Quantifying Causal Effects on Query Answering in Databases

Babak Salimi
University of Washington, Seattle, USA
bsalimi@cs.washington.edu

Leopoldo Bertossi
Carleton University, Ottawa, Canada
bertossi@scs.carleton.ca

Dan Suciu
University of Washington, Seattle, USA
suciu@cs.washington.edu

Guy Van den Broeck
University of California, Los Angeles, USA
guyvdb@cs.ucla.edu

Abstract

The notion of actual causation, as formalized by Halpern and Pearl, has been recently applied to relational databases, to characterize and compute *actual causes* for possibly unexpected answers to monotone queries. Causes take the form of database tuples, and can be ranked according to their *causal responsibility*, a numerical measure of their relevance as a cause for the query answer. In this work we revisit this notion, introducing and making a case for an alternative measure of causal contribution, that of *causal effect*. In doing so, we generalize the notion of actual cause, in particular, going beyond monotone queries. We show that causal effect provides intuitive and intended results.

1. Introduction

The central aim of many scientific disciplines, ranging from philosophy through law and physiology to computer science, is the elucidation of cause-effect relationships among variables or events. In data management in particular, there is a need to represent, characterize and compute causes that explain why certain query results are obtained or not. The notion of causality-based explanation for a query result was introduced in [16], on the basis of the deeper concepts of *counterfactual* and *actual causation* introduced by Halpern and Pearl in [13], which we call HP-causality. We will refer to this notion as *query-answer causality*, or simply, *QA-causality*.

Intuitively, a database atom (or simply, a tuple) τ is an *actual cause* for an answer $\bar{a}$ to a monotone query $\mathcal{Q}$ from a relational database instance D if there is a "contingent" subset of tuples Γ, accompanying τ, such that after removing Γ from D: (a) $\bar{a}$ is still an answer to the query, and (b) further removing τ from $D \setminus \Gamma$, makes $\bar{a}$ not an answer to the query anymore. (I.e. τ is a *counterfactual cause* under $D \setminus \Gamma$.)

In [16], the notion of causal responsibility in databases was introduced, to provide a metric to quantify the causal contribution, as a numerical degree, of a tuple to a query answer. This responsibility-based ranking is considered as one of the most important contributions of HP-causality and its extension [7] to data management [15]. In informal terms, causal responsibility as in [7] tells us that, for variables A and B, the degree of responsibility of A for B should be $\frac{1}{(N+1)}$, where N is the *minimum number of changes* that have to be made on other variables to obtain a situation where B counterfactually, directly depends on A. In the case of databases, the responsibility of a cause τ for an answer $\bar{a}$, is defined as $\frac{1}{1+|\Gamma|}$, where Γ is a smallest-size contingency set for τ.

Apart from the explicit use of causality, most of the related research on explanations for query results has concentrated on *data provenance* [3, 4, 8, 14]. Causality has been discussed in relation to *data provenance* [16, 15] and *workflow provenance* [6]. Specifically, in [16] a close connection between QA-causality and *why-provenance* (in the sense of [3]) was established.

In a different direction, correspondences between causal responsibility and other concepts and problems in databases, e.g. the *view-update problem* and *database repairs* have been established in [19, 20, 5, 21]. The underlying reason for these connections is the need to perform a minimal set or minimum number of changes on the database, so that the resulting state of the database has a desired property. Accordingly, we can see that actual causality and causal responsibility are indeed important concepts that may unify several problems in data management.

The notion of causal responsibility as introduced in [7] has been subject to some criticism lately [23, 2, 12, 18]. In the context of databases, it has been shown in [18] that causal responsibility only partially fulfils the original intention of adequately ranking tuples according to their causal contribution to an answer. We illustrate some of these issues by means of an example (for others and a discussion, see [18]).

E	A	B
t_1	a	b
t_2	a	c
t_3	c	b
t_4	a	d
t_5	d	e
t_6	e	b

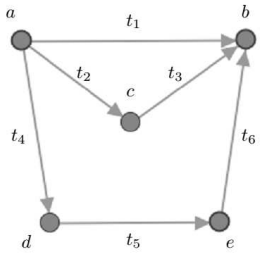

Figure 1: Instance D and its associated graph G

Example 1. Consider instance D with a single binary relation E as in Figure 1. For simplicity, we use the identifiers, t_1-t_6, to refer to the database tuples. Instance D can be represented as the directed graph $G(\mathcal{V}, \mathcal{E})$ in Figure 1, where $\mathcal{V}$ is the active domain of D, and $\mathcal{E}$ contains an edge (v_1, v_2) iff $E(v_1, v_2) \in D$. Tuple identifiers are used as labels for the corresponding edges in the graph.

