Cell-Based Causality for Data Repairs

Maxime Debosschere Floris Geerts
Universiteit Antwerpen, Belgium
{maxime.debosschere,floris.geerts}@uantwerpen.be

Abstract

In recent work, Salimi and Bertossi provide a tight connection between causality and tuple-based data repairs. We investigate this connection between causality and two other kinds of repair models. First, we consider cell-based V-repairs, i.e., repairs that are obtained by modifying cells in the data. In contrast, tuple-based repairs only allow for the deletion of tuples. Second, we introduce a new notion of repairs, called chase repairs, that take into account the procedural (chase) steps that lead to a repair. We establish a connection between causes (and the associated notion of responsibility) and V-repairs, and analyse the complexity of verifying whether a cell is a cause and whether its responsibility is above a certain threshold. Our understanding of chase repairs is still very preliminary, and we argue that provenance models that are specifically targeted to data repairs and data quality in general are needed to make formal connections between causality and chase repairs.

Categories and Subject Descriptors H.2.4. [*Database Management*]: Systems—Relational databases

General Terms Algorithms, Theory

Keywords Data Repair, Causality, Data Quality

1. Introduction

Causality is related to provenance, yet it is a more refined notion that can provide reasons and explanations for wrong or surprising results, by ranking provenance based on the notion of responsibility. It has been discussed in relation to data and workflow provenance. We refer to Meliou et al. [5, 6] for a comprehensive survey of related work in this context.

Inspired by the work by Salimi and Bertossi [7], we explore connections between causality and database repairs. Database repairs are important when consistent query answering is concerned [1] and for data cleaning [2]. In [7], it is shown how to obtain database repairs from causes, and the other way around. Furthermore, a strong connection is unveiled between computing causes and their responsibilities for conjunctive queries, on the one hand, and computing both subset-based and cardinality-based repairs in databases with regard to denial constraints, on the other hand.

In the first part of this paper, we extend these connections from tuple-based repairs, such as the subset-based and cardinality-based repairs considered in [7] to cell-based repairs, which are often used in practice [2]. We limit ourselves to functional dependencies as constraint language. Our results tell that the correspondences from [7] carry over to our setting. More precisely, one can associate cell-based repairs to causes, and vice versa. Furthermore, we provide a PTIME algorithm for determining whether a cell in the database is a *cause for conflict* with regard to functional dependencies. The algorithm relies on a cell-based notion of lineage. By contrast, we show that determining whether the responsibility of a cause exceeds a certain threshold is NP-hard. Finding a matching upper bound is left as future work.

In the second part of the paper, we identify some shortcomings of standard repair models and propose a new kind of model, referred to as chase repairs. Intuitively, chase repairs take into account the procedural steps to obtain a repair, starting from curated information. Causes in this context have a different meaning: they correspond to the cells in the database that need to be curated in order to successfully obtain a repair. In other words, rather than corresponding to conflicts, causes have a "positive" interpretation. They identify positions in the database on which the domain expert should focus to establish the *correctness of the values*. Once this is done, these positions can be regarded as carrying "true" and curated values, and a repair is guaranteed to exist. More importantly, during the repair process only curated information is used, ensuring to some extent the validity of the repair.

A full investigation of chase repairs is left as future work. One possible way of establishing this connection is by relating causes to some kind of lineage, just like in the first part of the paper. However, to our knowledge, no lineage/provenance model is yet in place that is expressive enough to model procedural repair steps. The development of such models is of interest on its own. Indeed, it would allow for a formal underpinning of the common belief that provenance is useful for improving the quality of data.

2. Cell-based repairs

We first formalise the notion of cell-based data repairs. In this paper, we consider database instances I over a single relation $R(A_1, \ldots, A_m)$, where $A_1, \ldots, A_m$ are attributes, each equipped with a domain $\mathsf{dom}(A_i)$. As usual, an instance I of R consists of a finite set of tuples; a tuple simply being an element of $\mathsf{dom}(A_1) \times \cdots \times \mathsf{dom}(A_m)$. A functional dependency (FD) is a constraint of the form $\varphi : X \rightarrow A$, where X is a set of attributes in R and A is an attribute in R. An instance I satisfies an FD $\varphi : X \rightarrow A$, denoted by $I \models \varphi$, if for any two tuples $s, t \in I$ such that $s[X] = t[X]$, it holds also that $s[A] = t[A]$. An instance I satisfies a set Σ of FDs, denoted by $I \models \Sigma$, if $I \models \varphi$ for each $\varphi \in \Sigma$. When $I \models \Sigma$ we also call I clean or consistent with regard to Σ; otherwise I is called dirty or inconsistent.

A position p in I refers to a single database "cell" and is given by a pair $\langle t, A\rangle$, where t stands for a particular tuple and A is a single attribute. The value of $p = \langle t, A\rangle$ in I, denoted by $\mathsf{val}(I, p)$, is $t[A]$. The set of all positions in an instance I is denoted by $\mathsf{Pos}(I)$.

TaPP 2015, July 8–9, 2015, Edinburgh, Scotland.

	FN	LN	City	Country	CC	AC	Phone
t_1	John	Lewis	**New York**	**USA**	**01**	212	509 3883
t_2	David	Smith	**New York**	**UK**	**44**	212	509 7639
t_3	Lucas	Reed	**New York**	**UK**	**01**	212	509 8291
t_4	Luke	White	**London**	**UK**	**01**	020	9876 0402
t_5	Jane	Taylor	**London**	**UK**	**43**	020	9876 3266
t_6	Mark	Hill	**London**	**UK**	**44**	020	9876 2911
t_7	Ron	Clark	**London**	**UK**	**44**	020	9876 1059
t_8	Will	Green	Toronto	Canada	01	416	511 7145
t_9	Lisa	Jones	Toronto	Canada	01	416	511 5200

(a)

City	Country	CC
New York	USA	01
London	UK	44
New York	***USA***	01
London	UK	***44***
Vienna	***Austria***	43
London	UK	44
London	UK	44
Toronto	Canada	01
Toronto	Canada	01

(b)

City	Country	CC
New York	USA	01
London	UK	44
New York	***USA***	01
London	UK	***44***
London	UK	***44***
London	UK	44
London	UK	44
Toronto	Canada	01
Toronto	Canada	01

(c)

$\Sigma = \{\varphi_1 : \texttt{City} \rightarrow \texttt{Country}, \varphi_2 : \texttt{Country} \rightarrow \texttt{CC}\}$

Figure 1. (a) Example of a database instance I with a set of functional dependencies Σ. Bold values belong to positions that are causes. (b) A possible SV-repair. (c) The only possible CV-repair.

