Data-graph repairs: the preferred approach

Sergio Abriola[1,2], Santiago Cifuentes[1,2], Nina Pardal[2,3] and Edwin Pin[1,2]

[1] Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Argentina
[2] ICC CONICET, Buenos Aires, Argentina
[3] Department of Computer Science, University of Sheffield, UK

Abstract

Repairing inconsistent knowledge bases is a task that has been assessed, with great advances over several decades, from within the knowledge representation and reasoning and the database theory communities. As information becomes more complex and interconnected, new types of repositories, representation languages and semantics are developed in order to be able to query and reason about it. Graph databases provide an effective way to represent relationships among data, and allow processing and querying these connections efficiently. In this work, we focus on the problem of computing preferred (subset and superset) repairs for graph databases with data values, using a notion of consistency based on a set of Reg-GXPath expressions as integrity constraints. Specifically, we study the problem of computing preferred repairs based on two different preference criteria, one based on weights and the other based on multisets, showing that in some cases it is possible to retain the same computational complexity as in the case where no preference criterion is available for exploitation.

Keywords
Data-graphs, Repairs, Preferences

1. Introduction

Graph databases are useful in many modern applications where the topology of the data is as important as the data itself, such as social networks analysis [1], data provenance [2], and the Semantic Web [3]. The structure of the database is commonly queried through navigational languages such as *regular path queries* or RPQs [4] that can capture pair of nodes connected by some specific kind of path. This query languages can be extended to add more expressiveness, while usually adding extra complexity in the evaluation as well. For example, C2RPQs are a natural extension of RPQs defined by adding to the language the capability of traversing edges backwards and closing the expressions under conjunction (similar to relational CQs).

RPQs and its most common extensions (C2RPQs and NREs [5]) can only act upon the edges of the graph, leaving behind any possible interaction with data values in the nodes. This led to the design of query languages for *data-graphs* (i.e. graph databases where data lies both in the paths and in the nodes themselves), such as REMs and Reg-GXPath [6].

As in the relational case, it is common to expect that the data preserves some semantic structure related to the world it represents. These *integrity constraints* can be expressed in graph databases through *path constraints* [7, 8].

AMW 2023: 15th Alberto Mendelzon International Workshop on Foundations of Data Management, May 22–26, 2023, Santiago, Chile

When a database does not satisfy its integrity constraints, a possible approach is to search for a 'similar' database that does satisfy the constraints. In the literature, this new database is called a *repair* [9], and in order to define it properly one has to precisely define the meaning of 'similar'.

In the literature one can find different notions of repairs, among others, set-based repairs [10], attribute-based repairs [11], and cardinality based repairs [12]. When considering set-based repairs G' of a graph database G under a set of Reg-GXPath expressions R, two natural restrictions of the problem are when G' is a sub-graph of G and when G' is a super-graph of G. These kind of repairs are usually called *subset* and *superset* repairs respectively [10, 13]. Since repairs may not be unique, it is possible to impose an ordering over the set of repairs and look for an 'optimum' repair over such ordering. There is a significant body of work on preferred repairs for relational databases [14, 15] and other types of logic-based formalisms [16, 17]. However, to the best of our knowledge, there is no such work focused on graph databases or data-graphs. In this work, we study the problem of finding a preferred repair based on two preference criteria that we propose.

This work is organized as follows. In Section 2 we introduce the necessary preliminaries and notation for the syntax and semantics for our data-graph model as well as the definitions of consistency and different types of repairs. In Section 3 we develop two different proposals to assign preferences to repairs. The first one is based on the assignment of weights, and the second one is based on lifting an ordering over edges and data to multiset orderings. For both proposals we study the computational complexity of the problem of computing a preferred repair. Conclusions and future work directions are discussed in Section 4.

2. Definitions

Fix a finite set of edge labels Σ_e and a countable (either finite or infinite enumerable) set of data values Σ_n (sometimes called data labels), which we assume non-empty and with $\Sigma_e \cap \Sigma_n = \emptyset$. A **data-graph** G is a tuple (V, L_e, D) where V is a set of nodes, L_e is a mapping from $V \times V$ to $\mathcal{P}(\Sigma_e)$ defining the edges of the graph, and D is a mapping from V to the set of data values Σ_n.

Reg-GXPath *expressions* are given by the following mutual recursion:

$$\begin{aligned}\varphi, \psi &:= \textsc{c}^{=} \mid \textsc{c}^{\neq} \mid \neg\varphi \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid \langle\alpha\rangle \mid \langle\alpha = \beta\rangle \mid \langle\alpha \neq \beta\rangle \\ \alpha, \beta &:= \epsilon \mid \_ \mid \textsc{a} \mid \textsc{a}^{-} \mid \alpha \circ \beta \mid \alpha \cup \beta \mid \alpha \cap \beta \mid \alpha^{*} \mid \overline{\alpha} \mid [\varphi] \mid \alpha^{n,m}\end{aligned}$$

where $\textsc{c}$ iterates over Σ_n and $\textsc{a}$ iterates over Σ_e. Formulas like φ are called *node expressions* and formulas like α are called *path expressions.* The subset of Reg-GXPath called Core-GXPath is obtained by allowing the Kleene star to be applied only to labels and their inverses (i.e. $\textsc{a}^{-}$). The semantics of these languages are defined in [6] in a similar fashion as the usual regular languages for navigating graphs [4], also adding some extra capabilities such as the complement of a path expression $\overline{\alpha}$ and data tests. The $\langle\alpha\rangle$ operator is the usual one for *nested regular expressions* (or NREs) used in [5]. Given a data-graph $G = (V, L, D)$, the semantics are:

$$\begin{aligned}&[\![\epsilon]\!]_G = \{(v, v) \mid v \in V\} && [\![\_]\!]_G = \{(v, w) \mid L(v, w) \neq \emptyset\} \\ &[\![\textsc{a}]\!]_G = \{(v, w) \mid \textsc{a} \in L(v, w)\} && [\![\textsc{a}^{-}]\!]_G = \{(w, v) \mid \textsc{a} \in L(v, w)\}\end{aligned}$$

$$
\begin{array}{ll}
[\![\alpha^*]\!]_G = \text{the reflexive transitive closure of } [\![\alpha]\!]_G & \\
[\![\alpha \star \beta]\!]_G = [\![\alpha]\!]_G \star [\![\beta]\!]_G \text{ for } \star \in \{\circ, \cup, \cap\} & \\
[\![\overline{\alpha}]\!]_G = V \times V \setminus [\![\alpha]\!]_G & [\![[\varphi]]\!]_G = \{(v, v) \mid v \in [\![\varphi]\!]_G\} \\
[\![\mathrm{c}^=]\!]_G = \{v \in V \mid D(v) = \mathrm{c}\} & [\![\mathrm{c}^{\neq}]\!]_G = \{v \in V \mid D(v) \neq \mathrm{c}\} \\
[\![\varphi \wedge \psi]\!]_G = [\![\varphi]\!]_G \cap [\![\psi]\!]_G & [\![\varphi \vee \psi]\!]_G = [\![\varphi]\!]_G \cup [\![\psi]\!]_G \\
[\![\neg\varphi]\!]_G = V \setminus [\![\varphi]\!]_G & [\![\langle\alpha\rangle]\!]_G = \{v \mid \exists w \in V,\, (v, w) \in [\![\alpha]\!]_G\} \\
[\![\langle\alpha = \beta\rangle]\!]_G = \{v \mid \exists u, w,\, (v, u) \in [\![\alpha]\!]_G,\, (v, w) \in [\![\beta]\!]_G,\, D(u) = D(w)\} & \\
[\![\langle\alpha \neq \beta\rangle]\!]_G = \{v \mid \exists u, w,\, (v, u) \in [\![\alpha]\!]_G,\, (v, w) \in [\![\beta]\!]_G,\, D(u) \neq D(w)\} &
\end{array}
$$

We use $\alpha \Rrightarrow \beta$ to denote the path expression $\beta \cup \overline{\alpha}$, and $\varphi \Rightarrow \psi$ to denote the node expression $\psi \vee \neg\varphi$. We also note a label $\textsc{a}$ as $\downarrow_{\textsc{a}}$ in order to easily distinguish the 'path' fragment of the expressions. For example, the expression $\textsc{a}[\mathrm{c}^=]\textsc{a}$ will be noted as $\downarrow_{\textsc{a}} [\mathrm{c}^=] \downarrow_{\textsc{a}}$. Naturally, the expression $\alpha \cap \beta$ can be rewritten as $\overline{\overline{\alpha} \cup \overline{\beta}}$ while preserving the semantics. Something similar happens with the operators $\wedge$ and $\vee$ for the case of node expressions using the $\neg$ operator. We define all these operators in this grammar since further on we will be interested in a fragment of Reg-GXPath called Reg-GXPath^{pos}, which has the same grammar except for the $\overline{\alpha}$ and $\neg\varphi$ productions. Thus, in Reg-GXPath^{pos} we will not be able to 'simulate' the $\cap$ operator unless it is present in the original Reg-GXPath grammar.

We will also denote by Reg-GXPath$^{pos}_{node}$ the subset of Reg-GXPath^{pos} that only contains node expressions.

Consistency Given a specific database, we want node or path expressions to represent some structural property we expect to find in our data. This kind of Core-GXPath or Reg-GXPath expression works as an *integrity constraint* by defining semantic relations among our data. Formally, we define the notion of consistency in the following way:

Definition 1 (Consistency)**.** Let G be a data-graph and $R = P \cup N$ a set of restrictions, where P consists of path expressions and N of node expressions. We say that G is **consistent** w.r.t. R, denoted by $G \models R$, if the following conditions hold: *(a)* for all $\varphi \in N$, we have that $[\![\varphi]\!] = V_G$, *(b)* for all $\alpha \in P$, we have that $[\![\alpha]\!] = V_G \times V_G$. Otherwise, we say that G is **inconsistent** w.r.t. R.

In the rest of the paper, we will simply say that G is (in)consistent whenever the restriction set R is clear from the context.

Example 2. Consider the film database from Figure 1, where we have nodes representing people from the film industry (such as actors or directors) and others representing movies or documentaries.