TaPP 2016, June 8–9, 2016, Washington, DC.

Now consider query, $\mathcal{Q}$ asking if there exists a path between a and b. This query is monotone, Boolean (i.e. it has a *true*/*false* answer), and can be expressed in recursive Datalog. The answer is *true* in D. All the tuples are actual causes for this answer, with the same causal responsibility: $\frac{1}{3}$. However, since t_1 provides a direct connection, it makes sense to claim that t_1 contributes more to this answer than the other tuples. Intuitively, tuples that belong to shorter paths between a and b contribute more to the answer than tuples that belong to longer paths. □

In this work we introduce the notion of causal effect in the context QA-causality in databases. Causal effect refers to the extent to which an input variable has a direct influence or drive on the next state of a output variable, i.e. addressing questions of the form: "If we change the state of the input, to what extent does that alter the state of the output?". To achieve this goal, we start from the central notion of *causal effect* that is used in the *theory of causal inference* proposed in [17]. Introducing it in the context QA-causality in databases allows us to define and investigate the notion of *causal effect of a tuple on the answer to a Boolean query*.

We show that, in databases, causal effect subsumes the notion of actual causal as introduced in [16], and can be sensibly applied not only to monotone queries, but also to first-order queries with negation, and to aggregate queries. Furthermore, we illustrate, by means of several examples, that this notion provides an intuitive and informative alternative to causal responsibility when ranking causes according to their causal contribution to a query result.

2. Preliminaries

We consider relational database schemas of the form $\mathcal{S} = (U, \mathcal{P})$, where U is a finite database domain of *constants* and $\mathcal{P}$ is a finite set of *database predicates* of fixed arities. In some cases, we may also have *built-in* predicates, e.g. $\neq$, that we leave implicit. A database instance D compatible with $\mathcal{S}$ can be seen as a finite set of ground atomic formulas of the form $P(c_1, ..., c_n)$, where $P \in \mathcal{P}$ has arity n, and $c_1, \ldots, c_n \in U$. In databases these formulas are usually called atoms or *tuples*. They will be denoted with $\tau, \tau_1, \ldots$. The active domain of an instance D, denoted $Adom(D)$, is the finite set of constants from U that appear in D.

In this work, we will mostly consider first-order (FO) queries, that is, formulas $\mathcal{Q}(\bar{x})$ of the language of FO predicate logic, $\mathcal{L}(\mathcal{S})$, associated to $\mathcal{S}$. In $\mathcal{Q}(\bar{x})$, $\bar{x}$ shows all the free variables in the formula. If $\bar{x}$ is non-empty, the query is *open*. If $\bar{x}$ is empty, the query is *Boolean*, i.e. a sentence, in which case, the answer is *true* or *false* in a database, denoted by $D \models \mathcal{Q}$ and $D \not\models \mathcal{Q}$, respectively. A sequence $\bar{c}$ of constants is an answer to an open query $\mathcal{Q}(\bar{x})$ if $D \models \mathcal{Q}[\bar{c}]$, i.e. the query becomes true in D when the variables are replaced by the corresponding constants in $\bar{c}$. We denote with $\mathcal{Q}(D)$ the set of all answers to query $\mathcal{Q}(\bar{x})$.

In particular, a *conjunctive query* (CQ) is a formula of the form $\mathcal{Q}(\bar{x}) : \exists \bar{y}(P_1(\bar{s}_1) \wedge \cdots \wedge P_m(\bar{s}_m))$, where the $P_i(\bar{s}_i)$ are atomic formulas, i.e. $P_i \in \mathcal{P}$, and the $\bar{s}_i$ are sequences of terms, i.e. variables or constants.[1] When $\bar{x}$ is empty, the query is *Boolean* conjunctive query (BCQ).

A query $\mathcal{Q}$ is *monotone* if for every two instances $D_1 \subseteq D_2$, $\mathcal{Q}(D_1) \subseteq \mathcal{Q}(D_2)$, i.e. the set of answers grows monotonically with the instance. For example, CQs and unions of CQs (UCQs) are monotone. Datalog queries [1], although not always expressible as FO queries, are also monotone. Although most of the work on QA-causality has concentrated on monotone queries, in this work we will also consider non-monotone queries.