In analogy to [4], we represent possible value modifications in I by means of V-instances. The difference between a V-instance and a "normal" instance is that tuples may contain variables. More specifically, if $P \subseteq \mathsf{Pos}(I)$ then we denote by I_P the instance I in which the value of every position $p = \langle t, A\rangle$ in P is replaced by a unique variable v_p, and I_P agrees with I on all other positions not in P. Intuitively, the positions in P indicate those positions in I to which value modifications will be applied.

Which values that these positions take is formalised by means of valuations. The domain of each variable v_p in I_P, $\mathsf{dom}(v_p)$, is the domain of the attribute corresponding to position p. A valuation ν of I_P is an assignment of the variables v_p to constants in their respective domains. By extending such valuation to the identity on constants, we denote by $\nu(I_P)$ the (standard) database instance consisting of tuples $\nu(t) = (\nu(t[A_1]), \ldots, \nu(t[A_m]))$ for $t \in I$.

Definition 1 (V-repair)**.** Given an instance I, set Σ of FDs, and a set $P \subseteq \mathsf{Pos}(I)$, we say that the V-instance I_P is a *V-repair for* (I, Σ) if there exists a valuation ν of I_P such that $\nu(I_P) \models \Sigma$. □

In other words, in case that $I \not\models \Sigma$, it suffices to change values in the positions in P to obtain a clean instance $\nu(I_P)$, *i.e.,* $\nu(I_P) \models \Sigma$.

We next identify two special classes of V-repairs corresponding to the standard notions of subset-repairs (S-repairs) and cardinality-repairs (C-repairs) [1]. Recall that both of these repairs remove tuples from inconsistent instances to make them consistent: a subset-repair removes a minimal subset of tuples to achieve consistency, whereas a cardinality-repair removes the smallest number of tuples overall. Clearly, every cardinality-repair is also a subset-repair. Inspired by this, we define the following:

Definition 2 (SV-repair)**.** A subset V-repair for (I, Σ) is a V-repair I_P for (I, Σ) such that none of the V-instances $I_{P'}$ for $P' \subsetneq P$ is a V-repair for (I, Σ). □

Definition 3 (CV-repair)**.** A cardinality V-repair for (I, Σ) is a V-repair I_P for (I, Σ) such that there does not exist a V-repair $I_{P'}$ with $|P'| < |P|$. □

An alternative point of view of SV- and CV-repairs is given as follows. Define the modification $\mathfrak{M}(I, I')$ between two instances I and I' as those positions $p \in \mathsf{Pos}(I)$ such that $val(I, p) \neq val(I', p)$. The modification distance $\Delta(I, I')$ between two instances I and I' is defined as $|\mathfrak{M}(I, I')|$. Then, an SV-repair I_P of (I, Σ) corresponds to an instance $I' \models \Sigma$ (by means of a valuation) such that there is no instance $I'' \models \Sigma$ for which $\mathfrak{M}(I, I'') \subsetneq \mathfrak{M}(I, I')$. Similarly, a CV-repair I_P of (I, Σ) corresponds to an instance $I' \models \Sigma$ such that there is no instance $I'' \models \Sigma$ for which $\Delta(I, I'') < \Delta(I, I')$.

Example 1. Consider the database instance I and the set of functional dependencies Σ given in Figure 1a. For every employee, the instance stores the first name (FN), last name (LN), city of residence (City), country in which that city is located (Country), country code (CC), area code (AC), and phone number (Phone). The set of functional dependencies is $\Sigma = \{\varphi_1 : \texttt{City} \rightarrow \texttt{Country}, \varphi_2 : \texttt{Country} \rightarrow \texttt{CC}\}$. An example of an SV-repair I_P for (I, Σ) is obtained by choosing positions $P = \{\langle t_2, \texttt{City}\rangle, \langle t_3, \texttt{Country}\rangle, \langle t_4, \texttt{CC}\rangle, \langle t_5, \texttt{City}\rangle, \langle t_5, \texttt{Country}\rangle\}$. Indeed, a valuation ν of these positions such that $\nu(I_P) \models \Sigma$ is depicted in Figure 1b. It can be readily verified that no subset $P' \subsetneq P$ of positions can lead to a V-repair $I_{P'}$ of (I, Σ). However, I_P is not a CV-repair. Indeed, consider positions $P' = \{\langle t_2, \texttt{City}\rangle, \langle t_3, \texttt{Country}\rangle, \langle t_4, \texttt{CC}\rangle, \langle t_5, \texttt{CC}\rangle\}$. Again, $I_{P'}$ is a V-repair. Figure 1c shows a valuation ν' such that $\nu'(I_{P'}) \models \Sigma$. Observe that $4 = |P'| < |P| = 5$. It can be easily verified that there exists no V-repair in which less than four positions are modified. Hence, $I_{P'}$ is a CV-repair. In fact, it is the only possible CV-repair for (I, Σ). □

The notion of V-repairs was studied by Kolahi and Lakshmanan [4]. In particular, they established the intractability of the V-REPAIR EXISTENCE PROBLEM. This problem is to determine, given an instance I, set Σ of FDs and integer value k, whether there exists a V-repair I_P for (I, Σ) such that $|P| < k$. They show that the V-REPAIR EXISTENCE PROBLEM is NP-complete even in the simple setting where Σ consists of two fixed unary FDs.