If we want to make a cut from that graph that preserves only actors who have worked with Philip Seymour Hoffman through a film by Paul Thomas Anderson, then we want the following formula to be satisfied:

$$\varphi = \langle \downarrow_{\textsc{type}} [\mathit{actor}^=] \rangle \Rightarrow \langle \downarrow_{\textsc{acts\_in}} \langle \downarrow_{\textsc{directed\_by}} [\mathit{Anderson}^=] \rangle \downarrow^{-}_{\textsc{acts\_in}} [\mathit{Hoffman}^=] \rangle.$$

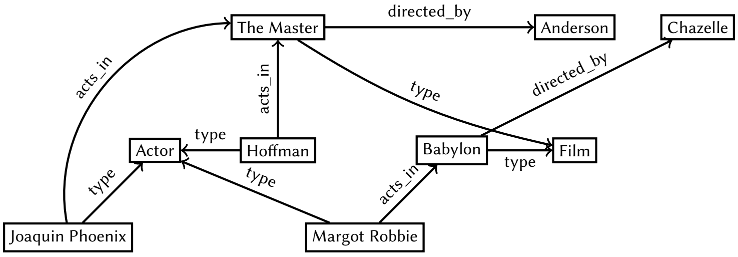

Figure 1: A film data-graph.

Notice that φ is not satisfied in the depicted data-graph, since Robbie did not work with Hoffman in a film directed by Anderson, hence we do not have consistency with respect to $\{\varphi\}$. Observe that the restriction also applies to Hoffman, thus it is required that he participates in at least one film by Anderson in order to satisfy the constraint.

Repairs If a graph database G is inconsistent with respect to a set of restrictions R (i.e. there is a path expression or node expression in R that is not satisfied), we would like to compute a new graph database G' consistent with respect to R that minimally differs from G. This new database G' is usually called a *repair* of G with respect to R, following some formal definition for the semantics of 'minimal difference'.

Here we consider *set repairs*, in which the notion of minimal difference is based on sets of nodes and edges. While we could provide a notion of distance between arbitrary data-graphs via an adequate definition of symmetric difference, it has been the case that the complexity of finding such repairs is quite high, so it is common to consider set repairs where one graph is obtained from the other by only adding or only deleting information [13, 18, 10]. This gives raise to subset and superset repairs, on which we focus in this work.

We say that a data-graph $G = (V, L_e, D)$ is a **subset** of a data-graph $G' = (V', L'_e, D')$ (written as $G \subseteq G'$) if and only if $V \subseteq V'$ and for all $v, v' \in V$ it happens that $L_e(v, v') \subseteq L'_e(v, v')$ and $D(v) = D'(v)$. In this case, we also say that G' is a **superset** of G.

Definition 3 (Subset and superset repairs)**.** Let R be a set of restrictions and G a data-graph. We say that G' is a **subset repair** (resp. **superset repair**) or $\subseteq$-repair (resp. $\supseteq$-repair) of G if: *(a)* $G' \models R$, *(b)* $G' \subseteq G$ (resp. $G' \supseteq G$), and *(c)* there is no data-graph G'' such that $G'' \models R$ and $G' \subset G'' \subseteq G$ (resp. $G' \supset G'' \supseteq G$). We note the set of subset (resp. superset) repairs of G with respect to R as $\subseteq$-$Rep(G, R)$ (resp. $\supseteq$-$Rep(G, R)$).

Example 4. In Example 1, by deleting the node with value Margot Robbie we obtain a $\subseteq$-repair of the graph database.

Preferences Now we introduce the two preference criteria that we will use to induce orderings on the set of repairs. In the manner done in [17] for Description Logic knowledge bases, we provide a notion of *weight* over graph databases, which can be translated into preferences via the induced ordering.

Definition 5 (Weight functions)**.** Given a function $w : \Sigma_e \sqcup \Sigma_n \to \mathbb{N}$ (where $\sqcup$ denotes disjoint union), we can extend w to any finite data-graph $G = (V, L_e, D)$ over Σ_e and Σ_n as

$$w(G) = \sum_{x,y \in V} \left(\sum_{z \in L_e(x,y)} w(z) \right) + \sum_{x \in V} w(D(x)).$$

When considering a way to select one among various possible subset or superset repairs, one approach is to consider that different edge labels and data values are prioritized differently by being assigned different weights. These weights can be aggregated to obtain a measure of the weight of a whole data-graph, and this aggregated value can then be compared for all the possible repairs to obtain a preferred repair based on the natural ordering of non-negative integers.

Definition 6 (Weight-based preferences)**.** Given a weight function w, we define that $G_1 <_w G_2$ iff $w(G_1) < w(G_2)$.

If $G_1 <_w G_2$, we say that G_1 is ***w*-preferred** to G_2 in the context of superset repairs, while we say that G_2 is ***w*-preferred** to G_1 in the case of subset repairs. We say that G is ***w*-preferred** if G is $<_w$-minimal for superset (resp. $<_w$-maximal for subset).

Example 7. Consider a context where data-graphs represent physical networks, and where edges represent two different quality levels of connection (e.g. varying robustness, resistance to physical attacks) which we call $\downarrow_{\text{LOW}}$ and $\downarrow_{\text{HIGH}}$. Let $R = \{\alpha_{connected\_dir}, \alpha_{2l \to good}\}$ be a set of restrictions, where

$$\alpha_{connected\_dir} = \_^{*}$$
$$\alpha_{2l \to good} = \downarrow_{\text{LOW}}\downarrow_{\text{LOW}} \Rrightarrow \downarrow_{\text{HIGH}}\downarrow_{\text{LOW}} \cup \downarrow_{\text{LOW}}\downarrow_{\text{HIGH}} \cup \downarrow_{\text{HIGH}}\downarrow_{\text{HIGH}} \cup \downarrow_{\text{HIGH}} \cup \downarrow_{\text{LOW}} .$$

$\alpha_{connected\_dir}$ expresses the notion of directed connectivity, and $\alpha_{2l \to good}$ establishes that if a node can be reached by two low-quality edges, then it is also possible to reach it by a 'good' path. That is, it can be reached in either only one step, or in two steps but using at least one high-quality edge.