Now we review the notions of QA-causality and responsibility as introduced in [16]. Assume the relational instance D is split in two disjoint sets: $D = D^n \cup D^x$, where D^n and D^x are the sets of *endogenous* and *exogenous* tuples, respectively.[2] Let $\mathcal{Q}$ be a monotone Boolean query. A tuple $\tau \in D^n$ is a *counterfactual cause* for an answer $\mathcal{Q}$ in D if $D \models \mathcal{Q}$, but $D \smallsetminus \{\tau\} \not\models \mathcal{Q}$. A tuple $\tau \in D^n$ is an *actual cause* for $\mathcal{Q}$ if there exists $\Gamma \subseteq D^n$, called a *contingency set*, such that τ is a counterfactual cause for $\mathcal{Q}$ in $D \smallsetminus \Gamma$. The *causal responsibility* of a tuple τ for answer $\bar{a}$, denoted $\rho_{\mathcal{Q}}(\tau)$, is $\frac{1}{1+|\Gamma|}$, where $|\Gamma|$ is a smallest-size contingency set for τ. When τ is not an actual cause for $\bar{a}$, no contingency set is associated to τ. In this case, $\rho_{\mathcal{Q}}(\tau)$ is defined as 0.

2.1 Lineage of a query

The *lineage (expression)* of a Boolean FO query $\mathcal{Q}$, as used in probabilistic databases [10], is a propositional formula, $\Phi_{\mathcal{Q}}$, over the finitely many potential tuples in an arbitrary database instance for the schema at hand, i.e. all tuples $\tau : P(c_1, \ldots, c_n)$, with n-ary $P \in \mathcal{P}$ and $c_1, \ldots, c_n \in U$. For each such a τ, we introduce a propositional a propositional variable X_τ (aka. a propositional atom). $Var(\mathcal{S}, U)$ denotes the set of variables associated to tuples. It depends on the schema and data domain, and determines a propositional language $\mathcal{L}(Var(\mathcal{S}, U))$.

Formula $\Phi_{\mathcal{Q}}$ expresses which input tuples must be present in the database and which tuples must be absent from it for the query to be true. $\Phi_{\mathcal{Q}}$ is defined inductively for first-order (FO) queries $\mathcal{Q}$, as follows: 1. If $\mathcal{Q}$ is a tuple τ, $\Phi_\tau := X_\tau$. 2. $\Phi_{a=a} := true$. 3. $\Phi_{a=b} := false$. 4. $\Phi_{\mathcal{Q}_1 \wedge \mathcal{Q}_2} := \Phi_{\mathcal{Q}_1} \wedge \Phi_{\mathcal{Q}_2}$. 5. $\Phi_{\mathcal{Q}_1 \vee \mathcal{Q}_2} := \Phi_{\mathcal{Q}_1} \vee \Phi_{\mathcal{Q}_2}$. 6. $\Phi_{\exists x \mathcal{Q}} := \bigvee_{c \in U} \Phi_{\mathcal{Q}[\frac{c}{x}]}$. 7. $\Phi_{\neg \mathcal{Q}} := \neg(\Phi_{\mathcal{Q}})$.

For a query $\mathcal{Q}$, $Var(\Phi_{\mathcal{Q}})$ denotes the set of propositional variables in $\Phi_{\mathcal{Q}}$. Clearly, $Var(\Phi_{\mathcal{Q}}) \subseteq Var(\mathcal{S}, U)$.

We can make the lineage of a query depend on the instance D at hand, by assigning -at least conceptually- truth values to some of the variables appearing in $\Phi_{\mathcal{Q}}$ depending on the contents of D. More specifically, under the assumption that negation appears only in front of propositional variables (producing negative *literals*), the *D-lineage* of $\mathcal{Q}$, denoted $\Phi_{\mathcal{Q}}(D)$ is obtained from $\Phi_{\mathcal{Q}}$ by: (a) Making *false* each positive occurrence of variable X_τ for which $\tau \notin D$. (b) Making each $\neg X_\tau$ *false* for which $\tau \in D$. We denote with $Var^D(\Phi_{\mathcal{Q}}(D))$ the set of variables in $\Phi_{\mathcal{Q}}(D)$. *We will assume that: (a) a variable X_τ never appears both positively and negatively in $\Phi_{\mathcal{Q}}(D)$; and (b) every $\tau \notin D$ that appears negatively in $\Phi_{\mathcal{Q}}(D)$ is considered to be endogenous.* (Then, endogenous tuples are those in D^n plus some outside the instance at hand.)