3. Causality and cell-based repairs

We next turn our attention to causality. More specifically, we want to identify the causes for inconsistencies in dirty database instances and rank these causes in terms of their responsibility. Causality in databases was defined on the level of tuples [5] and has been used in the context of tuple-based repairs by Salimi and Bertossi [7]. We revisit the notion of causality on the level of individual cells (positions). More specifically, consider a database instance I and a set of functional dependencies Σ.

Definition 4 (Causality)**.** A position $p \in \mathsf{Pos}(I)$ is a *cause* for (I, Σ) if there exists at least one set $\Gamma \subset \mathsf{Pos}(I)$, called a *contingency set* for p, such that $I_{\Gamma \cup p}$ is a V-repair but I_Γ is not a V-repair.

In other words, p is a cause when modifying I only in positions taken from Γ does not suffice to repair the data, but when p is also modified, a repair can be obtained.

For a cause p, all contingency sets from p with the lowest cardinality are called *minimal* contingency sets (Γ^m). The responsibility of a cause is then defined, following [5], as follows:

Definition 5 (Degree of responsibility)**.** Position p has a *degree of responsibility* $\rho(p)$ of $\frac{1}{|\Gamma^m|+1}$.

By convention, positions p that are not causes have a degree of responsibility $\rho(p) = 0$.

Example 2. Continuing with the previous example, we have displayed all causes for (I, Σ) in bold in Figure 1a. Indeed, for every of these positions p, a contingency set $\Gamma \subset \mathsf{Pos}(I)$ can be found for which $I_{\Gamma \cup p}$ is a V-repair but I_Γ is not a V-repair. For example, consider the cause $p = \langle t_5, \texttt{City} \rangle$ for (I, Σ). Its minimal contingency set is $\Gamma = \{\langle t_2, \texttt{City} \rangle, \langle t_3, \texttt{Country} \rangle, \langle t_4, \texttt{CC} \rangle, \langle t_5, \texttt{Country} \rangle\}$. Hence p has a degree of responsibility of $\frac{1}{|\Gamma|+1} = 0.20$. Note that $I_{\Gamma \cup p}$ is equal to the V-repair I_P given in Example 1. The degree of responsibility may differ among causes. To illustrate this, consider the cause $p' = \langle t_3, \texttt{Country} \rangle$ for (I, Σ). Its minimal contingency set is $\Gamma' = \{\langle t_2, \texttt{City} \rangle, \langle t_4, \texttt{CC} \rangle, \langle t_5, \texttt{CC} \rangle\}$, yielding a responsibility of 0.25. Since the degree of responsibility of p' is higher than that of p, it can be derived from this that position p' has a greater involvement than p in the conflicts from (I, Σ). In this example, both p and p' have only one minimal contingency set, but it is possible for positions to have multiple such sets. □

Given the revised notion of a cause, we next revisit the following two problems (taken from [5]):

Causality problem: Given I and Σ, compute all causes for (I, Σ).

Responsibility problem: Given I, Σ, cause p for (I, Σ) and rational number r, decide whether $\rho(p) > r$.

More specifically, we are interested in the *data complexity* of these problems, where Σ is fixed and the complexity is a function of the size of the database instance I.

3.1 The causality problem

We show that the causality problem, for causes as given in Definition 4, is in PTIME. First, we recall what is known in the tuple-based repair setting. Note that Meliou et al. [5] also provide a PTIME algorithm for the causality problem, however, causality is there defined in terms of tuple deletions and relative to conjunctive queries. More specifically, a tuple t is said to be a cause for instance I and a boolean conjunctive query q if there exists a set of tuples $\Gamma_t \subseteq I$ such that $(I \setminus \Gamma_t) \models q$, *i.e., q* returns true when evaluated on $I \setminus \Gamma_t$, but $(I \setminus (\Gamma_t \cup \{t\})) \not\models q$. Salimi and Bertossi [7] observe that this notion can be used when tuple-based repairs are concerned, and this for general classes of constraints (denial constraints).

More specifically, rephrased in the context of FDs, Salimi and Bertossi [7] observe that one can associate with a set Σ of FDs a union of boolean conjunctive queries $q_\Sigma = \cup_{\varphi \in \Sigma} q_\varphi$, where for $\varphi : X \to A$ in Σ, $q_\varphi = \exists \bar{x}, \bar{y}, \bar{y}', x_A, x'_A \, (R(\bar{x}, x_A, \bar{y}) \wedge R(\bar{x}, x'_A, \bar{y}') \wedge x_A \neq x'_A)$. Clearly, $I \models q_\Sigma$ iff $I \not\models \Sigma$. Causes are then defined relative to I and q_Σ and are closely related to subset and cardinality repairs. Furthermore, it is shown in [7] that the PTIME algorithm for the causality problem given in [5] can be extended from a single boolean conjunctive query to a union of such queries, which is required to deal with q_Σ.

Example 3. Recall the set of functional dependencies Σ given in Figure 1a, i.e., $\Sigma = \{\varphi_1 : \texttt{City} \to \texttt{Country}, \varphi_2 : \texttt{Country} \to \texttt{CC}\}$. The conjunctive query $q_{\varphi_1} = \exists$ *ci, ctr, fn, ln, cc, ac, p, ctr', fn', ln',cc', ac', p'* $\big(R$*(fn, ln, ci, ctr, cc, ac, p)* $\wedge R$*(fn', ln', ci, ctr', cc', ac', p')* $\wedge$ *ctr* $\neq$ *ctr'*$\big)$ evaluates to true on instance I iff a violation of the FD φ_1 exists in I. Similarly for q_{φ_2}. Finally, the union of these boolean queries $q_\Sigma = q_{\varphi_1} \cup q_{\varphi_2}$ allows for detecting whether or not Σ is violated on an instance. □

In the context of FDs, the PTIME algorithm for the causality problem works by computing the *lineage* of q_Σ relative to I. The lineage is computed in the standard way, i.e., each tuple $t \in I$ is adorned with a variable X_t, and these variables are then subsequently combined by means of the logical operators $\wedge$ and $\vee$ based on the structure of q_Σ. It is easily verified that the lineage Φ of q_Σ on instance I is given by

$$\Phi = \bigvee_{(s,t) \in \mathsf{vio}(I, \Sigma)} X_s \wedge X_t,$$

where X_s and X_t are boolean variables corresponding to tuples s and t, and where $\mathsf{vio}(I, \Sigma)$ is the set of all tuples in I that make up a violation of an FD in Σ.

