A central notion is that of a *(subset) repair*, defined as a maximal subset of the dataset that satisfies the constraints. Intuitively, repairs represent all different ways of minimally modifying the data to satisfy the constraints. As we do not know which repair corresponds to the true part of the data, the CQA semantics stipulates that a tuple is a query answer if it is an answer w.r.t. every repair (in line with how skeptical inference is defined in many KR settings).

Inconsistency-tolerant semantics have also drawn considerable interest in the setting of ontology-mediated query answering (OMQA) (Poggi et al. 2008; Bienvenu and Ortiz 2015; Xiao et al. 2018), where the ontology not only specifies constraints on the data but also captures other forms of domain knowledge, which can be exploited at query time. In addition to the *AR semantics* (the OMQA analog of the CQA semantics), several other inconsistency-tolerant semantics have been proposed (see (Bienvenu and Bourgaux 2016; Bienvenu 2020) for surveys and references), among which: the *brave semantics* (Bienvenu and Rosati 2013), which only requires a tuple to be an answer w.r.t. some repair, provides a natural notion of possible answer, and the *IAR semantics* (Lembo et al. 2010), which answers queries over the intersection of the repairs, identifies the most reliable answers.

The basic notion of repair can be refined by exploiting preference information. A prominent approach, introduced by Staworko, Chomicki, and Marcinkowski (2012) in the database setting and recently explored in the OMQA setting (Bienvenu and Bourgaux 2020), assumes that preferences are given by a binary *priority relation* between conflicting facts. Three notions of 'best' repairs w.r.t. a priority relation were proposed, namely, Pareto-optimal, globally-optimal, and completion-optimal repairs, and can be used in place of subset repairs in any repair-based semantics.

The complexity of answering queries under (optimal) repair-based semantics has been extensively studied in the database and OMQA settings, refer to (Wijsen 2019; Bienvenu and Bourgaux 2016) for an overview and references. We can briefly summarize these (many!) complexity results as follows: query answering under the AR (or CQA) semantics is coNP-hard in data complexity even in the simplest of settings (e.g. key constraints, class disjointness), and adopting optimal repairs in place of subset repairs leads to (co)NP-hardness for the brave and IAR semantics as well. Membership in (co)NP holds for AR, brave, and IAR semantics w.r.t. subset, Pareto-optimal, and completion-optimal repairs in the most commonly considered settings i.e. for database constraints given by primary keys or more generally, functional dependencies (FDs), and for ontologies formulated in data-tractable description logics such as those of the DL-Lite family (Calvanese et al. 2007).

The preceding (co)NP complexity results naturally suggest the interest of employing SAT solvers. Two recent sys-

tems, CQAPri and CAvSAT, have begun to explore such an approach. CQAPri (Bienvenu, Bourgaux, and Goasdoué 2014; 2019) uses tractable approximations together with calls to SAT solvers to answer queries over inconsistent DL-Lite knowledge bases, under the AR, brave, and IAR semantics, w.r.t. subset repairs as well as optimal repairs for the restricted class of score-structured priority relations. CAvSAT (Dixit and Kolaitis 2019) targets relational databases equipped with denial constraints (which include FDs as a special case) and computes query answers under the AR semantics w.r.t. subset repairs. While geared to different forms of constraints, the two systems solve essentially the same problem, yet they employ SAT solvers in different ways. CQAPri makes a single SAT call for each candidate query answer, whereas CAvSAT treats all candidate answers at the same time via calls to a weighted MaxSAT solver.

This paper presents a comprehensive study of the use of SAT-based approaches for inconsistency-tolerant query answering, which abstracts from the particular setting and provides a solid foundation for the future development of such systems. Our contributions can be summarized as follows. In Section 3, we provide propositional encodings of the AR, brave, and IAR semantics, including the first encodings for Pareto- and completion-optimal repairs. Our encodings are generic and are built in a modular manner from a core set of basic formulas. Based upon these encodings, we develop in Section 4 several SAT-based algorithms, which utilize different functionalities of modern SAT solvers: weighted MaxSAT, MUS enumeration, iterative SAT calls with assumptions. In Section 5, we present our implemented system, ORBITS, which computes query answers under the chosen semantics using the selected encoding and algorithm. Section 6 presents the result of our extensive experimental evaluation, using existing OMQA and database benchmarks, aimed at comparing the different semantics, and understanding the relative performances of different encodings and/or algorithms for the same semantics. Proofs, pseudo-code for algorithms, and details on the experimental evaluation are provided in the appendix of (Bienvenu and Bourgaux 2022).

2 Querying Inconsistent Knowledge Bases

We introduce notation and terminology for talking about knowledge bases and then recall different inconsistency-tolerant semantics. We shall assume throughout the paper that readers are familiar with propositional and first-order logic. Key notions are illustrated in Example 1.

Knowledge bases By *knowledge base (KB)*, we mean a pair $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ consisting of a *dataset* $\mathcal{D}$ and a *logical theory* $\mathcal{T}$. The dataset $\mathcal{D}$ is a finite set of ground *facts*, i.e. atoms $P(c_1, \ldots, c_n)$ where P is an n-ary predicate and each c_i is a constant. The theory $\mathcal{T}$ is a finite set of first-order logic (FOL) sentences. An $\mathcal{L}$ *KB* is a KB whose theory is formulated in the $\mathcal{L}$ fragment of FOL. Typically, $\mathcal{T}$ will be either an *ontology* (with $\mathcal{L}$ a description logic or decidable class of existential rules) or a set of *database constraints*, for instance, letting $\mathcal{L}$ be the class of *denial constraints*, which take the form $\forall \vec{x} \neg(\alpha_1 \wedge \ldots \wedge \alpha_n)$, where each α_i is a relational or inequality atom whose variables are among $\vec{x}$. Denial constraints generalize the more well-known functional dependencies (FDs) and (primary) key constraints.

A KB $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ is *consistent*, and its dataset $\mathcal{D}$ is called $\mathcal{T}$*-consistent*, if $\mathcal{D} \cup \mathcal{T}$ has at least one model. Otherwise, $\mathcal{K}$ is *inconsistent*, denoted $\mathcal{K} \models \bot$. To identify the reasons for a KB $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ being inconsistent, we use the notion of a *conflict* of $\mathcal{K}$, defined as an inclusion-minimal subset $\mathcal{D}' \subseteq \mathcal{D}$ such that $(\mathcal{D}', \mathcal{T}) \models \bot$. We use $\mathit{Conf}(\mathcal{K})$, or $\mathit{Conf}(\mathcal{D}, \mathcal{T})$, to refer to the set of all conflicts of $\mathcal{K} = (\mathcal{D}, \mathcal{T})$. Note that $\mathit{Conf}(\mathcal{D}, \mathcal{T}) \subseteq \mathit{Conf}(\mathcal{D}', \mathcal{T})$ whenever $\mathcal{D} \subseteq \mathcal{D}'$. In particular, this means that adding more facts cannot render an inconsistent KB consistent.

We will be interested in answering queries over KBs. In this paper, when we speak of queries, we mean *conjunctive queries*, which take the form of conjunctions of relational atoms $P(t_1, \ldots, t_n)$ (with each t_i a constant or variable), where some variables may be existentially quantified. Given a query $q(\vec{x})$, with free variables $\vec{x} = (x_1, \ldots, x_k)$, and a tuple of constants $\vec{a} = (a_1, \ldots, a_k)$, we denote by $q(\vec{a})$ the first-order sentence obtained by replacing each variable in $\vec{x}$ by the corresponding constant in $\vec{a}$. A *(certain) answer* to $q(\vec{x})$ over $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ is a tuple of constants $\vec{a}$ from $\mathcal{D}$ such that $q(\vec{a})$ holds in every model of $\mathcal{K}$. We use $\mathcal{K} \models q(\vec{a})$ to indicate that $\vec{a}$ is a certain answer to $q(\vec{x})$ over $\mathcal{K}$.

Certain answers are preserved under the addition of facts: if $(\mathcal{D}, \mathcal{T}) \models q(\vec{a})$ and $\mathcal{D} \subseteq \mathcal{D}'$, then $(\mathcal{D}', \mathcal{T}) \models q(\vec{a})$. It thus makes sense to consider the minimal subsets of the data responsible for an answer. We call a $\mathcal{T}$-consistent subset $\mathcal{C} \subseteq \mathcal{D}$ a *cause* for $q(\vec{a})$ w.r.t. $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ if $(\mathcal{C}, \mathcal{T}) \models q(\vec{a})$ and $(\mathcal{C}', \mathcal{T}) \not\models q(\vec{a})$ for every $\mathcal{C}' \subsetneq \mathcal{C}$; the set of causes for $q(\vec{a})$ w.r.t. $\mathcal{K}$ is denoted by $\mathit{Causes}(q(\vec{a}), \mathcal{K})$.