Example 2. Consider an schema with two relations, $R(A, B)$ and $S(B)$, instance $D = \{R(a, b), R(a, c), R(c, b), S(c)\}$, with data domain $U = \{a, b, c\}$, and the Boolean query $\mathcal{Q} : \exists x(R(x, y) \wedge \neg S(y))$, which has answer *true* in D. We obtain the following "instantiated" lineage: $\Phi_{\mathcal{Q}}(D) = (X_{R(a,b)} \wedge \neg X_{S(b)}) \vee (X_{R(c,b)} \wedge \neg X_{S(b)})$, with $Var^D(\Phi_{\mathcal{Q}}(D)) = \{X_{R(a,b)}, X_{S(b)}, X_{R(c,b)}, X_{S(b)}\}$. □

For monotone FO queries, this instantiated lineage corresponds to the PosBool provenance semi-ring [14], and is related to the *minimal-witness-basis*, or *why-provenance* [3]. Notice that lineage can be naively extended to Datalog queries by considering the ground instantiation of a program and disjunctively collecting paths from the query goal all the way down, through backward-propagation via the ground propositional rules, to the ground extensional tuples τ, for which variables X_τ are introduced.

[1] We say it explicitly when we allow the P_i to be built-ins.

[2] Endogenous tuples are admissible, possible candidates for causes, as opposed to exogenous tuples. The partition is application-dependent and captures predetermined factors, such as users preferences that may affect QA-causal analysis.

3. Interventions and Causal Effect

HP-causality [13], which is the basis for the notion of QA-causality in [16], provides a "structural" model of actual causation. According to that approach, a *causal model* of a particular domain is represented in terms of variables, say $A, B, ...$, their values, and a set of *structural equations* representing causal relationships between variables [13]. In this context, the statement "A is an actual cause for B" claims that there is a set of possible *interventions* (or contingencies) on the causal model that makes B *counterfactually* depend on A. That is, had A not happened, B wouldn't have happened.

In QA-causality, counterfactual questions take concrete forms, such as: "What would be (or how would change) the answer to a query $\mathcal{Q}$ if the tuple τ is deleted/inserted from/into the database D?" A question like this can be addressed by building a corresponding causal model, which, for a query $\mathcal{Q}$ and instance D, becomes the combination of the query lineage $\Phi_{\mathcal{Q}}(D)$ and the truth assignment σ_D determined by D (X_τ is true iff $\tau \in D$). This models captures the causal relationships between database tuples (or their propositional variables) and $\mathcal{Q}$.

The interventions that represent counterfactual hypothesis become, in this context, insertions or deletions of database tuples τ, which change the truth values originally assigned by σ_D to the propositional variables X_τ appearing in the query lineage $\Phi_{\mathcal{Q}}(D)$.

Informally for the moment, interventions will be assignments (or changes) of truth values to (some of) the variables in the lineage. At some point later on, we will deal with the truth values assigned to variables in $Var^x(\Phi_{\mathcal{Q}}(D))$, the set of variables in $Var(\Phi_{\mathcal{Q}}(D))$ corresponding to exogenous tuples. Positive "exogenous variables" in $\Phi_{\mathcal{Q}}(D)$ form the set $Var^{x,+}(\Phi_{\mathcal{Q}}(D))$, and negative ones, the set $Var^{x,-}(\Phi_{\mathcal{Q}}(D))$. Instead of dealing with these variables at the lineage level, we will consider the values interventions assign to them (see (1) below).

Now, an *intervention on an instance D wrt. a Boolean query* $\mathcal{Q}$ can be represented by a truth assignment $\sigma : Var(\Phi_{\mathcal{Q}}(D)) \to \{0, 1\}$. The intervention $\mathcal{I}_\sigma$ on D associated to σ is the restriction of σ to those variables X_τ such that $\sigma(X_\tau) \neq \sigma_D(X_\tau)$. That is, $\mathcal{I}_\sigma$ represents only the changes of truth values, i.e. insertions or deletions of tuples into/from D. Then, the set of variables that change values wrt. D becomes the domain of $\mathcal{I}_\sigma$.

If we consider that assignments can be randomly and uniformly chosen, we obtain a probability space, with *outcome space* $\Omega = \{\sigma \mid \sigma : Var(\Phi_{\mathcal{Q}}(D)) \to \{0, 1\}\}$, and P the uniform distribution.