Example 4. Continuing with the previous example, one can easily verify that $\mathsf{vio}(I, \Sigma)$ consists all pairs of tuples taken from $\{t_1, t_2, t_3\}$ and pairs taken from $\{t_4, t_5, t_6, t_7\}$. The lineage of the query q_Σ given in the previous example is equivalent to $(X_{t_1} \wedge X_{t_2}) \vee (X_{t_1} \wedge X_{t_3}) \vee (X_{t_2} \wedge X_{t_3}) \vee (X_{t_4} \wedge X_{t_5}) \vee (X_{t_4} \wedge X_{t_6}) \vee (X_{t_4} \wedge X_{t_7}) \vee (X_{t_5} \wedge X_{t_6}) \vee (X_{t_5} \wedge X_{t_7}) \vee (X_{t_6} \wedge X_{t_7})$. □

The results by Meliou et al. [5] imply that a tuple t in I is a cause if and only if X_t appears in Φ [1]. This can be checked in PTIME.

Example 5. Continuing with the previous example, note that every tuple in $t \in \mathsf{vio}(I, \Sigma)$ occurs as X_t in the lineage Φ of q_Σ. Furthermore, one can easily verify that Φ only contains variables X_t for tuples $t \in \mathsf{vio}(I, \Sigma)$. □

In fact, the correspondence shown in the previous example always holds:

Observation 1. For instance I and set Σ of FDs, a tuple $t \in I$ is a cause for q_Σ iff $t \in \mathsf{vio}(I, \Sigma)$. □

Since $\mathsf{vio}(I, \Sigma)$ can be computed in PTIME, the causality problem is indeed in PTIME in the tuple-based setting. This concludes our short recap of what is known in the tuple-based setting. We next consider the cell-based setting.

As it turns out, a similar correspondence exists between causes in our setting (Definition 4) and so-called *violating positions*, which we formally define below. First, we revise the notion of lineage from tuple to cell level by introducing variables corresponding to positions. That is, for a position $p = \langle t, A \rangle$ we let X_t^A be its corresponding variable. Consider an FD $\varphi : [B_1, \ldots, B_\ell] \to C$ and corresponding boolean query q_φ. Let s and t be two tuples in $\mathsf{vio}(I, \varphi)$. We encode this violation of φ by the *cell lineage*

$$(X_s^{B_1} = X_t^{B_1}) \wedge \cdots \wedge (X_s^{B_\ell} = X_t^{B_\ell}) \wedge (X_s^C \neq X_t^C),$$

which we also denote by $s[B_1, \ldots, B_\ell] = t[B_1, \ldots, B_\ell] \wedge s[C] \neq t[C]$. Consequently, the cell lineage for φ is then given by

$$\Phi_\varphi = \bigvee_{s,t \in \mathsf{vio}(I, \varphi)} s[B_1, \ldots, B_\ell] = t[B_1, \ldots, B_\ell] \wedge s[C] \neq t[C]$$

and the cell lineage for Σ is given by $\Phi = \bigvee_{\varphi \in \Sigma} \Phi_\varphi$.

Example 6. Recall the lineage Φ given in Example 4 and consider the disjunct $X_{t_1} \wedge X_{t_3}$ in Φ corresponding to a violation of $\varphi_2 : \texttt{Country} \to \texttt{CC}$. In the cell-based lineage $X_{t_1} \wedge X_{t_3}$ is replaced by $(X_{t_1}^{\texttt{Country}} = X_{t_3}^{\texttt{Country}}) \wedge (X_{t_1}^{\texttt{CC}} \neq X_{t_3}^{\texttt{CC}})$. Similarly for all other disjuncts in Φ. □

The semantics of Φ is as expected: $I \models (X_s^A = X_t^A)$ iff $\mathsf{val}(I, \langle s, A \rangle) = \mathsf{val}(I, \langle t, A \rangle)$ and thus $I \models \Phi$ iff $I \not\models \Sigma$.

In the cell-based setting, however, the full cell-lineage as defined just now exhibits certain redundancies. In analogy to [5], we say that a disjunct $s[V] = t[V] \wedge s[C] \neq t[C]$ in Φ is redundant if Φ contains another disjunct of the form $s[U] = t[U] \wedge s[C] \neq t[C]$ with $U \subsetneq V$. Clearly, redundant disjuncts can be identified in PTIME and can be omitted from the lineage: they do not add any additional information. We denote by Φ_{nr} the disjunction obtained

[1] For constraints beyond FDs, an additional pruning of redundant terms in Φ is required [5].

from Φ by removing all its redundant disjuncts. We call Φ_{nr} the *reduced* cell lineage. With this in place, we can make the connection between cell-based causes and lineage precise:

Theorem 1. *Let I be a database instance and Σ a set of FDs. Then $p = \langle t, A\rangle$ is a cause for (I, Σ) if X_t^A occurs in the reduced cell lineage Φ_{nr}.* □

An immediate consequence is that the causality problem is also in PTIME when cell-based repairs and FDs are concerned. Indeed, Φ_{nr} can be computed in PTIME.