Observe that if $\mathcal{K}$ is inconsistent, then $\mathcal{K} \models q(\vec{a})$ for *every* candidate answer $\vec{a}$, which is uninformative and motivates the need for alternative inconsistency-tolerant semantics.

Repairs In order to extract meaningful information from an inconsistent KB, it is useful to consider the parts of the data that are consistent with the theory. This can be formalized using the notion of repair:

Definition 1. A *(subset) repair* of a KB $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ is an inclusion-maximal subset $\mathcal{R} \subseteq \mathcal{D}$ such that $(\mathcal{R}, \mathcal{T}) \not\models \bot$. We use $\mathit{SRep}(\mathcal{K})$ to denote the set of repairs of KB $\mathcal{K}$.

Subset repairs treats all facts equally. However, preferences between conflicting facts should be taken into account when they are available. Following (Staworko, Chomicki, and Marcinkowski 2012; Bienvenu and Bourgaux 2020), we assume such preferences are given as a priority relation:

Definition 2. A *priority relation* $\succ$ for a KB $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ is an acyclic binary relation over the facts of $\mathcal{D}$ such that if $\alpha \succ \beta$, then there exists $\mathcal{C} \in \mathit{Conf}(\mathcal{K})$ such that $\{\alpha, \beta\} \subseteq \mathcal{C}$. We say that $\succ$ is *total* if for every pair $\alpha \neq \beta$ such that $\{\alpha, \beta\} \subseteq \mathcal{C}$ for some $\mathcal{C} \in \mathit{Conf}(\mathcal{K})$, either $\alpha \succ \beta$ or $\beta \succ \alpha$. A *completion* of $\succ$ is a total priority relation $\succ' \supseteq \succ$.

Definition 3. A *prioritized KB* $\mathcal{K}_\succ$ consists of a KB $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ and a priority relation $\succ$ for $\mathcal{K}$.

We recall two[1] natural ways of refining the notion of re-

[1] Staworko et al. (2012) defined a third notion, *globally-optimal repairs*, which we do not consider due to their higher complexity.

pair to exploit priority relations (Staworko, Chomicki, and Marcinkowski 2012; Bienvenu and Bourgaux 2020):

Definition 4. Consider a prioritized KB $\mathcal{K}_\succ$ with $\mathcal{K} = (\mathcal{D}, \mathcal{T})$, and let $\mathcal{R} \in SRep(\mathcal{K})$.

- A *Pareto improvement* of $\mathcal{R}$ is a $\mathcal{T}$-consistent $\mathcal{B} \subseteq \mathcal{D}$ such that there is $\beta \in \mathcal{B} \setminus \mathcal{R}$ with $\beta \succ \alpha$ for every $\alpha \in \mathcal{R} \setminus \mathcal{B}$.

The repair $\mathcal{R}$ is a:

- *Pareto-optimal repair* of $\mathcal{K}_\succ$ if there is no Pareto improvement of $\mathcal{R}$.
- *completion-optimal repair* of $\mathcal{K}_\succ$ if $\mathcal{R}$ is a Pareto-optimal repair of $\mathcal{K}_{\succ'}$, for some completion $\succ'$ of $\succ$.

We denote by $PRep(\mathcal{K}_\succ)$ and $CRep(\mathcal{K}_\succ)$ the sets of Pareto- and completion-optimal repairs.

It is known that $CRep(\mathcal{K}_\succ) \subseteq PRep(\mathcal{K}_\succ) \subseteq SRep(\mathcal{K})$, with each of the inclusions potentially strict. Interestingly, however, if $\succ$ is induced by assigning scores to facts, then Pareto- and completion-optimal repairs coincide:

Definition 5. A priority relation $\succ$ for $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ is *score-structured* if there exists a scoring function $s : \mathcal{D} \to \mathbb{N}$ such that for every pair of facts α and β that appear together in a conflict of $\mathcal{K}$, we have $\alpha \succ \beta$ iff $s(\alpha) > s(\beta)$.

Theorem 1. *(Livshits and Kimelfeld 2017; Bourgaux 2016) Let $\mathcal{K}_\succ$ be a prioritized KB such that $\succ$ is score-structured. Then $CRep(\mathcal{K}_\succ) = PRep(\mathcal{K}_\succ)$.*

Bourgaux (2016) further shows that for score-structured priorities, Pareto- and completion-optimal repairs also coincide with the $\subseteq_P$-repairs from (Bienvenu, Bourgaux, and Goasdoué 2014) based upon lexicographic set inclusion.

Repair-based semantics We next recall three prominent inconsistency-tolerant semantics (brave, AR, and IAR), which we parameterize by the considered type of repair:

Definition 6. Fix X $\in \{S, P, C\}$ and consider a prioritized KB $\mathcal{K}_\succ$ with $\mathcal{K} = (\mathcal{D}, \mathcal{T})$, query $q(\vec{x})$, and tuple of constants $\vec{a}$ from $\mathcal{D}$ with $|\vec{x}| = |\vec{a}|$. Then $\vec{a}$ is an answer to q over $\mathcal{K}_\succ$

- under *X-brave semantics*, denoted $\mathcal{K}_\succ \models^X_{\text{brave}} q(\vec{a})$, if $(\mathcal{R}, \mathcal{T}) \models q(\vec{a})$ for some $\mathcal{R} \in XRep(\mathcal{K}_\succ)$
- under *X-AR semantics*, denoted $\mathcal{K}_\succ \models^X_{\text{AR}} q(\vec{a})$, if $(\mathcal{R}, \mathcal{T}) \models q(\vec{a})$ for every $\mathcal{R} \in XRep(\mathcal{K}_\succ)$
- under *X-IAR semantics*, denoted $\mathcal{K}_\succ \models^X_{\text{IAR}} q(\vec{a})$, if $(\mathcal{B}, \mathcal{T}) \models q(\vec{a})$ where $\mathcal{B} = \bigcap_{\mathcal{R} \in XRep(\mathcal{K}_\succ)} \mathcal{R}$

The AR semantics is arguably the most natural way of defining *plausible* query answers, and it is the semantics used for consistent query answering in databases (Bertossi 2011). The brave semantics captures the notion of *possible* answers, while the IAR semantics identifies the answers that can be obtained using only the *most reliable* facts. Observe that $\mathcal{K}_\succ \models^X_{\text{IAR}} q \Rightarrow \mathcal{K}_\succ \models^X_{\text{AR}} q \Rightarrow \mathcal{K}_\succ \models^X_{\text{brave}} q$.

Example 1. Let $\mathcal{K} = (\mathcal{D}, \mathcal{T})$ where $\mathcal{D}$ contains four facts $\alpha = R(a, b)$, $\beta = R(a, c)$, $\gamma = S(d, c)$, and $\delta = S(d, b)$, and $\mathcal{T}$ contains FDs $\forall x, y, z \neg(R(x, y) \wedge R(x, z) \wedge y \neq z)$, $\forall x, y, z \neg(S(x, y) \wedge S(x, z) \wedge y \neq z)$ and the denial constraint $\forall x, y, z \neg(R(y, x) \wedge S(z, x))$.

The conflicts of $\mathcal{K}$ are $\{\alpha, \beta\}$, $\{\gamma, \delta\}$, $\{\alpha, \delta\}$ and $\{\beta, \gamma\}$, hence $SRep(\mathcal{K}) = \{\{\alpha, \gamma\}, \{\beta, \delta\}\}$. If we define $\succ$ by $\alpha \succ \beta$ and $\gamma \succ \delta$, we obtain $PRep(\mathcal{K}_\succ) = SRep(\mathcal{K})$ but $CRep(\mathcal{K}_\succ) = \{\{\alpha, \gamma\}\}$. Indeed, any completion $\succ'$ of $\succ$ is such that $\alpha \succ' \delta$ or $\gamma \succ' \beta$ by acyclicity.

If $q(x) = \exists y R(x, y)$, $Causes(q(a), \mathcal{K}) = \{\{\alpha\}, \{\beta\}\}$. Hence, $\mathcal{K}_\succ \models^C_{\text{IAR}} q(a)$, $\mathcal{K}_\succ \models^P_{\text{AR}} q(a)$ but $\mathcal{K}_\succ \not\models^P_{\text{IAR}} q(a)$.

We briefly recall what is known about the complexity of query answering under (optimal) repair-based semantics. Note that *when we speak of complexity, we mean data complexity*, measured solely in terms of the size of the dataset.

Theorems 2 and 3 summarize upper and lower bounds from the database and ontology settings. All of them are known (Staworko, Chomicki, and Marcinkowski 2012; Bienvenu and Bourgaux 2020; Rosati 2011), except the lower bounds for X-brave and X-IAR semantics in the case of FDs, proven in (Bienvenu and Bourgaux 2022). We say that a logic $\mathcal{L}$ *enjoys* PTIME *consistency checking (resp. query entailment)* if the problem of deciding whether $\mathcal{K} \models \bot$ (resp. $\mathcal{K} \models q(\vec{a})$) for an input $\mathcal{L}$ KB is in PTIME.