We could consider a weight function that attempts to represent the costs of building nodes and connections in this network. For example, it could assign a uniform weight to all data values $w(x) = 20$, a low cost for low-quality connections $w(\downarrow_{\text{LOW}}) = 1$, and higher costs for high-quality connections $w(\downarrow_{\text{HIGH}}) = 5$.

Now, given a data-graph G that does not satisfy the restrictions, a w-preferred superset repair can be interpreted as the most cost-effective way of making a superset of the network that satisfies the restrictions while minimizing the costs given by w. For a full example, see Figure 2.

The weight function w (over Σ_n and Σ_e) is considered fixed in general, and it should be an 'easy' function to compute. That is, given a reasonable encoding for $\Sigma_e \sqcup \Sigma_n$, we expect that $w(x) \leq 2^{p(|x|)}$ for every $x \in \Sigma_e \sqcup \Sigma_n$ and some polynomial $p(n)$, and also assume that the result of $w(x)$ should be computable in polynomial time over $|x|$, the *size* of x. Other kinds of restrictions could be made upon w (for example, that $w(x) \leq q(|x|)$ for some polynomial $q(n)$)

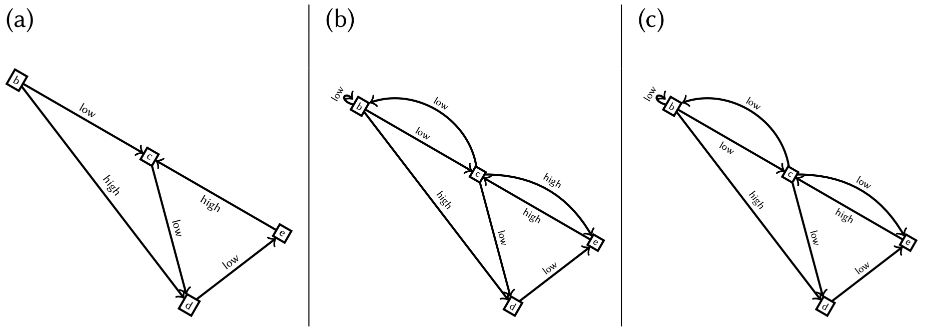

Figure 2: (a) A data-graph that does not satisfy $\alpha_{connected\_dir}$ nor $\alpha_{2l\to good}$ from Example 7: it is not connected as a directed graph, and the pair (c, e) of nodes is connected via two $\downarrow_{\text{LOW}}$ but cannot be connected via a 'good' path. (b) A possible w superset repair with respect to the example of figure (a); note that the removal of any of the new edges ends up violating R. The associated weight of this repair is the original weight plus 1 + 1 + 5 (from the added LOW and HIGH edges). (c) A w-preferred superset repair with respect to the example of figure (a). The associated extra weight of this repair is 3, and it can be proved that there is no other superset repair with a lower weight.

depending on the kind of weight function wanted to be modeled, but we note that, without any restriction, w could even be an uncomputable function.

The second type of preference criteria we study is based on *multisets*.

Definition 8 (Multisets)**.** Given a set A, its set of **finite multisets** is defined as $\mathcal{M}_{<\infty}(A) = \{M : A \to \mathbb{N} \mid M(x) \neq 0$ only for a finite number of $x\}$. Given a strict partial order $(A, <)$, the **multiset ordering** $(\mathcal{M}_{<\infty}(A), <_{mset})$ is defined as in *[19, 20]*: $M_1 <_{mset} M_2$ iff $M_1 \neq M_2$ and for all $x \in A$, if $M_1(x) > M_2(x)$, then there exists some $y \in A$ such that $x < y$ and $M_1(y) < M_2(y)$.

If $(A, <)$ is a strict partial (resp. total) order, then $(\mathcal{M}_{<\infty}(A), <_{mset})$ is a partial (resp. total) order. If $(A, <)$ is a well-founded order[1], then we have that $(\mathcal{M}_{<\infty}(A), <_{mset})$ is also a well-founded order [19].

Definition 9. Given a finite data-graph G over Σ_e and Σ_n, we define its **multiset of edges and data values** as the multiset $G^{\mathcal{M}}$ over Σ_e and Σ_n such that:

$$G^{\mathcal{M}}(x) = \begin{cases} |[\![x]\!]_G| & x \in \Sigma_e \\ |[\![x^{=}]\!]_G| & x \in \Sigma_n. \end{cases}$$

Note that all these multisets of edges and data values belong to $\mathcal{M}_{<\infty}(A)$ with $A = \Sigma_e \sqcup \Sigma_n$.

Definition 10 (Multiset-based preferences)**.** Let G_1, G_2 be two finite data-graphs over Σ_e and Σ_n, and let $<$ be a partial order defined over $A = \Sigma_e \sqcup \Sigma_n$. We say that $G_1 <_{Gmset} G_2$ if $G_1^{\mathcal{M}} <_{mset} G_2^{\mathcal{M}}$.

[1] I.e. for all $S \subseteq A$, if $S \neq \emptyset$ then there exists $m \in S$ such that $s \not< m$ for every $s \in S$.

If $G_1 <_{Gmset} G_2$, we say that G_1 is **$\mathcal{M}$-preferred** to G_2 in the context of superset repairs, while we say that G_2 is **$\mathcal{M}$-preferred** to G_1 in the case of subset repairs. We say that G is **$\mathcal{M}$-preferred** if G is $<_{\mathcal{M}}$-minimal for superset (resp. $<_{\mathcal{M}}$-maximal for subset).