Example 7. The need for only considering non-redundant disjuncts is illustrated by the following example. Consider the following set of FDs $\Sigma' = \{\varphi_3 : \mathtt{A} \to \mathtt{C}, \varphi_4 : \mathtt{AB} \to \mathtt{C}\}$ over a relation $R(\mathtt{A}, \mathtt{B}, \mathtt{C})$. Consider instance $I' = \{s_1 = (a, b, c_1), s_2 = (a, b, c_2)\}$. Clearly, $I' \not\models \Sigma'$. The cell-based lineage is given by $\Phi' = \big((s_1^{\mathtt{A}} = s_2^{\mathtt{A}}) \wedge (s_1^{\mathtt{C}} \neq s_2^{\mathtt{C}})\big) \vee \big((s_1^{\mathtt{A}} = s_2^{\mathtt{A}}) \wedge (s_1^{\mathtt{B}} = s_2^{\mathtt{B}}) \wedge (s_1^{\mathtt{C}} \neq s_2^{\mathtt{C}})\big)$. The second disjunct in Φ' is redundant. Hence, $\Phi'_{nr} = (s_1^{\mathtt{A}} = s_2^{\mathtt{A}}) \wedge (s_1^{\mathtt{C}} \neq s_2^{\mathtt{C}})$. Consider position $\langle s_1, \mathtt{B}\rangle$ which occurs in Φ' but not in Φ'_{nr}. This absence is for a good reason. Indeed, $\langle s_1, \mathtt{B}\rangle$ is not a cause. Changing the value of this position does not resolve the violation for φ_3. Furthermore, allowing a change in any other position results in a repair, either by ensuring that $(s_1^{\mathtt{A}} \neq s_2^{\mathtt{A}})$ or that $(s_1^{\mathtt{C}} = s_2^{\mathtt{C}})$ holds. Hence, there is no contingency set for this position. The use of the reduced cell lineage is thus crucial. □

We conclude our discussion on the causality problem by providing an alternative characterisation of causes. For this purpose we consider *violating positions*. That is, given an instance I and a set Σ of FDs, we say that a position $p = \langle t, A\rangle$ is *violating* if $t \in \mathsf{vio}(I, \varphi)$ for some $\varphi : X \to B$ in Σ and $A \in X \cup B$. The set of violating positions for (I, Σ) will be denoted by $\mathsf{viop}(I, \Sigma)$.

Example 8. All positions in bold in Figure 1a are violating positions for $\Sigma = \{\varphi_1 : \mathtt{City} \to \mathtt{Country}, \varphi_2 : \mathtt{Country} \to \mathtt{CC}\}$. □

To rephrase Theorem 1 in terms of violating positions we reduce the set Σ of FDs such that it is guaranteed that when the cell lineage is computed, it is automatically reduced. More precisely, let Σ_{nr} be the subset of Σ consisting of all FDs $U \to C$ such that there is no FD $V \to C$ with $U \subsetneq V$ in Σ. Then, from the previous discussion and from Theorem 1 we may conclude the following.

Observation 2. For an instance I and set Σ of FDs, a position p is a cause for (I, Σ) iff $p \in \mathsf{viop}(I, \Sigma_{nr})$. □

Note that Σ_{nr} is used rather than the original set Σ.

Example 9. For Σ given in Figure 1a, observe that $\Sigma_{nr} = \Sigma$ and hence all bold positions in Figure 1a are causes. Indeed, these are precisely the violating positions. Furthermore, for the set of FDs given in Example 7, $\Sigma'_{nr} = \{\varphi_3 : \mathtt{A} \to \mathtt{C}\} \neq \Sigma'$ and the violating positions (causes) in I' are $\{\langle s_1, \mathtt{A}\rangle, \langle s_2, \mathtt{A}\rangle, \langle s_1, \mathtt{C}\rangle, \langle s_2, \mathtt{C}\rangle\}$. □

We may conclude that the notion of causality in the context of cell-based repairs has a very intuitive interpretation in terms of violations of FDs.

3.2 The responsibility problem

We next consider the responsibility problem. More specifically, we show that this problem is NP-hard. Note that the intractability results of the responsibility problem given in [5] and [7] are not applicable in our setting. Indeed, Meliou et al. [5] consider self-join-free conjunctive queries whereas q_φ for $\varphi \in \Sigma$ clearly contains self-joins; Salimi and Bertossi [7] consider tuple-based repairs, while we consider cell-based repairs.

Proposition 1. *The responsibility problem is NP-hard.*

Proof. The NP-hardness is established by a reduction from the V-repair checking problem, stated at the end of the previous section, which is known to be NP-complete [4]. Given an instance I of R, set Σ of FDs and integer k as input for the repair checking problem, we construct an instance I' of R', set Σ' of FDs, identify a position $p \in \mathsf{Pos}(I')$, and rational number r, such that $\rho(p) > r$ if and only if there is a V-repair I_P for (I, Σ) such that $|P| < k$.

The construction is as follows. First, we expand $R(A_1, \ldots, A_m)$ to $R'(A, B, A_1, \ldots, A_m)$ where A and B are new attributes. Next, we let $\Sigma' = \Sigma \cup \{A \to B\}$. Furthermore, I' consists of the following tuples: for each $t \in I$, $t' = (a_t, b_t, t) \in I'$ where a_t and b_t are constants not used anywhere else. In addition, I' contains tuples $s = (a, b, c_1, \ldots, c_m)$ and $t = (a, b', d_1, \ldots, d_m)$, again using new constants. Finally, we set $r = \frac{1}{k+1}$ and let $p = \langle t, B\rangle$. This concludes the construction.

We next verify its correctness. Suppose that I_P is a V-repair for (I, Σ) with $|P| < k$. Then clearly, I'_P is not a V-repair for (I', Σ') since s and t are violating the FD $A \to B$, and P does not contain any of the positions $\langle s, A\rangle$, $\langle s, B\rangle$, $\langle t, A\rangle$, and $\langle t, B\rangle$. Observe, however, that $I'_{P\cup p}$ is a V-repair for (I', Σ'). Indeed, it suffices to extend the valuation $\nu(I_P) \models \Sigma$ to the valuation ν' of $I'_{P\cup p}$ such that $\nu'(t[B]) = s[B]$, ν' coincides with ν on positions in I, and ν' is the identity everywhere else. Clearly, $\nu'(I'_{P\cup p}) \models \Sigma'$. Hence, P is a contingency set for p. We thus have that $\rho(p) = \frac{1}{1+|\Gamma^m|} \geqslant \frac{1}{1+|P|} > \frac{1}{1+k} = r$, as desired.