Theorem 2. *Let $\mathcal{L}$ be any FOL fragment that enjoys* PTIME *consistency checking and query entailment, and let X $\in \{S, P, C\}$. Then query entailment for $\mathcal{L}$ KBs is*

- *in* NP *under X-brave semantics, and*
- *in co*NP *under X-AR and X-IAR semantics.*

Theorem 3. *Query entailment for $\mathcal{L}$ KBs is*

- NP*-hard under X-brave semantics (X $\in \{P, C\}$)*
- *co*NP*-hard under X-AR semantics (X $\in \{S, P, C\}$)*
- *co*NP*-hard under X-IAR semantics (X $\in \{P, C\}$)*

for any $\mathcal{L}$ that extends DL-Lite$_{\mathsf{core}}$, $\mathcal{EL}_\bot$, or FDs.

Remark 1. Query entailment under S-brave and S-IAR is tractable both for DL-Lite ontologies and denial constraints (Bienvenu and Rosati 2013).

3 SAT Encodings

The (co)NP complexity results from the previous section suggest a SAT-based approach to query entailment under (optimal) repair-based semantics. While our work is not the first to explore SAT encodings for inconsistency-tolerant semantics, our contribution is a uniform approach that covers a wide range of semantics and settings, and provides the first encodings for Pareto- and completion-optimal repairs.

3.1 Overview

Our propositional encodings are built from:

- the set $Conf(\mathcal{K})$ of conflicts of the considered KB $\mathcal{K}$,
- a set $PotAns$ of *potential answers*, and for each $\vec{a} \in PotAns$, the (non-empty[2]) set $Causes(q(\vec{a}), \mathcal{K})$,
- the priority relation $\succ$ of $\mathcal{K}$,

and are of polynomial size w.r.t. to these inputs. Note that for DL-Lite ontologies and denial constraints, the sets of conflicts, candidate answers, and their causes, can be computed in PTIME via database query evaluation, so our encodings will yield procedures of the expected co(NP) complexity.

[2]If $Causes(q(\vec{a}), \mathcal{K}) = \emptyset$, $q(\vec{a})$ holds for none of our semantics.

To simplify the treatment, *we shall assume that every conflict contains at most two facts*, a property that is satisfied by the most common DL-Lite dialects and for FDs. (The extension to non-binary conflicts is discussed later in the section.) By restricting our attention to binary conflicts, we can *use a convenient notation $\alpha\bot\beta$ in place of $\{\alpha,\beta\} \in Conf(\mathcal{K})$*, and define a graph representation of conflicts and priorities. The *directed conflict graph* $\mathcal{G}_{\mathcal{K}_\succ}$ has facts from $Conf(\mathcal{K})$ as nodes and an edge from α to β iff $\alpha\bot\beta$ and $\alpha\not\succ\beta$.

Our encodings will use variables of the form x_α to indicate whether fact α appears in a (partial) repair. Including one such variable for each fact in $\mathcal{D}$ would yield prohibitively large encodings, which is why we use $\mathcal{G}_{\mathcal{K}_\succ}$ to focus on relevant facts: given any $F \subseteq \mathcal{D}$, we let $\mathsf{R}(F)$ be the set of all *facts that are reachable in* $\mathcal{G}_{\mathcal{K}_\succ}$ from some $\alpha \in F$.

Before proceeding to the details, let us give a high-level overview of our encodings. All of the encodings try to construct a set of facts that is consistent with the theory (i.e. does not contain any conflicts) and that can be extended to an optimal repair. For the X-AR semantics, we are trying to build an optimal repair that does *not* entail the considered query answer(s), which can be done by including facts that contradict each of the answer(s) causes. For the X-brave semantics, we must ensure that the repair entails the considered query answer(s), achieved by requiring the presence of some cause in the chosen subset. Finally, for the X-IAR semantics, we ensure the existence of optimal repairs that omit a given cause or fact appearing in a cause by including facts that contradict the considered cause or fact.

In the next section, we will present SAT-based algorithms that consider each potential answer in turn, as well as algorithms that treat all potential answers conjointly. For that reason, in what follows, we will present encodings both for a single answer and for several answers at a time.

3.2 Basic Building Blocks

Our various encodings rely upon some common ingredients, which are presented next.

Absence or presence of causes To contradict a specific cause $\mathcal{C}$, we use $\varphi_{\neg\mathcal{C}}$, with two alternative definitions inspired respectively by the CQAPri and CAvSAT encodings:

$$\varphi_{\neg\mathcal{C}} = \bigvee_{\alpha\in\mathcal{C}} \bigvee_{\alpha\bot\beta,\alpha\not\succ\beta} x_\beta \qquad (\mathsf{neg}_1)$$

$$\varphi_{\neg\mathcal{C}} = \bigvee_{\alpha\in\mathcal{C}} \neg x_\alpha \wedge \bigwedge_{\alpha\in\mathcal{C}} (x_\alpha \vee \bigvee_{\alpha\bot\beta,\alpha\not\succ\beta} x_\beta) \qquad (\mathsf{neg}_2)$$

For encodings with multiple answers, we use a variant $\varphi'_{\neg\mathcal{C}}(y)$ obtained by adding $\neg y$ as a disjunct of the (first) clause of $\varphi_{\neg\mathcal{C}}$, so it is only 'active' when the given variable y is true. To block all causes of the BCQ $q(\vec{a})$, we can use

$$\varphi_{\neg q(\vec{a})} = \bigwedge_{\mathcal{C}\in Causes(q(\vec{a}),\mathcal{K})} \varphi_{\neg\mathcal{C}}$$

or, in the case of encodings for several answers, a variant $\varphi'_{\neg q(\vec{a})}$ obtained by replacing $\varphi_{\neg\mathcal{C}}$ by $\varphi'_{\neg\mathcal{C}}(x_{\vec{a}})$.

If instead we want to force that cause $\mathcal{C}$ holds, we use:

$$\varphi_{\mathcal{C}} = \bigwedge_{\alpha\in\mathcal{C}} x_\alpha$$

and to ensure *some* cause for BCQ $q(\vec{a})$ is present, we use:

$$\varphi_{q(\vec{a})} = (\bigvee_{\mathcal{C}\in Causes(q(\vec{a}),\mathcal{K})} x_{\mathcal{C}}) \wedge \bigwedge_{\mathcal{C}\in Causes(q(\vec{a}),\mathcal{K})} \bigwedge_{\alpha\in\mathcal{C}} \neg x_{\mathcal{C}} \vee x_\alpha$$

or, for multi-answer encodings, a variant $\varphi'_{q(\vec{a})}$ obtained from $\varphi_{q(\vec{a})}$ by adding $\neg x_{\vec{a}}$ as a disjunct of the first clause.

Consistency We use $\varphi_{\mathsf{cons}}(F)$ to ensure that the valuation of $\{x_\alpha \mid \alpha \in F\}$ corresponds to a $\mathcal{T}$-consistent set of facts:

$$\varphi_{\mathsf{cons}}(F) = \bigwedge_{\alpha,\beta\in F,\alpha\bot\beta} \Big(\neg x_\alpha \vee \neg x_\beta\Big).$$

Extension to optimal repair The most intricate part of the encoding is ensuring that the selected set of facts can be extended to a repair of the desired type. To this end, we introduce formulas of the form $\varphi_{\mathsf{X\text{-}max}}(F)$, where F provides the facts that may appear, and X is the type of repair.

- Subset repairs: we can simply set $\varphi_{\mathsf{S\text{-}max}}(F) = \top$, as every consistent set of facts extends to some S-repair.
- Pareto-optimal repairs: we prove that the following encoding of maximality for $\subseteq_P$-repairs (w.r.t. score-structured $\succ$) from Bienvenu et al. (2014) in fact also works for Pareto-optimal repairs and arbitrary $\succ$:

  $$\varphi_{\mathsf{P_1\text{-}max}}(F) = \bigwedge_{\alpha\in\mathsf{R}(F)} (x_\alpha \vee \bigvee_{\alpha\bot\beta,\alpha\not\succ\beta} x_\beta)$$

  Essentially, it states that a relevant fact can only be omitted if we include a non-dominated contradicting fact. We also propose an alternative encoding with fewer variables:

  $$\varphi_{\mathsf{P_2\text{-}max}}(F) = \bigwedge_{\alpha\in\mathsf{R}^-(F)} \bigwedge_{\beta\succ\alpha} (\neg x_\alpha \vee \bigvee_{\beta\bot\gamma,\beta\not\succ\gamma} x_\gamma)$$

  where $\mathsf{R}^-(F) = \bigcup_{i=1}^{\infty} \mathsf{R}_i$ with $\mathsf{R}_0 = F$ and
  $\mathsf{R}_{i+1} = \mathsf{R}_i \cup \{\gamma \mid \exists\alpha \in \mathsf{R}_i, \beta \succ \alpha, \beta\bot\gamma, \beta \not\succ \gamma\}$.