Example 11. Consider the data-graphs from Figure 2. Ignoring any possible data value, observe that the multisets corresponding to graphs b) and c) are (with the informal multiset notation): $\{\text{LOW}, \text{LOW}, \text{LOW}, \text{LOW}, \text{LOW}, \text{HIGH}, \text{HIGH}, \text{HIGH}\}$ and $\{\text{LOW}, \text{LOW}, \text{LOW}, \text{LOW}, \text{LOW}, \text{LOW}, \text{HIGH}, \text{HIGH}\}$, respectively. Assuming $\text{LOW} > \text{HIGH}$, then in this case data-graph b) is $\mathcal{M}$-preferred to c).

3. Preferred repairs

In this section we consider subset and superset repairs and Ω-preferred criteria where $\Omega \in \{w, \mathcal{M}\}$, using different subsets of Reg-GXPath for $\mathcal{L}$.

We will always consider the weight function $w : \Sigma_e \sqcup \Sigma_n \to \mathbb{N}$ fixed and efficiently computable: given a codification of $x \in \Sigma_e \sqcup \Sigma_n$ of size n the value $w(x)$ is computable in $poly(n)$. This implies that, for any data-graph G, $w(G)$ is also computable in $poly(|G|)$.

In the same manner, when considering multiset-preferred repairs, we will assume that the order $<$ defined over $\Sigma_e \sqcup \Sigma_n$ can be computed efficiently: given $x, y \in \Sigma_e \sqcup \Sigma_n$ it is possible to decide if $x < y$ in polynomial time on the representation on both x and y. Thus, we can decide whether $G_1 <_{Gmset} G_2$ in $poly(|G_1| + |G_2|)$. Furthermore, we assume that the order $<$ is well founded. This implies that there are no infinite descending chains $c_1 > c_2 > \dots$

Note that the notions of weight-based and multiset-based preferences induce an ordering over finite data-graphs, which does not admit infinite descending chains. Hence, if a superset repair exists, then there is also a (weight-based or multiset-based) preferred repair. On the other hand, the number of subsets of a given data-graph is finite, so if a repair exists, there must necessarily be a preferred repair.

The complexity of the problems when the set of expressions R is fixed is commonly denominated *data complexity*. Most lower bounds we derive apply to this case.

3.1. Preferred Subset Repairs

For the case of $\subseteq$-repairs, it was proved in [21] that deciding whether there exists a non-trivial repair (i.e. different from the $\emptyset$ data-graph) is an NP-COMPLETE problem for a fixed set R of Reg-GXPath^{pos} expressions, and that the problem is tractable if we only allow node expressions from Reg-GXPath^{pos} as $\mathcal{L}$. Observe that:

Proposition 12. *The problem of deciding whether G has a non-trivial $\subseteq$-repair with respect to R can be reduced to the problem of deciding whether G has a non-trivial Ω-preferred $\subseteq$-repair with respect to R.*

Given a fixed weight function w (resp. an ordering $<$), the existence of a $\subseteq$-repair G' of G with respect to R implies the existence of a preferred $\subseteq$-repair for G (and vice versa). Therefore, it follows directly from Proposition 12 and the aforementioned results in [21] that:

Theorem 13. *The problem of deciding whether there exists a non-trivial Ω-preferred $\subseteq$-repair for a given data-graph G and a set of expressions R is NP-COMPLETE for a fixed set of Reg-GXPath^{pos} path expressions.*

When $\mathcal{L} \subseteq$ Reg-GXPath$^{pos}_{node}$, a subset repair can be computed in polynomial time and, furthermore, it is unique [21]. Therefore, it must be the preferred one:

Theorem 14. *Given a data-graph G, a set of Reg-GXPath$^{pos}_{node}$ expressions R, and a preference criteria Ω, there exists an algorithm that computes the Ω-preferred $\subseteq$-repair of G with respect to R.*

3.2. Preferred Superset Repairs

There is a restriction set R containing only Reg-GXPath$^{pos}_{node}$ expressions such that computing w-preferred $\supseteq$-repair is already intractable:

Theorem 15. *Given a data-graph G, a set of Reg-GXPath$^{pos}_{node}$ expressions R and a natural number K, let Π_w be the problem of deciding whether there exists a w-preferred $\supseteq$-repair of G with respect to R whose weight is bounded by K. Then, there exists a set of positive node expressions R and a weight function w such that the problem is NP-COMPLETE.*

Notice that we could have equivalently defined Π_w as the problem of deciding whether there exists some data-graph $G' \supseteq G$ such that $G' \models R$ and $w(G') \leq K$ (i.e., requiring minimality is not necessary).

Proof. The problem is in NP in general: if there exists a repair of G with respect to R, due to [21, Theorem 24], then there is one in 'standard form'. The size of this repair is bounded by $poly(|G| + |R|)$, and by inspecting the proof of the theorem and considering that w is a non-negative function, it can be shown that the preferred repair always has this 'standard form'. Then, a positive certificate consists of a data-graph G' such that $|G'| \leq poly(|G| + |R|)$, $G' \models R$, $G \subseteq G'$ and $w(G') \leq K$.