Conversely, suppose that $\rho(p) > r$ and consider the minimal contingency set Γ^m for p and (I', Σ'). From the construction of I' and Σ' it can be readily verified that Γ^m can only contain positions in $\mathsf{Pos}(I)$. However, since $I'_{\Gamma^m\cup p}$ is a V-repair for (I', Σ') it must be the case that I_{Γ^m} is a V-repair for (I, Σ). Observe that $|\Gamma^m| < \frac{1}{r} - 1 = k$. Hence, there is a V-repair I_P for (I, Σ) such that $|P| < k$, as desired. □

For the upper bound, we only have the trivial $\Sigma_2^p = \mathrm{NP}^{\mathrm{NP}}$ upper bound for the responsibility problem. Indeed, one can (i) simply guess a set Γ of at most $\frac{1}{r} - 1$ positions in I, and verify that (ii) I_Γ is not a V-repair for (I, Σ) and (iii) $I_{\Gamma\cup p}$ is a V-repair. Step (ii) requires a call to an coNP-oracle and step (iii) requires a call to an NP-oracle. Hence, the overall algorithm is indeed in Σ_2^p. We leave it for future work to establish the precise complexity of the responsibility problem in our setting.

3.3 Relationship to SV- and CV-repairs

In analogy to Salimi and Bertossi [7], we next relate causes and contingency sets to SV- and CV-repairs. We need the following notation. Given an instance I and set Σ of FDs, we define

$$\mathcal{S} = \{(p, \Gamma) \mid \Gamma \text{ is a contingency set for } p \text{ and } (I, \Sigma), \text{ and no } \Gamma' \subsetneq \Gamma \text{ is a contingency set for } p \text{ and } (I, \Sigma)\}.$$

Proposition 2. *Given I, Σ and $P \subseteq \mathsf{Pos}(I)$, we have that I_P is an SV-repair if and only if $P = \Gamma \cup p$, for $(p, \Gamma) \in \mathcal{S}$.*

Proof. Suppose that I_P is an SV-repair. Then, I_P is a V-repair and for any $P' \subsetneq P$, $I_{P'}$ is not a V-repair. This implies that for any $p \in P$, $P' = P \setminus p$ is a contingency set of p. Suppose, for the sake of contradiction, that there is a $P'' \subsetneq P$ such that P'' is a contingency set of p as well. This would imply that $I_{P''\cup p}$ is a V-repair, contradicting the fact that I_P is an SV-repair. Hence, $(p, P \setminus p) \in \mathcal{S}$.

Conversely, let $(p, \Gamma) \in \mathcal{S}$. Then, $I_{\Gamma\cup p}$ is a V-repair, whereas I_Γ is not. Suppose, for the sake of contradiction, that there is a $\Gamma' \subsetneq \Gamma \cup p$ such that $I_{\Gamma'}$ is a V-repair. In other words, that $I_{\Gamma\cup p}$ is not an SV-repair. Note that Γ' must include p. However, this implies that $I_{\Gamma'\setminus p}$ is not a V-repair and thus $\Gamma' \setminus p$ is a contingency set

	FN	LN	City	Country	CC	AC	Phone
t_1	John	Lewis	New York	USA	01	212	509 3883
t_2	David	Smith	New York	UK	44	212	509 7639
t_3	Lucas	Reed	New York	UK	01	212	509 8291
t_4	Luke	White	London	UK	01	020	9876 0402
t_5	Jane	Taylor	London	UK	43	020	9876 3266
t_6	Mark	Hill	London	UK	44	020	9876 2911
t_7	Ron	Clark	London	UK	44	020	9876 1059
t_8	Will	Green	Toronto	Canada	01	416	511 7145
t_9	Lisa	Jones	Toronto	Canada	01	416	511 5200

(a)

$I_1 =$

City	Country	CC
New York	USA	01
New York	USA	44
New York	USA	01
London	UK	01
London	UK	43
London	UK	44
London	UK	44
Toronto	Canada	01
Toronto	Canada	01

(b)

$I_2 =$

City	Country	CC
New York	USA	01
New York	USA	44
New York	USA	01
London	UK	01
London	UK	43
London	UK	44
London	UK	44
Toronto	Canada	01
Toronto	Canada	01

(c)

$I_3 =$

City	Country	CC
New York	USA	01
New York	USA	01
New York	USA	01
London	UK	01
London	UK	43
London	UK	44
London	UK	44
Toronto	Canada	01
Toronto	Canada	01

(d)

$I_4 =$

City	Country	CC
New York	USA	01
New York	USA	01
New York	USA	01
London	UK	44
London	UK	44
London	UK	44
London	UK	44
Toronto	Canada	01
Toronto	Canada	01

(e)

$\Sigma = \{\varphi_1 : \texttt{City} \rightarrow \texttt{Country}, \varphi_2 : \texttt{Country} \rightarrow \texttt{CC}\}$

Figure 2. (a) Example of chase sequence on a database instance *I* with a set of functional dependencies Σ as detailed in Example 12. Here, underlined values belong to positions that are curated and grey shaded positions indicate that those positions are relevant for each of the chase steps. The intermediate chase results I_1, I_2, I_3 and end result I_4 are shown in (b), (c), (d), and (e), respectively.

for p, which is a subset of Γ. This contradicts the assumption that $(p, \Gamma) \in \mathcal{S}$. Hence, $I_{\Gamma \cup p}$ is an SV-repair, as desired. □

Given that a CV-repair I_P for (I, Σ) is an SV-repair such that $|P|$ is minimal, the following correspondence follows immediately from the previous proposition.