  Intuitively, to include $\alpha \in F$, we must contradict every more preferred fact that conflicts with α, then do the same for selected contradicting facts.
- Completion-optimal repairs (w.r.t. arbitrary $\succ$): we use

  $$\varphi_{\mathsf{C\text{-}max}}(F) = \varphi_{\mathsf{pref}} \wedge \varphi_{\mathsf{compl}} \wedge \varphi_{\mathsf{acyc}}$$

  where the subformulas φ_{pref}, φ_{compl}, and φ_{acyc} are defined in Figure 1. The formula φ_{pref} states that if x_α is omitted, then we must include a contradicting fact β that is preferred to α according to $\succ'$. We use φ_{compl} to ensure that $\succ'$ compares all contradicting facts and extends $\succ$, while the acyclicity of $\succ'$ is ensured by φ_{acyc}, which uses variables $t_{\alpha,\beta}$ to compute the transitive closure of $\succ'$.

Non-binary conflicts We briefly discuss how to modify the preceding formulas to handle non-binary conflicts. The most essential difference is that instead of choosing a single contradicting fact, we may need to choose a conjunction of facts, and use additional variables to refer to them. We must also redefine $\mathcal{G}_{\mathcal{K}_\succ}$ as a directed hypergraph, and use hypergraph reachability to define $\mathsf{R}(F)$. Details of the required modifications are provided in (Bienvenu and Bourgaux 2022).

$$\varphi_{\mathsf{pref}} = \bigwedge_{\alpha \in \mathsf{R}(F)} \Big(x_\alpha \vee \bigvee_{\beta \in \mathsf{R}(F), \alpha \perp \beta} x_{\beta \to \alpha} \Big) \wedge \bigwedge_{\alpha,\beta \in \mathsf{R}(F), \alpha \perp \beta} \Big((\neg x_{\beta \to \alpha} \vee x_\beta) \wedge (\neg x_{\beta \to \alpha} \vee x_{\beta \succ' \alpha}) \Big)$$

$$\varphi_{\mathsf{compl}} = \bigwedge_{\alpha,\beta \in \mathsf{R}(F), \alpha \perp \beta} \Big(x_{\alpha \succ' \beta} \vee x_{\beta \succ' \alpha} \Big) \wedge \bigwedge_{\alpha,\beta \in \mathsf{R}(F), \alpha \perp \beta} \Big(\neg x_{\alpha \succ' \beta} \vee \neg x_{\beta \succ' \alpha} \Big) \wedge \bigwedge_{\alpha,\beta \in \mathsf{R}(F), \alpha \succ \beta} x_{\alpha \succ' \beta}$$

$$\varphi_{\mathsf{acyc}} = \bigwedge_{\alpha,\beta \in \mathsf{R}(F), \alpha \perp \beta} \Big(\neg x_{\alpha \succ' \beta} \vee t_{\alpha,\beta} \Big) \wedge \bigwedge_{\alpha,\beta \in \mathsf{R}(F), \alpha \perp \beta} \Big(\neg x_{\alpha \succ' \beta} \vee \neg t_{\beta,\alpha} \Big) \wedge \bigwedge_{\alpha,\beta,\gamma \in \mathsf{R}(F), \beta \perp \gamma} \Big(\neg t_{\alpha,\beta} \vee \neg x_{\beta \succ' \gamma} \vee t_{\alpha,\gamma} \Big)$$

Figure 1: Subformulas of $\varphi_{\mathsf{C\text{-}max}}(F)$, which is used to ensure it is possible to extend the selected subset of F to a completion-optimal repair.

3.3 Propositional Encodings

We now present our encodings, built from the preceding components following the intuitions given in Section 3.1.

Note that the following results hold no matter which variant of $\varphi_{\neg \mathcal{C}}$ we use and with $\varphi_{\mathsf{P\text{-}max}}$ instantiated as either $\varphi_{\mathsf{P_1\text{-}max}}$ or $\varphi_{\mathsf{P_2\text{-}max}}$. The notation $\mathsf{facts}(\varphi)$ will be used for the set of facts α such that the variable x_α occurs in φ.

X-AR semantics Formula $\Phi_{X\text{-}AR}(q(\vec{a}))$ in Figure 2 is used to test whether a particular tuple $\vec{a}$ holds. Roughly speaking, it selects a way to contradict every clause, checking that the resulting set of facts is contained in a X-optimal repair. The second formula $\Psi_{X\text{-}AR}(PotAns)$ simultaneously handles all tuples in $PotAns$. Clauses of the form $x_{\vec{a}}$ can be added to activate $\varphi'_{\neg q(\vec{a})}$.

The correctness of our encodings is given in the next result, which applies to all X $\in \{S, P, C\}$ and $\vec{a} \in PotAns$:

Theorem 4. *The following are equivalent : (i) $\mathcal{K}_\succ \models^X_{AR} q(\vec{a})$, (ii) $\Phi_{X\text{-}AR}(q(\vec{a}))$ is unsatisfiable, and (iii) $x_{\vec{a}}$ is false in every satisfying assignment of $\Psi_{X\text{-}AR}(PotAns)$.*

X-brave semantics Figure 2 presents our encodings for the X-brave semantics. Formula $\Phi_{X\text{-}brave}(q(\vec{a}))$ checks a particular tuple $\vec{a}$ and is essentially the same as $\Phi_{X\text{-}AR}(q(\vec{a}))$, but with $\varphi_{q(\vec{a})}$ in place of $\varphi_{\neg q(\vec{a})}$. A variant $\Phi_{X\text{-}brave}(\mathcal{C})$ can be used to check whether a particular cause $\mathcal{C}$ holds in some X-optimal repair. For a multi-answer encodings, we use the formula $\Psi_{X\text{-}brave}(PotAns)$, again adding clauses of the form $x_{\vec{a}}$ to activate $\varphi'_{q(\vec{a})}$. We obtain an analogous correctness result:

Theorem 5. *The following are equivalent: (i) $\mathcal{K}_\succ \models^X_{brave} q(\vec{a})$, (ii) $\Phi_{X\text{-}brave}(q(\vec{a}))$ is satisfiable, (iii) $\Phi_{X\text{-}brave}(\mathcal{C})$ is satisfiable for some $\mathcal{C} \in Causes(q(\vec{a}), \mathcal{K})$, and (iv) $x_{\vec{a}}$ is true in some satisfying assignment of $\Psi_{X\text{-}brave}(PotAns)$.*

X-IAR semantics Formula $\Phi_{X\text{-}IAR}(\mathcal{C})$ from Figure 2 can be used to test whether there exists a X-optimal repair that excludes cause $\mathcal{C}$. To check a potential answer $\vec{a}$, we use $\Phi_{X\text{-}IAR}(q(\vec{a}))$, which is essentially the conjunction of $\Phi_{X\text{-}IAR}(\mathcal{C})$ for every cause $\mathcal{C}$ of $q(\vec{a})$, but where the *conjuncts for different causes use distinct variables* ($x^{\mathcal{C}}_\alpha$ in place of x_α). A multi-answer encoding $\Psi_{X\text{-}IAR}(PotAns)$ can be obtained by taking the conjunction of $\Phi_{X\text{-}IAR}(q(\vec{a}))$ for all $\vec{a} \in PotAns$, but with $\varphi^{\mathcal{C}}_{\neg \mathcal{C}}$ replaced by $\varphi'^{\mathcal{C}}_{\neg \mathcal{C}}(x_{\vec{a}})$ (see the appendix of (Bienvenu and Bourgaux 2022) for details). We obtain the following:

Theorem 6. *The following are equivalent: (i) $\mathcal{K}_\succ \models^X_{IAR} q(\vec{a})$, (ii) $\Phi_{X\text{-}IAR}(q(\vec{a}))$ is unsatisfiable, (iii) $\Phi_{X\text{-}IAR}(\mathcal{C})$ is unsatisfiable for some $\mathcal{C} \in Causes(q(\vec{a}), \mathcal{K})$, and (iv) $x_{\vec{a}}$ is false in every satisfying assignment of $\Psi_{X\text{-}IAR}(PotAns)$.*

We shall also require encodings for individual facts, using

$$\Phi_{X\text{-}IAR}(\alpha) = \Phi_{X\text{-}IAR}(\{\alpha\})$$

to test if α holds in all X-optimal repairs. We also consider a multi-fact version $\Psi_{X\text{-}IAR}(Rel)$, parameterized by a set of facts Rel, and to which we add clause y_α to activate $\varphi'_{\neg\{\alpha\}}(y_\alpha)$.