For the hardness, we reduce 3-SAT to Π_w, with R fixed. For the reduction, we consider $\Sigma_e \supseteq \{\text{VALUE\_OF}, \text{APPEARS\_IN}, \text{APPEARS\_NEGATED\_IN}\}$ and $\Sigma_n \supseteq \{clause, var, \top, \bot\}$. We define the weight function w as $w(c) = 2$ for $c \in \Sigma_n \cup \{\text{APPEARS\_IN}, \text{APPEARS\_NEGATED\_IN}\}$ and $w(\text{VALUE\_OF}) = 1$.

Given a 3-CNF formula ϕ with n variables $x_1, \ldots, x_n$ and m clauses $c_1, \ldots, c_m$ we build a data-graph G, a set of Reg-GXPath$^{pos}_{node}$ node expressions R and define a number K such that ϕ is satisfiable if and only if G has a superset repair G' with respect to R such that $w(G') \leq K$.

We define the graph as $G = (V_G, L_G, D_G)$ where:

$$V_G = \{x_i \mid 1 \leq i \leq n\} \cup \{c_j \mid 1 \leq j \leq m\} \cup \{\bot, \top\}$$

$$L_G(x_i, c_j) = \begin{cases} \{\text{APPEARS\_IN}, \text{APPEARS\_NEGATED\_IN}\} \text{ if } x_i \text{ and } \neg x_i \text{ appear in } c_j \\ \{\text{APPEARS\_IN}\} \text{ if only } x_i \text{ appears in } c_j \\ \{\text{APPEARS\_NEGATED\_IN}\} \text{ if only } \neg x_i \text{ appears in } c_j \\ \emptyset \text{ otherwise} \end{cases}$$

$$L_G(v, w) = \emptyset \text{ for every other pair } v, w \in V_G$$

$$D_G(x) = x \text{ for } x \in \{\bot, \top\}$$
$$D_G(x_i) = var \text{ for } 1 \leq i \leq n$$
$$D_G(c_j) = clause \text{ for } 1 \leq j \leq m.$$

The structure of ϕ is codified in the APPEARS edges. We define $K = w(G) + n$, and we want any superset repair of G with respect to R with weight K to codify an assignment of the 'node' variables by using the edges VALUE_OF. In order to do this we define the Reg-GXPath$^{pos}_{node}$ expressions

$$\psi_1 = \langle [var^{\neq}] \cup \downarrow_{\textsc{value\_of}} [\top] \cup \downarrow_{\textsc{value\_of}} [\bot] \rangle$$
$$\psi_2 = \langle [clause^{\neq}] \cup \downarrow^{-}_{\textsc{appears\_in}} \downarrow_{\textsc{value\_of}} [\top] \cup \downarrow^{-}_{\textsc{appears\_negated\_in}} \downarrow_{\textsc{value\_of}} [\bot] \rangle.$$

The expression ψ_1 forces every variable node to have a VALUE edge directed to a boolean node, while the expression ψ_2 forces every clause to be 'satisfied' in any repair of G. Therefore, we define $R = \{\psi_1, \psi_2\}$. Now we show that ϕ is satisfiable if and only if G has a superset repair with respect to R with weight at most $w(G) + n$.

$\Longrightarrow$) Let f be a valuation on the variables of ϕ that evaluates ϕ to true. We then 'add' edges to G in the following way: if $f(x_i) = \top$ we add the edge $(x_i, \textsc{value\_of}, \top)$, and otherwise we add $(x_i, \textsc{value\_of}, \bot)$. This graph satisfies both expressions from R and has cost $w(G) + n$, since $w(\textsc{value\_of}) = 1$.

$\Longleftarrow$) Let G' be a superset repair of G with respect to R with weight at most $w(G) + n$. Since every superset repair of G has to add at least n VALUE edges and they cost 1 unit each we know that G' has to be exactly the original graph G plus n VALUE edges (one for each node variable). Then we can define a valuation of the variables of ϕ by using this edges: if the edge $(x_i, \textsc{value\_of}, \top)$ is present in G' we define $f(x_i) = \top$, and otherwise $f(x_i) = \bot$. Since ψ_2 is satisfied in G' this assignment must satisfy ϕ. □ □

The hardness of Π_w implies that, unless P = NP there is no algorithm to compute w-preferred $\supseteq$-repairs for fixed sets of Reg-GXPath$^{pos}_{node}$ expressions.

Furthermore, computing a multiset preferred repair is also a hard problem for simple Reg-GXPath$^{pos}_{node}$ expressions:

Theorem 16. *Given a data-graph G, a set of Reg-GXPath$^{pos}_{node}$ expressions R, an edge label $\textsc{a} \in \Sigma_e$, and a natural number K, let $\Pi_{\mathcal{M}}$ be the problem of deciding whether G has a $\mathcal{M}$-preferred $\supseteq$-repair G' with respect to R such that G' has at most K edges with label A. Then, there exists a set of positive node expressions R and a well-ordering $<$ such that the problem is NP-COMPLETE.*

Proof. The proof is analogous to the one of Theorem 15, considering the order $\textsc{value\_of} < \textsc{appears\_in} < \textsc{appears\_negated\_in} < clause < var < \top < \bot$ and $K = n$. □ □

It follows that the hardness of $\Pi_{\mathcal{M}}$ implies the hardness of computing $\mathcal{M}$-preferred $\supseteq$-repairs for fixed sets of Reg-GXPath$^{pos}_{node}$ expressions.