Proposition 3. *Given I, Σ and $P \subseteq \mathsf{Pos}(I)$, we have that I_P is a CV-repair if and only if $P = \Gamma^m \cup p$ and the degree of responsibility $\rho(p)$ is highest among all causes for (I, Σ).* □

Recall that Γ^m is the minimal cardinality contingency set for p and (I, Σ).

4. Chase repairs

The notions of SV-repairs and CV-repairs are not entirely satisfactory in practice. Indeed, it is common that some tuples or even individual values in an instance may be considered entirely trustworthy. This happens, for example, when these have been manually curated by an expert, are the result of a merge from a trusted instance with another, or when provenance information is provided from which a sense of trust can be derived. The repair models considered so far largely ignore such curated positions.

Furthermore, SV-repairs and CV-repairs do not take into account any procedural aspects that underlie many of the existing repairing algorithms (see e.g., [1, 2] for an overview of repairing algorithms). More precisely, most repairing algorithms can be seen as variations of the chase procedure. In such a chase procedure constraints are repeatedly applied (chased) until either all inconsistencies have been removed, or some conflict is encountered that prevents a successful repair of the data (in which case the chase fails).

In this section we propose a new repair model, called *chase repair*, as an attempt to address the above shortcomings of SV-repairs and CV-repairs. We further revisit the notion of cause in this context and argue for the need of a provenance (lineage) model for data quality and data repairing in particular. We start off with an example.

Example 10. Recall the SV-repair I_P and CV-repair $I_{P'}$ from Example 1. Also consider the original instance I, but this time supplemented with a set of curated positions. These curated positions are underlined in Figure 2(a). Based on the set of curated positions, it is readily observable that the SV-repair I_P performed incorrect value modifications at positions $\langle t_2, \texttt{City} \rangle$ and $\langle t_5, \texttt{City} \rangle$. Indeed, these positions are curated and thus should not be modified. Similarly, the CV-repair $I_{P'}$ performed an incorrect value modification at position $\langle t_2, \texttt{City} \rangle$, for the same reason. □

Although one could remedy this situation by extending the notion of V-repairs such that no modifications are allowed in curated positions, we further want to use the presence of curated positions to guide the repairing process, hereby leveraging the functional dependencies. Abstracting away particularities of existing chase-based repairing algorithms we observe the following: (i) when resolving violations of an FD $X \rightarrow A$, the violating tuples must have curated positions in the attributes in X; (ii) in order to repair the inconsistency in attribute A, a unique curated value should be present, which will be used to resolve the violation; and finally, (iii) when such a repair has been made, the previous violating tuples now also carry curated positions in their A-attribute. In other words, the FDs are used to propagate curated information.

Example 11. Consider FD $\varphi_1 : \texttt{City} \rightarrow \texttt{Country}$ and tuples $\{t_1, t_2, t_3\} \in \mathsf{vio}(I, \varphi_1)$. These tuples all carry curated positions for attribute City and moreover $\langle t_1, \texttt{Country} \rangle$ is curated. This dictates that these violations should be resolved by setting $t_2[\texttt{Country}]$ and $t_3[\texttt{Country}]$ to the value $t_1[\texttt{Country}] = \text{USA}$. Furthermore, $\langle t_2, \texttt{Country} \rangle$ and $\langle t_3, \texttt{Country} \rangle$ become curated, although these positions were not originally marked as such. □

We next formalise these observations in terms of *chase repairs*. We first define a chase step.

Definition 6 (Chase step)**.** Consider a database instance I, an FD $\varphi : X \rightarrow A$ and a set of curated positions $\mathsf{CPos}(I)$. Let T be a set of tuples $\{t_1, \ldots, t_m\}$ in I for which $t_i[X] = t_j[X]$, for each $i, j \in [1, m]$, and furthermore, all positions in $T[X]$ are curated. We say that φ is applicable to T if either (a) there is a violation for

φ in T (in other words, $\mathsf{vio}(T, \varphi)$ is non-empty); or (b) there is no violation but there is at least one $t_i \in T$ such that $\langle t_i, A\rangle$ is not yet curated.

If φ is applicable to T and none of the positions $\langle t_i, A\rangle$ is curated, then the result of the chase is a "fail". Similarly, the chase fails when there are two curated positions $\langle t_i, A\rangle$ and $\langle t_j, A\rangle$ that carry distinct values. Otherwise, the chase of T with φ is successful and results in an update of I in which all $t_i[A]$ carry the same value as dictated by a curated position $\langle t_j, A\rangle$; and an update of $\mathsf{CPos}(I)$ by adding all the positions $\langle t_i, A\rangle$, for $t_i \in T$. □

We denote by $(I, \mathsf{CPos}(I)) \rightarrow_{T,\varphi} (I', \mathsf{CPos}(I'))$ that I' is the result of chasing T in I with φ; similarly for $\mathsf{CPos}(I)$ and $\mathsf{CPos}(I')$. A chase sequence for (I, Σ) relative to a set $\mathsf{CPos}(I)$ of curated positions is a sequence of the form $(I, \mathsf{CPos}(I)) \rightarrow_{T_1,\varphi_1} (I_1, \mathsf{CPos}(I_1)) \rightarrow_{T_2,\varphi_2} \cdots \rightarrow_{T_n,\varphi_n} (I_n, \mathsf{CPos}(I_n))$, such that each $\varphi_i \in \Sigma$, and no further chase steps can be applied. In this case, we call I_n a result of chasing I with Σ relative to $\mathsf{CPos}(I)$.

Definition 7 (Chase repair)**.** Given a database instance I, set Σ of FDs, and an initial set of curated positions $\mathsf{CPos}(I)$, we say that $(I, \mathsf{CPos}(I))$ is a chase repair if there is an instance $I' \models \Sigma$ that can be obtained as the result of chasing I with Σ, relative to $\mathsf{CPos}(I)$. □

In other words, the dirty database I has been sufficiently curated to guarantee that the chase procedure returns a clean instance I'.