Theorem 7. *For every $\alpha \in \mathcal{D}$, the following are equivalent: (i) $\mathcal{K}_\succ \models^X_{IAR} \alpha$, (ii) $\Phi_{X\text{-}IAR}(\alpha)$ is unsatisfiable, and (iii) if $\alpha \in Rel$ then y_α is false in every satisfying assignment of $\Psi_{X\text{-}IAR}(Rel)$.*

4 Algorithms

Inspired by the different use of SAT solvers made by CQAPri and CAvSAT, we propose several algorithms based on the encodings of Section 3. The pseudo code of all algorithms is available in (Bienvenu and Bourgaux 2022).

Before describing the algorithms, note that we can show that the encodings of Section 3 are still valid if $Causes(q(\vec{a}), \mathcal{K})$ is actually *a superset of the causes* such that every superfluous $\mathcal{B}$ in it either (1) includes a real cause of $q(\vec{a})$ or (2) contains two distinct facts that form a conflict. Since checking consistency and minimality to obtain the real causes can be costly in practice, our algorithms accept such 'sets of causes'. They actually even accept the presence of self-inconsistent facts in the 'causes' (which would make the encodings for X-AR and X-IAR not applicable) and handle them in a preprocessing step.

Our high-level algorithm takes as input a semantics Sem $\in$ {brave, AR, IAR}, a repair notion X $\in \{S, P, C\}$, a directed conflict graph $\mathcal{G}_{\mathcal{K}_\succ}$, and a set of potential answers $PotAns$ with their causes, and outputs the set of tuples from $PotAns$ that are answers under the X-Sem semantics.

An initial preprocessing step serves to (1) check whether $\mathcal{G}_{\mathcal{K}_\succ}$ contains some self-inconsistent facts, remove them from $\mathcal{G}_{\mathcal{K}_\succ}$, discard all causes that contain such facts, then all answers that do not have any cause left, and (2) find some answers that trivially hold under X-Sem. To do so, it removes from the causes all facts that do not have any outgoing edge in the directed conflict graph, and thus trivially belong to all

$$\Phi_{X\text{-}AR}(q(\vec{a})) = \varphi_{\neg q(\vec{a})} \wedge \varphi_{\mathsf{X\text{-}max}}(F_1) \wedge \varphi_{\mathsf{cons}}(F_2) \qquad F_1 = \mathsf{facts}(\varphi_{\neg q(\vec{a})}), F_2 = F_1 \cup \mathsf{facts}(\varphi_{\mathsf{X\text{-}max}}(F_1))$$

$$\Psi_{X\text{-}AR}(PotAns) = \bigwedge_{\vec{a} \in PotAns} \varphi'_{\neg q(\vec{a})} \wedge \varphi_{\mathsf{X\text{-}max}}(F'_1) \wedge \varphi_{\mathsf{cons}}(F'_2) \qquad F'_1 = \mathsf{facts}(\bigwedge_{\vec{a} \in PotAns} \varphi'_{\neg q(\vec{a})}), F'_2 = F'_1 \cup \mathsf{facts}(\varphi_{\mathsf{X\text{-}max}}(F'_1))$$

$$\Phi_{X\text{-}brave}(q(\vec{a})) = \varphi_{q(\vec{a})} \wedge \varphi_{\mathsf{X\text{-}max}}(G_1) \wedge \varphi_{\mathsf{cons}}(G_2) \qquad G_1 = \mathsf{facts}(\varphi_{q(\vec{a})}), G_2 = G_1 \cup \mathsf{facts}(\varphi_{\mathsf{X\text{-}max}}(G_1))$$

$$\Phi_{X\text{-}brave}(\mathcal{C}) = \varphi_{\mathcal{C}} \wedge \varphi_{\mathsf{X\text{-}max}}(G^*_1) \wedge \varphi_{\mathsf{cons}}(G^*_2) \qquad G^*_1 = \mathsf{facts}(\varphi_{\mathcal{C}}), G^*_2 = G^*_1 \cup \mathsf{facts}(\varphi_{\mathsf{X\text{-}max}}(G^*_1))$$

$$\Psi_{X\text{-}brave}(PotAns) = \bigwedge_{\vec{a} \in PotAns} \varphi'_{q(\vec{a})} \wedge \varphi_{\mathsf{X\text{-}max}}(G'_1) \wedge \varphi_{\mathsf{cons}}(G'_2) \qquad G'_1 = \mathsf{facts}(\bigwedge_{\vec{a} \in PotAns} \varphi'_{q(\vec{a})}), G'_2 = G'_1 \cup \mathsf{facts}(\varphi_{\mathsf{X\text{-}max}}(G'_1))$$

$$\Phi_{X\text{-}IAR}(\mathcal{C}) = \varphi_{\neg \mathcal{C}} \wedge \varphi_{\mathsf{X\text{-}max}}(H_1) \wedge \varphi_{\mathsf{cons}}(H_2) \qquad H_1 = \mathsf{facts}(\varphi_{\neg \mathcal{C}}), H_2 = H_1 \cup \mathsf{facts}(\varphi_{\mathsf{X\text{-}max}}(H_1))$$

$$\Phi_{X\text{-}IAR}(q(\vec{a})) = \bigwedge_{\mathcal{C} \in Causes(q(\vec{a}), \mathcal{K})} \left(\varphi^{\mathcal{C}}_{\neg \mathcal{C}} \wedge \varphi^{\mathcal{C}}_{\mathsf{X\text{-}max}}(H^{\mathcal{C}}_1) \wedge \varphi^{\mathcal{C}}_{\mathsf{cons}}(H^{\mathcal{C}}_2)\right) \qquad H^{\mathcal{C}}_1 = \mathsf{facts}(\varphi^{\mathcal{C}}_{\neg \mathcal{C}}), H^{\mathcal{C}}_2 = H^{\mathcal{C}}_1 \cup \mathsf{facts}(\varphi^{\mathcal{C}}_{\mathsf{X\text{-}max}}(H^{\mathcal{C}}_1))$$

Figure 2: Single- and multi-answer encodings for X-AR (top) and X-brave (middle) semantics, single-answer encodings for X-IAR (bottom).

optimal repairs. The *trivial answers* that have some cause that contains only such facts hold under X-Sem semantics and are filtered during this step.

It then remains to filter the remaining potential answers. The four first algorithms we propose to do so are generic in the sense that they can be used for all semantics.

- Simple is similar to the algorithm used by CQAPri. For each answer to filter $\vec{a}$, it checks whether $\Phi_{X\text{-Sem}}(q(\vec{a}))$ is satisfiable, which decides whether $\mathcal{K}_\succ \models^X_{\text{Sem}} q(\vec{a})$.
- All-MaxSAT is similar to the CAvSAT algorithm. It constructs a weighted MaxSAT instance $\Psi_{X\text{-Sem}}(PotAns) \wedge \bigwedge_{\vec{a} \in PotAns} x_{\vec{a}}$, where the $x_{\vec{a}}$ are soft clauses, and all other clauses are hard. It then relies on the solver to maximize the number of soft clauses satisfied, which filters the corresponding answers. After each iteration, the $\neg x_{\vec{a}}$ corresponding to the satisfied soft clauses are added to the set of assumed literals for the next iteration.
- All-MUSes is based on the observation that if $x_{\vec{a}}$ is false in every satisfying assignment of $\Psi_{X\text{-Sem}}(PotAns)$, then $\{x_{\vec{a}}\}$ is a minimal unsatisfiable subset (MUS) of $\bigwedge_{\vec{a} \in PotAns} x_{\vec{a}}$ w.r.t. $\Psi_{X\text{-Sem}}(PotAns)$. All-MUSes relies on the solver to compute all MUSes, and decides whether $\mathcal{K}_\succ \models^X_{\text{Sem}} q(\vec{a})$ by looking at those of size one.
- Assumptions iteratively evaluates $\Psi_{X\text{-Sem}}(PotAns)$, treating the variables $x_{\vec{a}}$, with $\vec{a} \in PotAns$ as assumptions. If $\Psi_{X\text{-Sem}}(PotAns)[x_{\vec{a}}]$ is satisfiable, there exists a satisfying assignment of $\Psi_{X\text{-Sem}}(PotAns)$ in which $x_{\vec{a}}$ is true, which decides whether $\mathcal{K}_\succ \models^X_{\text{Sem}} q(\vec{a})$.

While we may need to consider all causes to decide whether an answer holds under X-AR semantics, in the X-brave or X-IAR case it is sufficient to find a single cause that belongs to some or all optimal repairs. Moreover, the encoding $\Phi_{X\text{-}IAR}(q(\vec{a}))$ is a conjunction of independent subproblems built on distinct variables for each cause, which does not seem very fit for a SAT solver. We hence propose algorithms specific to these cases.