Remark 17. In the case of $\supseteq$-repairs, it was proved in [21] that deciding whether there exists at least one repair is an undecidable problem for a fixed set of Reg-GXPath expressions, and NP-HARD if Reg-GXPath$^{pos} \subseteq \mathcal{L}$. Meanwhile, it was shown that if $\mathcal{L} \subseteq$ Reg-GXPath^{pos} and the restriction set is fixed, or rather if $\mathcal{L} \subseteq$ Reg-GXPath$^{pos}_{node}$, then there are polynomial-time algorithms to compute a superset repair. Theorems 15 and 16 show that these tractable cases become intractable when considering preferred repairs.

Looking at the proof of Theorem 15 it is clear that assigning a non zero weight to the edges is necessary to achieve the hardness result. If we assume that $w(\textsc{a}) = 0$ for all $\textsc{a} \in \Sigma_e$ we can show that:

Theorem 18. *There exists an algorithm that given a data-graph G and a set of Reg-GXPath$^{pos}_{node}$ expressions R computes a w-preferred $\supseteq$-repair in polynomial time whenever w satisfies that $w(\textsc{a}) = 0$ for all $\textsc{a} \in \Sigma_e$.*

Proof. The standard form described in [21, Theorem 24] minimizes the number of nodes added, and the number of standard forms is bounded by $2^{|R|}$. Therefore, if R is fixed, it is possible to search for the one with smallest weight in polynomial time. □

If $R \subseteq$ Reg-GXPath$^{pos}_{node}$ then it is actually possible to compute a $\supseteq$ repair even in combined complexity (when R is part of the input). Nonetheless, it is not possible to compute a w-preferred one, even when w ignores edge labels:

Theorem 19. *There is a fixed w function that assigns 0 cost to all edge labels such that the problem Π_w (see Theorem 15) is NP-complete in combined complexity when restricting that $R \subseteq$ Reg-GXPath$^{pos}_{node}$.*

Proof. We reduce 3-SAT to our problem. Given a formula ϕ on n variables $x_1, \dots x_n$ and m clauses $c_1, \dots, c_m$ we will construct a data-graph G and a set of Reg-GXPath$^{pos}_{node}$ expressions R such that ϕ is satisfiable if and only if G has a $\supseteq$-repair G' with respect to R such that $w(G') \leq w(G) + n$. We denote by $l_{j,k}$ the kth literal of c_j, for $1 \leq j \leq m$ and $1 \leq k \leq 3$. For example, if $c_3 = (x_1 \vee x_3 \vee \neg x_4)$ then $l_{3,1} = x_1$ and $l_{3,3} = \neg x_4$. The fixed weight function w is defined as $w(d) = 1$ for $d \in \Sigma_n$, and $w(d) = 0$ otherwise.

Let $\Sigma_e \supseteq \{\textsc{down}\}$ and $\Sigma_n \supseteq \{x_i : i \in \mathbb{N}\} \cup \{\neg x_i : i \in \mathbb{N}\} \cup \{c_j : j \in \mathbb{N}\}$, and let us define $G = (V, L, D)$ as:

$$\begin{aligned} V &= \{c_j : 1 \leq j \leq m\} \\ L(v, w) &= \emptyset \text{ for } v, w \in V \\ D(c_j) &= c_j \text{ for } 1 \leq j \leq m \end{aligned}$$

We also define $R = \{\varphi_j : 1 \leq j \leq m\} \cup \{\psi_i : 1 \leq i \leq n\}$ where

$$\varphi_j = c_j^{=} \vee \langle \downarrow_{\textsc{down}} [\bigvee_{k=1}^{3} l_{j,k}^{=}] \rangle$$

$$\psi_i = \langle \downarrow_{\textsc{down}} [x_i^{=} \vee \neg x_i^{=}] \rangle$$

The formula φ_j ensures that in any repair there is a node with a data value related to a literal that satisfies c_j. The formula ψ_i ensures that in every repair there is either a node with data value x_i or $\neg x_i$.

We now show that ϕ is satisfiable if and only if G has a $\supseteq$-repair with respect to R bounded by $w(G) + n$:

$\Longrightarrow$) If ϕ is satisfiable then there is a valuation of its variables $f : \{x_i : 1 \leq i \leq n\} \to \{\top, \bot\}$ that satisfies every clause. If we add to G the set of nodes $\{f(x_i) : 1 \leq i \leq n\}$ with data values $D(f(x_i)) = x_i$ if $f(x_i) = \top$ and $D(f(x_i)) = \neg x_i$ otherwise, and every possible edge, then the obtained data-graph satisfies R and has weight $w(G) + n$.

$\Longleftarrow$) Let G' be a $\supseteq$-repair of G with respect to R such that $w(G') \leq w(G) + n$. Observe that in order to satisfy each ψ_i G' must have a node with data value x_i or $\neg x_i$. Since the weight is bounded by $w(G) + n$ we can conclude that there is either a node with data value x_i or one with data value $\neg x_i$, but not both. We can define a valuation f on the variables $x_1, \ldots, x_n$ as $f(x_i) = \top$ if and only if there is a node v in G' such that $D(v) = x_i$. Such valuation satisfies ϕ because all the φ_j are satisfied in G'.

□

The approach of restricting the priority criteria to only consider data values can be carried on in the same way for the $\mathcal{M}$-criteria, by asking only for an order over Σ_n rather than both Σ_n and Σ_e. Then, it is possible to prove theorems analogous to both Theorems 18 and 19.