Example 12. Consider the database instance I and the set Σ of FDs given in Figure 2a. All underlined values belong to positions in $\mathsf{CPos}(I)$. Grey shaded entries indicate the set T of positions that are used in the different chase steps. To show that $(I, \mathsf{CPos}(I))$ is a chase repair, consider the following chase sequence which starts from I and $\mathsf{CPos}(I)$ and then: (i) chases with FD φ_1 and $T_1 = \{t_1, t_2, t_3\}$ to obtain a new instance $I_1 = I$, where $\mathsf{val}(I_1, \langle t_2, \texttt{Country}\rangle) = \mathsf{val}(I_1, \langle t_3, \texttt{Country}\rangle) =$ USA. Furthermore, $\mathsf{CPos}(I_1) = \mathsf{CPos}(I) \cup \langle t_2, \texttt{Country}\rangle \cup \langle t_3, \texttt{Country}\rangle$, as shown in Figure 2(b); next (ii) chases with FD φ_1 and $T_2 = \{t_4, t_5\}$ to obtain a new instance I_2 and an updated set of curated positions, as shown in Figure 2(c); next (iii) chases with FD φ_2 and $T_3 = \{t_1, t_2, t_3\}$ to obtain a new instance I_3 and an updated set of curated positions, as shown in Figure 2(d); and finally (iv) chases with FD φ_2 and $T_4 = \{t_4, t_5, t_6\}$ to obtain the end result $I_4 \models \Sigma$ and an updated set of curated positions, as shown in Figure 2(e). □

We are particularly interested in studying the causality and responsibility problem for chase repairs. Note that causes in this setting are related to positions that affect the *success* of chasing the dirty instance with the FDs, rather that the cause for inconsistencies.

Definition 8. A position p in $\mathsf{Pos}(I)$ is a cause for (I, Σ) if there is a set of curated positions C in I such that (I, C) is not a chase repair whereas $(I, C \cup p)$ is a chase repair. □

In other words, p together with C must be curated in order for the chase-based repairing algorithm to be successful.

Example 13. It is easy to verify that position $p = \langle t_1, \texttt{Country}\rangle$ is a cause for the instance I and set Σ of FDs, shown in Figure 2a. Indeed, consider $C = \{\langle t_1, \texttt{City}\rangle, \langle t_2, \texttt{City}\rangle, \langle t_3, \texttt{City}\rangle, \langle t_4, \texttt{City}\rangle, \langle t_5, \texttt{City}\rangle, \langle t_6, \texttt{Country}\rangle, \langle t_3, \texttt{CC}\rangle, \langle t_6, \texttt{CC}\rangle\}$. Then, (I, C) is not a chase repair since the violation of φ_1 in the first three tuples cannot be resolved by the chase. However, $(I, C \cup p)$ is a chase repair as illustrated in the previous example. □

Further investigation is required as to establish the complexity of the corresponding causality and responsibility problem, and to provide a characterisation of causes similar to the ones given for V-repairs.

With regard to the causality problem, one way forward is to develop a provenance (lineage) model that is expressive enough to record information about chase-like procedures such as the one used by chase repairs. Note that in general, the order in which chase steps may be applied may be important and thus the provenance model must be capable of dealing with this. With such a provenance model in place, one could envisage a characterisation similar to the one given in Theorem 1. More specifically, causes and their corresponding (curated) contingency sets must be closely related to the obtained chase provenance. To our knowledge, no lineage (or more generally provenance) model is in place yet that ticks all these boxes [2].

Provenance models for repairing strategies may not only be helpful in understanding causality. Indeed, we believe that they must form an essential part of data quality systems. It is often said that provenance is useful in the context of data quality, but so far no formal framework is in place to validate this claim. Finding a suitable provenance representation for chase repairs may be a first step in this direction.

We are also convinced that causes, and their responsibility in particular, may be helpful in designing better repairing algorithms. Indeed, the most responsible causes should be the ones that are repaired first.

5. Conclusions

This paper is a preliminary investigation on the interaction of cell-based V-repairs and causality and the use of provenance information to better understand causality in the context of data quality. For more complex repair models, like chase repairs, we strongly believe that suitable provenance models need to be developed to make similar connections between causality and provenance. In particular, provenance models that reflect the operational steps executed to obtain a repair are required to better understand why and how repairs are obtained, and how to debug the repairing process if the suggested repair is not satisfactory to the user.

References

[1] Leopoldo E. Bertossi, *Database Repairing and Consistent Query Answering*. Synthesis Lecture Series, Morgan & Claypool, 2011.

[2] Wenfei Fan and Floris Geerts, *Foundations of Data Quality Management*. Synthesis Lecture Series, Morgan & Claypool, 2012.

[3] Ioana Ileana, Bogdan Cautis, Alin Deutsch, and Yannis Katsis, *Complete Yet Practical Search for Minimal Query Reformulations Under Constraints*. SIGMOD, pp 1015–1026, 2014.

[4] Solmaz Kolahi and Laks V.S. Lakshmanan, *On Approximating Optimum Repairs for Functional Dependency Violations*. ICDT, pp. 53–62, 2009.

[5] Alexandra Meliou, Wolfgang Gatterbauer, Joseph Y. Halpern, Christoph Koch, Katherine F. Moore, and Dan Suciu, *Causality in Databases*. IEEE Data Eng. Bull., 33(3), pp. 59–67, 2010.

[6] Alexandra Meliou, Wolfgang Gatterbauer, and Dan Suciu, *Bringing Provenance to its Full Potential using Causal Reasoning.* TAPP, 2011.

[7] Babak Salimi and Leopoldo E. Bertossi, *From Causes for Database Queries to Repairs and Model-Based Diagnosis and Back*. ICDT, pp. 342–362, 2015.

[2] The interaction between chase and provenance has been studied in [3], but for the purpose of query reformulation optimization. This approach does not seem applicable to our setting.