- Cause-by-cause can be used for X-brave and X-IAR. For each answer to filter $\vec{a}$, it checks whether there is a cause $\mathcal{C}$ of $q(\vec{a})$ such that $\Phi_{X\text{-Sem}}(\mathcal{C})$ is (un)satisfiable: if Sem $=$ brave and $\Phi_{X\text{-Sem}}(\mathcal{C})$ is satisfiable, or Sem $=$ IAR and $\Phi_{X\text{-Sem}}(\mathcal{C})$ is unsatisfiable, then $\mathcal{K}_\succ \models^X_{\text{Sem}} q(\vec{a})$; if no cause witnesses $\mathcal{K}_\succ \models^X_{\text{Sem}} q(\vec{a})$, then $\mathcal{K}_\succ \not\models^X_{\text{Sem}} q(\vec{a})$.
- IAR-causes is specific to X-IAR. It considers the answers in turn while maintaining two sets of facts: the X-IAR facts that belong to the intersection of the optimal repairs and the non-X-IAR facts. For each cause $\mathcal{C}$ of $q(\vec{a})$ that does not contain any known non X-IAR fact, it removes the known X-IAR facts from $\mathcal{C}$. If $\mathcal{C}$ becomes empty, then $\mathcal{K}_\succ \models^X_{\text{IAR}} q(\vec{a})$. Otherwise, for each remaining $\alpha \in \mathcal{C}$, it checks whether $\mathcal{K}_\succ \models^X_{\text{IAR}} \alpha$ using $\Phi_{X\text{-}IAR}(\alpha)$ and adds α to the corresponding set of facts. If every $\alpha \in \mathcal{C}$ is such that $\mathcal{K}_\succ \models^X_{\text{IAR}} \alpha$, then $\mathcal{K}_\succ \models^X_{\text{IAR}} q(\vec{a})$.
- IAR-facts is also specific to X-IAR and considers the answers in turn while maintaining the two sets of X-IAR and non X-IAR facts. The difference is that for each answer, it uses $\Psi_{X\text{-}IAR}(Rel) \wedge \bigwedge_{\alpha \in Rel} y_\alpha$ and a weighted MaxSAT solver to decide which facts hold under X-IAR among the set Rel of facts that belong to some cause and have not already been assigned to one of the two sets. Then it checks whether there is a cause that only contains X-IAR facts.

5 Implementation & Experimental Setting

We implemented the algorithms presented in Section 4 in java 11. Our system ORBITS (Optimal Repair-Based Inconsistency-Tolerant Semantics) takes as input two JSON files containing the directed conflict graph $\mathcal{G}_{\mathcal{K}_\succ}$, and the potential answers $PotAns$ of the query associated with their causes. The user specifies a semantics (AR, IAR, or brave), a value X (among S, P_1, P_2, or C) to use in $\varphi_{\mathsf{X\text{-}max}}(F)$, the desired encoding for $\varphi_{\neg \mathcal{C}}$ (neg_1 or neg_2), and the algorithm to use to compute the answers w.r.t. the chosen semantics. The set of answers is output as a JSON file.

ORBITS relies on the Sat4j java library (version 2.3.4) to solve the SAT, weighted MaxSAT, and MUS enumeration problems (Berre and Parrain 2010). In principle, a standalone solver could be used, but we found that the time needed to print out the encoding to pass it to an external solver tends to be prohibitive compared to using Sat4j.

The source code of ORBITS is available at https://github.com/bourgaux/orbits, the inputs files we used in the experiments at https://zenodo.org/record/5946827, and details on the experimental setting in (Bienvenu and Bourgaux 2022).

Experimental Environment All experiments were run with 16GB of RAM in a cluster node running CentOS 7.9 with linux kernel 3.10.0, with processor 2x Cascade Lake Intel Xeon 5218 16 cores, 2.4GHz. Reported times are averaged over 5 runs, with a 30 minutes time-out. Since we aim at comparing our different algorithms and encodings, in what follows, we focus on the time needed to filter the candidate answers, excluding the time needed to load the inputs from the JSON files or serialize the output. This input loading time is generally below 1 second and never exceeds a few seconds. However, in real-world applications, we would not use ORBITS as a standalone tool, but rather make it a library to be integrated in a full query answering system.

Test KBs We evaluate ORBITS on three (sets of) KBs. The first is the CQAPri benchmark (Bourgaux 2016), a synthetic benchmark crafted to evaluate inconsistency-tolerant query answering over DL-Lite KBs, adapted from the $\text{LUBM}^{\exists}_{20}$ benchmark (Lutz et al. 2013). The two others, called Food Inspection and Physicians, are real-world datasets built from public open data, which have already been used to evaluate data cleaning and consistent query answering systems (Rekatsinas et al. 2017; Dixit and Kolaitis 2019). They consist of relational databases built from the original csv files, on which typical integrity constraints have been added. We briefly summarize their main characteristics below.

We use the DL-Lite ontology (which includes 875 disjointness axioms) and 20 queries of the CQAPri benchmark, together with the 18 datasets named uXcY with $X \in \{1, 5, 20\}$ and $Y \in \{1, 5, 10, 20, 30, 50\}$. Parameters X and Y are related to the size and the proportion of facts involved in some conflicts respectively (the higher the bigger), and the datasets are such that $\mathsf{uXcY} \subseteq \mathsf{uXcY'}$ for $Y \leq Y'$ and $\mathsf{uXcY} \subseteq \mathsf{uX'cY}$ for $X \leq X'$. Their sizes range from 75K to 2M facts and their proportions of facts involved in some conflict from 3% to 46%. The induced conflict graphs contain from 2K to 946K facts and from 2K to 3M conflicts.

The Food Inspection dataset contains data about inspection of restaurants in New York and Chicago (Dat c; Dat a). We use the database schema and six queries proposed by Dixit and Kolaitis (2019): there are four relations, each having a key constraint and one having a further FD. The dataset contains 523K facts, 37% of them belong to some conflict. The conflict graph contains 192K facts and 219K conflicts.

We build the Physicians dataset from the National Downloadable File provided by the Centers for Medicare & Medicaid Services (Dat b). It contains information on medical professionals and their affiliations. We decompose it into seven relations, add four reasonable key constraints and two FDs, and design six queries. In total, the dataset contains more than 8M facts and 2% of them are in some conflict. The conflict graph contains 183K facts and 2.7M conflicts.

Priority Relations We build score-structured priority relations by randomly assigning each fact a score between 1 and n. To construct a non-score-structured priority relation, we consider each conflict and assign a random direction to the corresponding edge in the conflict graph with a probability p, except if doing so creates a cycle, and verify that the resulting priority is indeed not score-structured. On the Food Inspection and Physicians datasets, we build four priority relations: two score-structured with $n = 2$ and $n = 5$, and two non-score-structured with $p = 0.5$ and $p = 0.8$. For the CQAPri benchmark, we build two priority relations, one score-structured with $n = 5$ and one non-score-structured with $p = 0.8$, on our largest dataset ($\mathsf{u20c50}$), then propagate them to the other datasets.

	Triv.	IAR\Triv.	AR\IAR	brave\AR	not brave
S	1,655		7	77	0
{P,C}-2	1,668	0	7	48	16
{P,C}-5	1,679	1	7	29	23
P-0.5	1,671	1	7	45	15
C-0.5	1,671	23	0	30	15
P-0.8	1,680	5	7	25	22
C-0.8	1,680	24	0	13	22

Table 1: Number of answers of $\mathsf{q1}$ over the Physicians dataset depending on the priority relation (none, score-structured with $n = 2$ or $n = 5$, and not score-structured with $p = 0.5$ or $p = 0.8$) and type of repairs (standard, Pareto- or completion-optimal).

6 Experimental Evaluation

Our experimental evaluation aims at assessing (i) the impact of adopting different kinds of repairs, and (ii) the relative performances of alternative procedures for the same semantics. More precisely, we consider the following questions.

- What is the impact in terms of number of answers of adopting optimal repairs rather than standard repairs, or completion-optimal repairs instead of Pareto-optimal repairs when the priority relation is not score-structured?
- What is the impact of using one kind of repairs rather than another on the computation time?
- Given a semantics and type of repair, what is the impact in terms of computation times of the choice of:
  - how to encode optimality (P_1 or P_2 for Pareto-optimal repairs, P_1, P_2 or C when $\succ$ is score-structured)?
  - how to encode contradictions ($\varphi_{\neg\mathcal{C}}$ with neg_1 or neg_2)?
  - the algorithm used to filter the non-trivial answers?

In what follows, we summarize our main observations. Detailed results are given in (Bienvenu and Bourgaux 2022).