4. Conclusions

In this work, we analyze preferred repairing for data-graphs. We specifically focus on the problem of deciding whether a data-graph G has a non-trivial preferred repair under two different data-aware preference criteria, one based on weights and the other based on multiset orderings. We showed that in some cases, these criteria do not make the repair decision problem harder than the version lacking preferences.

Some questions in this context remain open, such as that of finding refined tractable versions of the problem that might be based on real-world applications. Alternative definitions of types of repairs for data-graphs [13], like those based on symmetric-difference [10], are worth studying in the preference-based setting. It would also be interesting to study more general families of criteria, such as the ones proposed in [15], and analyze whether the complexity of the decision problems changes for data-graphs.

In the definition of preferred repairs presented in this work, we never discuss how to handle scenarios with multiple solutions. While introducing a notion of preference reduces the set of repairs of interest, this does not address the problem of choosing one in the presence of several options. In practice, the preference criterion should be adjusted based on the general use case, in order to reduce the set of obtained repairs to a size that is acceptable in expectation. Still, to avoid choosing one repair over another, one could ask for the information that is contained in every possible repair: this is precisely the problem of *consistent query answering*, whose complexity remains open in this context. However, based on the results presented in this work, some observations can already be made: for example, in the case of subset repairs considering positive node expressions as integrity constraints, we proved that there is a unique preferred repair computable in polynomial time, which therefore implies that the CQA problem can be solved efficiently as well.

References

[1] W. Fan, Graph pattern matching revised for social network analysis, in: Proceedings of the 15th International Conference on Database Theory, 2012, pp. 8–21.

[2] M. K. Anand, S. Bowers, B. Ludäscher, Techniques for efficiently querying scientific workflow provenance graphs., in: EDBT, volume 10, 2010, pp. 287–298.

[3] M. Arenas, J. Pérez, Querying semantic web data with sparql, in: Proceedings of the thirtieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, 2011, pp. 305–316.

[4] P. Barceló Baeza, Querying graph databases, in: Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGAI symposium on Principles of database systems, ACM, 2013, pp. 175–188.

[5] P. Barceló, J. Pérez, J. L. Reutter, Relative expressiveness of nested regular expressions., AMW 12 (2012) 180–195.

[6] L. Libkin, W. Martens, D. Vrgoč, Querying graphs with data, Journal of the ACM (JACM) 63 (2016) 1–53.

[7] S. Abiteboul, V. Vianu, Regular path queries with constraints, Journal of Computer and System Sciences 58 (1999) 428–452.

[8] P. Buneman, W. Fan, S. Weinstein, Path constraints in semistructured databases, Journal of Computer and System Sciences 61 (2000) 146–193.

[9] M. Arenas, L. Bertossi, J. Chomicki, Consistent query answers in inconsistent databases, in: PODS, volume 99, Citeseer, 1999, pp. 68–79.

[10] B. ten Cate, G. Fontaine, P. G. Kolaitis, On the data complexity of consistent query answering, in: Proceedings of the 15th International Conference on Database Theory, ICDT '12, 2012, pp. 22–33.

[11] J. Wijsen, Condensed representation of database repairs for consistent query answering, in: Proceedings of the 9th International Conference on Database Theory, ICDT '03, Springer-Verlag, 2002, pp. 378–393.

[12] A. Lopatenko, L. Bertossi, Complexity of consistent query answering in databases under cardinality-based and incremental repair semantics, in: In ICDT, Springer, 2007, pp. 179–193.

[13] P. Barceló, G. Fontaine, On the data complexity of consistent query answering over graph databases, Journal of Computer and System Sciences 88 (2017) 164–194.

[14] S. Flesca, F. Furfaro, F. Parisi, Preferred database repairs under aggregate constraints, in: International Conference on Scalable Uncertainty Management, Springer, 2007, pp. 215–229.

[15] S. Staworko, J. Chomicki, J. Marcinkowski, Prioritized repairing and consistent query answering in relational databases, Annals of Mathematics and Artificial Intelligence 64 (2012) 209–246.

[16] G. Brewka, Preferred subtheories: An extended logical framework for default reasoning, in: N. S. Sridharan (Ed.), Proceedings of the 11th International Joint Conference on Artificial Intelligence. Detroit, MI, USA, August 1989, Morgan Kaufmann, 1989, pp. 1043–1048. URL: http://ijcai.org/Proceedings/89-2/Papers/031.pdf.

[17] M. Bienvenu, C. Bourgaux, F. Goasdoué, Querying inconsistent description logic knowledge bases under preferred repair semantics, in: Proceedings of the AAAI Conference on

Artificial Intelligence, volume 28, 2014.

[18] T. Lukasiewicz, M. V. Martinez, G. I. Simari, Complexity of inconsistency-tolerant query answering in datalog+/−, in: OTM Confederated International Conferences On the Move to Meaningful Internet Systems, Springer, 2013, pp. 488–500.

[19] N. Dershowitz, Z. Manna, Proving termination with multiset orderings, Communications of the ACM 22(8) (August 1979) 465–476.

[20] G. Huet, D. C. Oppen, Equations and rewrite rules: A survey, in: Formal Language Theory, Elsevier, 1980, pp. 349–405.

[21] S. Abriola, S. Cifuentes, M. V. Martínez, N. Pardal, E. Pin, On the complexity of finding set repairs for data-graphs, Journal of Artificial Intelligence Reasoning 76 (2023) 721–759. URL: https://arxiv.org/abs/2206.07504. doi:10.48550/ARXIV.2206.07504.