Comparing Semantics w.r.t. Number of Answers Table 1 shows the impact on the number of answers of the type of priority relation and chosen notion of optimal repairs for an example query. For each priority relation and repair type X, it gives the number of answers that: are trivially X-IAR (i.e. some cause contains only facts without outgoing edges in the directed conflict graph), hold under X-IAR but not trivially, hold under X-AR but not under X-IAR, hold under X-brave but not X-AR, and do not hold under X-brave semantics.

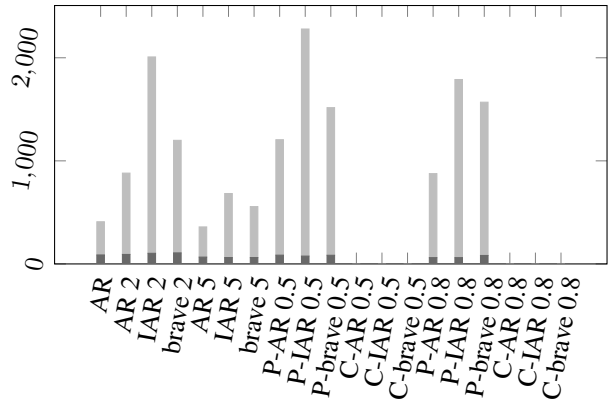


Figure 3: Best running times (in milliseconds) for each semantics and priority relation (none, score-structured with $n = 2$ or $n = 5$, not score-structured with $p = 0.5$ or $p = 0.8$) for query q2 over the Food Inspection dataset. An empty bar means that the query ran out of time / memory for all possible algorithms and encodings. The lower part of bars is the time to identify self-inconsistent facts and trivial answers, the upper part the time to filter non-trivial answers.

Priority relations leads to fewer edges in the directed conflict graph, which in turn makes more answers hold trivially. The more the priority relation sets preferences between facts (higher parameter n or p of the priority relation), the more trivially X-IAR answers we obtain. Adopting optimal repairs also significantly increases the number of potential answers that do not hold under X-brave semantics, which are very rare when using classical subset repairs.

An interesting observation is that while, in the absence of a priority relation, many queries of the CQAPri benchmark do not have any AR answers that are not trivial, which makes trivial answers a good approximation of AR, this is no longer the case for optimal repairs. It hence seems even more important to actually compute the answers that hold under the desired semantics rather simply computing the polynomial lower bound given by the trivial answers.

Regarding the impact of the choice between Pareto- and completion-optimal repairs, in many cases ORBITS did not manage to compute the answers for completion-optimal repairs in our given time and memory limits. When it does manage to compute them (for 5 queries on the Physicians dataset; none on the Food Inspection dataset; and from between 7 and 13 on uXc1 to less than 3 on uXc50), we observe a difference with Pareto-answers in only two cases.

Comparing Semantics w.r.t. Computation Time Figure 3 shows the best running times (across all algorithms and encoding variants) for each semantics and an example query. We did this comparison for all queries on Physicians and Food Inspection datasets and two CQAPri datasets.

Given a kind of repair X, the relative difficulty of the X-AR, X-IAR and X-brave semantics depends on the query, dataset and sometimes the nature of the priority relation.

Comparing S-AR, P-AR, and C-AR semantics, we observe that using optimal repairs may either increase or decrease the answer filtering time, intuitively because there are more trivial answers, but the encodings are more complex.

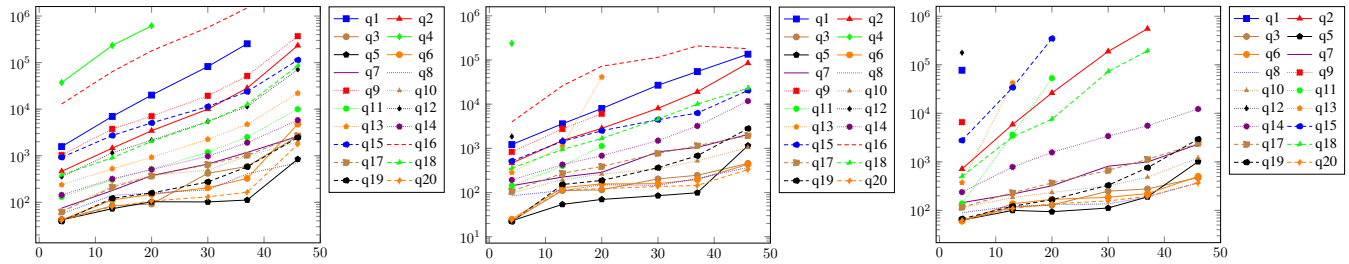
Given a dataset, query and semantics, if we compare two priority relations of the same kind (score-structured or not), the one that sets preferences between more facts leads to lower running times. This can be explained by the increase of the number of trivial answers and maybe by the fact that the encodings involve less facts and encode more 'forced choices' between facts so that there are less possibilities to explore. When we compare the 'easiest' non score-structured priority relations ($p = 0.8$) and the hardest score-structured ones ($n = 2$), which lead to comparable sizes of directed conflict graphs, score-structured priority relations seem to be easier than non-score-structured ones.

Finally, we conclude that our procedures do not perform well for completion-optimal repair-based semantics, which form most of the cases that fail due to lack of time or memory, and very often have higher running times than Pareto-optimal-based semantics with the same priority relation.

		q1	q2	q3	q4	q5	q6
Alg. 1	P_1	417	141	350	12,799	224	4,009
	P_2	804	142	379	326,594	213	8,684
	C	252,694	179	550	oom	284	t.o
Alg. 2	P_1	**268**	166	326	**1,730**	214	11,263
	P_2	502	163	333	2,961	221	10,833
	C	oom	632	t.o	t.o	551	oom
Alg. 3	P_1	272	154	313	t.o	211	245,804
	P_2	466	146	**281**	t.o	**201**	241,030
	C	oom	624	t.o	t.o	550	oom
Alg. 4	P_1	362	166	997	42,544	281	559,923
	P_2	566	193	972	36,923	304	546,199
	C	oom	764	t.o	t.o	846	oom
Alg. 5	P_1	383	**135**	335	8,192	211	**3,419**
	P_2	565	157	309	225,170	207	5,963
	C	192,429	164	544	oom	238	t.o

Table 2: Query answer filtering time (in milliseconds, t.o:time out, oom:out of memory) under X-brave semantics (X $\in$ {P,C}), for each algorithm and encoding $\varphi_{\text{X-max}}$, on Physicians dataset with score-structured priority ($n = 2$). Alg. 1: Simple, Alg. 2: All-MaxSAT, Alg. 3: All-MUSes, Alg. 4: Assumptions, Alg. 5: Cause-by-cause. Best time in bold red and 'close to best times' (i.e., not exceeding the best by more than 50ms or 10%) on grey.

Choice of Algorithm & Encoding for Given Semantics Table 2 presents the time needed to filter the candidate answers that hold under brave semantics based on optimal-repairs (score-structured case) for some example queries. It illustrates the huge impact that the choice of an algorithm and encoding can have. For example, q6 answers are filtered in 3.5s with algorithm Cause-by-cause and $\varphi_{\text{P}_1\text{-max}}$, but need at least 546s with the Assumptions algorithm, at least 5.9s with $\varphi_{\text{P}_2\text{-max}}$, and cannot be filtered in our time and memory limits with encoding $\varphi_{\text{C-max}}$. For X-AR and X-IAR semantics, we also observe sometimes huge variations when using the neg_1 or neg_2 version of $\varphi_{\neg\mathcal{C}}$. For example, for X-AR semantics on u20c50 with a score-structured priority relation, the best times for queries q2 and q18 are both obtained



(a) Simple (b) All-MaxSAT (c) All-MUSes

Figure 4: Time (in milliseconds, log. scale) to filter query answers under X-AR semantics (X $\in$ {P,C}) w.r.t. percentage of facts involved in some conflict for u20cY with score-structured priority relation ($\varphi_{\mathsf{P_1\text{-}max}}$ and neg_1 encoding). Missing queries ran out of time or memory.

with algorithm All-MaxSAT and $\varphi_{\mathsf{P_1\text{-}max}}$, but with the neg_2 variant for q2 (50 seconds versus 85 with neg_1) and neg_1 for q18 (23 seconds versus 47 with neg_2). The comparison of the possible procedures for each semantics on the different datasets and queries shows that there is not a 'best' method in general. However, we still gain some relevant insights.

The first one concerns the choice of $\varphi_{\mathsf{X\text{-}max}}$. The encoding $\varphi_{\mathsf{P_1\text{-}max}}$ is generally the best one for Pareto-optimal repairs, in the sense that it achieves 'close to best times' (cf. Table 2) much more often than the others. However, there are a few cases where $\varphi_{\mathsf{P_2\text{-}max}}$ performs significantly better, especially on the CQAPri datasets with fewer conflicts (e.g., on u20c1 with a score-structured priority relation, the best time for filtering q9 answers under P-AR semantics is 480ms with $\varphi_{\mathsf{P_2\text{-}max}}$, while the best time with another encoding is 825ms). When the priority relation is score-structured, $\varphi_{\mathsf{C\text{-}max}}$ never significantly outperforms $\varphi_{\mathsf{P_1\text{-}max}}$ and $\varphi_{\mathsf{P_2\text{-}max}}$ and leads to much more time or memory failures.

The second concerns the choice of algorithm for X-IAR semantics. For all kinds of repairs, algorithm IAR-causes is generally better than the others in terms of frequency of 'close to best times'. It is sometimes outperformed by algorithms Cause-by-cause or IAR-facts. The 'generic' algorithms (Simple, All-MaxSAT, All-MUSes, Assumptions) perform quite poorly, except on the simplest cases.

For the AR and brave semantics, it is more difficult to find an algorithm that is superior to the others. For non-score-structured priority relations and completion-optimal repairs, algorithm Simple seems to be the best choice for both C-AR and C-brave semantics, but all algorithms fail in most cases. A related observation is that algorithms that consider answers individually and use smaller encodings seems to be often more robust in terms of time-out and out-of-memory, with a notable exception for P-AR and P-brave semantics on u20c50 with non-score-structured priority, where algorithms All-MaxSAT and All-MUSes are more robust. For S-AR, P-AR and P-brave, we observe different behaviours depending on the benchmark: All-MaxSAT and All-MUSes tend to perform better for the CQAPri benchmark, while Simple tends to perform better for the Food Inspection dataset.

Finally, comparing neg_1 and neg_2 versions of $\varphi_{\neg\mathcal{C}}$, we observe that the relative performance depends on the dataset, query, algorithm, and choice of $\varphi_{\mathsf{X\text{-}max}}$. However, we note that $\varphi_{\mathsf{P_2\text{-}max}}$ usually works better with neg_1. Even if there is not a direct relationship between the encoding sizes and the running times, this is probably due to the fact that neg_2 enforces that both the facts that occur in the causes and their conflicts are part of the encoding, which significantly increases the size of $\varphi_{\mathsf{P_2\text{-}max}}$, but has little impact on $\varphi_{\mathsf{P_1\text{-}max}}$.

Figure 4 shows the evolution of the running times of three algorithms using the same encoding variants as the proportion of facts involved in some conflicts grows, on the u20cY datasets with a score-structured priority for X-AR semantics (X $\in$ {P,C}). It illustrates the fact that the relative performance of the algorithms depends on the query and dataset (here, the proportion of facts involved in some conflict): For example, All-MaxSAT is the best for q9 over the three first datasets, but runs out of time on the three last (more that 20% of facts in conflict), while Simple can handle them.

7 Conclusion

We have presented a comprehensive exploration of SAT-based approaches to querying inconsistent data using (optimal) repair-based inconsistency tolerant semantics, including the proposal of novel encodings and algorithms. Our generic framework places existing approaches into a broader context and makes our results and system directly applicable to both the (pure) database and OMQA settings.

Our experimental comparison of different SAT-based algorithms and encoding variants shows that the choice of algorithm and encoding may have huge impact on the computation time. While in some cases our results can be used to single out some approaches as more effective, more often there are no clear winner(s). This suggests that to minimize runtimes, it may make sense to launch multiple algorithms in parallel, and/or devise methods that can help predict which algorithm and encoding will perform best on a given dataset and query, e.g. using machine learning techniques.

Our work lays important foundations for the future development of mature systems for querying inconsistent data. We plan to investigate different ways of improving the performance for optimal repair-based semantics. For example, it would be interesting to explore alternative approaches for completion-optimal repairs based upon SAT modulo graph techniques (Gebser, Janhunen, and Rintanen 2014). Another promising direction is to employ more refined polynomial lower approximations than the trivial answers, such as the grounded semantics (Bienvenu and Bourgaux 2020).

Acknowledgements

This work was supported by the ANR AI Chair INTENDED (ANR-19-CHIA-0014).

References

Arenas, M.; Bertossi, L. E.; and Chomicki, J. 1999. Consistent query answers in inconsistent databases. In *Proceedings of the 18th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS)*, 68–79.

Berre, D. L., and Parrain, A. 2010. The sat4j library, release 2.2. *JSAT* 7(2-3):59–64.

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

Bienvenu, M., and Bourgaux, C. 2016. Inconsistency-tolerant querying of description logic knowledge bases. In *Tutorial Lectures of the 12th International Reasoning Web Summer School*, 156–202.

Bienvenu, M., and Bourgaux, C. 2020. Querying and repairing inconsistent prioritized knowledge bases: Complexity analysis and links with abstract argumentation. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 141–151.

Bienvenu, M., and Bourgaux, C. 2022. Querying inconsistent prioritized data with ORBITS: Algorithms, implementation, and experiments. arxiv.org/abs/2202.07980 [cs.LO].

Bienvenu, M., and Ortiz, M. 2015. Ontology-mediated query answering with data-tractable description logics. In *Tutorial Lectures of the 11th Reasoning Web International Summer School*, 218–307.

Bienvenu, M., and Rosati, R. 2013. Tractable approximations of consistent query answering for robust ontology-based data access. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*.

Bienvenu, M.; Bourgaux, C.; and Goasdoué, F. 2014. Querying inconsistent description logic knowledge bases under preferred repair semantics. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI)*, 996–1002.

Bienvenu, M.; Bourgaux, C.; and Goasdoué, F. 2019. Computing and explaining query answers over inconsistent DL-Lite knowledge bases. *Journal of Artificial Intelligence Research (JAIR)* 64:563–644.

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

Bourgaux, C. 2016. *Inconsistency Handling in Ontology-Mediated Query Answering. (Gestion des incohérences pour l'accès aux données en présence d'ontologies)*. Ph.D. Dissertation, University of Paris-Saclay, France.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *Journal of Automated Reasoning (JAR)* 39(3):385–429.

Dataset: Food Inspections, Chicago Data Portal. https://data.cityofchicago.org/Health-Human-Services/Food-Inspections/4ijn-s7e5. Accessed December 7, 2020.

Dataset: National Downloadable File, Centers for Medicare & Medicaid Services. https://data.cms.gov/provider-data/dataset/mj5m-pzi6. Accessed December 10, 2020.

Dataset: New York City Restaurant Inspection Results, Department of Health and Mental Hygiene (DOHMH), NYC Open Data. https://data.cityofnewyork.us/Health/DOHMH-New-York-City-Restaurant-Inspection-Results/43nn-pn8j. Accessed December 7, 2020.

Dixit, A. A., and Kolaitis, P. G. 2019. A SAT-based system for consistent query answering. In *Proceedings of the 22nd International Conference on Theory and Applications of Satisfiability Testing (SAT)*, 117–135.

Fan, W. 2015. Data quality: From theory to practice. *SIGMOD Rec.* 44(3):7–18.

Gebser, M.; Janhunen, T.; and Rintanen, J. 2014. SAT modulo graphs: Acyclicity. In *Proceedings of the 14th European Conference on Logics in Artificial Intelligence (JELIA)*, 137–151.

Lembo, D.; Lenzerini, M.; Rosati, R.; Ruzzi, M.; and Savo, D. F. 2010. Inconsistency-tolerant semantics for description logics. In *Proceedings of the 4th International Conference on Web Reasoning and Rule Systems (RR)*, 103–117.

Livshits, E., and Kimelfeld, B. 2017. Counting and enumerating (preferred) database repairs. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS)*, 289–301.

Lutz, C.; Seylan, I.; Toman, D.; and Wolter, F. 2013. The combined approach to OBDA: Taming role hierarchies using filters. In *Proceedings of the 12th International Semantic Web Conference (ISWC)*, 314–330.

Poggi, A.; Lembo, D.; Calvanese, D.; De Giacomo, G.; Lenzerini, M.; and Rosati, R. 2008. Linking data to ontologies. *Journal of Data Semantics* 10:133–173.

Rekatsinas, T.; Chu, X.; Ilyas, I. F.; and Ré, C. 2017. Holoclean: Holistic data repairs with probabilistic inference. *Proceedings of the VLDB Endowment (PVLDB)* 10(11):1190–1201.

Rosati, R. 2011. On the complexity of dealing with inconsistency in description logic ontologies. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, 1057–1062.

Staworko, S.; Chomicki, J.; and Marcinkowski, J. 2012. Prioritized repairing and consistent query answering in relational databases. *Annals of Mathematics and Artifcial Intelligence (AMAI)* 64(2-3):209–246.

Wijsen, J. 2019. Foundations of query answering on inconsistent databases. *SIGMOD Record* 48(3):6–16.

Xiao, G.; Calvanese, D.; Kontchakov, R.; Lembo, D.; Poggi, A.; Rosati, R.; and Zakharyaschev, M. 2018. Ontology-based data access: A survey. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, 5511–5519.