Next, we can use Pearl's notation for *interventions* [17], i.e. expressions of the form $do(X_\tau = x)$, where $x \in \{0, 1\}$. It denotes the intervention that makes X_τ take value 1 (i.e. becomes *true*) or 0 (i.e. becomes *false*), corresponding to inserting or deleting τ into/from a database instance, respectively. This notation can be generalized for multiple, simultaneous interventions, with the obvious meaning, to: $do(\bar{X}) = \bar{x}$, where $\bar{X} \subseteq Var(\Phi_{\mathcal{Q}}(D))$ is a list of m different variables, and $\bar{x} \in \{0, 1\}^m$. More technically, an intervention $do(\bar{X} = \bar{x})$ becomes an *event* (a subset of Ω): $do(\bar{X} = \bar{x}) := \{\sigma \in \Omega \mid (\sigma(X))_{X \in \bar{X}} = \bar{x}\}$.

Query $\mathcal{Q}$ can be seen as a Bernouilli *random variable*: $\mathcal{Q} : \Omega \to \{0, 1\}$, defined by $\mathcal{Q}(\sigma) = 1$ iff $\sigma \models \Phi_{\mathcal{Q}}$. Accordingly, for $y \in \{0, 1\}$, we may consider the event "$\mathcal{Q} = y$" $:= \{\sigma \in \Omega \mid \mathcal{Q}(\sigma) = y\}$. Furthermore, we obtain properly defined conditional probabilities of the form $P(\mathcal{Q} = y \mid do(\bar{X} = \bar{x}))$.

For a tuple τ and a value v for X_τ, we can compute the so-called *interventional conditional expectation* of (the truth value of) $\mathcal{Q}$, namely: $E(\mathcal{Q} \mid do(X_\tau = v)) = P(\mathcal{Q} = 1 \mid do(X_\tau = v))$.

In database causality, some tuples are endogenous and others exogenous, but our assignments σ on the set of variables do not make such a distinction. In the following, the expected value will be conditioned on the exogenous variables (those in $Var^x(\Phi_{\mathcal{Q}}(D))$) taking the value 1 when positive, and value 0 when negative. Accordingly, for an endogenous tuple τ, we redefine:

$$E(\mathcal{Q} \mid do(X_\tau = v)) \;:=\; E(\mathcal{Q} \mid \; do(X_\tau = v) \cap \bigcap_{X_{\tau'} \in Var^{x,+}(\Phi_{\mathcal{Q}}(D))} do(X_{\tau'} = 1) \cap \bigcap_{X_{\tau'} \in Var^{x,-}(\Phi_{\mathcal{Q}}(D))} do(X_{\tau'} = 0\,)). \quad (1)$$

In the following, we assume that conditional expectations are conditioned on exogenous tuples as in (1). We can now define a *measure of the causal effect of an intervention* [17] in terms of the average difference between the effects of two interventions.

Definition 1. Let D be an instance, $\mathcal{Q}$ a Boolean FO query, and $\tau \in D^n$. The *causal effect* of tuple τ on $\mathcal{Q}$ in D is:

$$\mathcal{E}^D_{\tau,\mathcal{Q}} \;:=\; E(\mathcal{Q} \mid do(X_\tau = v)) - E(\mathcal{Q} \mid do(X_\tau = 1 - v)), \quad (2)$$

where $v = 1$ if $\tau \in D$, and $v = 0$ if $\tau \notin D$. □

Intuitively, $\mathcal{E}^D_{\tau,\mathcal{Q}}$ shows how deleting an existing tuple from instance D or inserting an absent tuple into D affects the mean of the distribution of $\mathcal{Q}$ (the expectation taken in the space of random interventions on the remaining tuples).

Proposition 1. Let D be an instance, $\mathcal{Q}$ a Boolean FO query, and $\tau \in D^n$. It holds $\mathcal{E}^D_{\tau,\mathcal{Q}} \geq 0$. □

We will say that *a tuple has a causal effect on* $\mathcal{Q}$ in D when $\mathcal{E}^D_{\tau,\mathcal{Q}} > 0$. Causal effect allows us to compare the causal contribution of tuples: τ has *higher causal effect* on $\mathcal{Q}$ than τ' if $\mathcal{E}^D_{\tau,\mathcal{Q}} > \mathcal{E}_{\tau',\mathcal{Q}}$. Notice than the definition of causal effect does not require the query to be true in the given instance D. We claim that causal effect captures the notion of actual cause.

Proposition 2. Let D be an instance, $\mathcal{Q}$ a monotone Boolean FO query with $D \models \mathcal{Q}$, and $\tau \in D^n$. It holds: τ is an actual cause for $\mathcal{Q}$ in D iff τ has positive causal effect on $\mathcal{Q}$ in D. □

Example 3. (ex. 1 cont.) We can compute the causal effects of tuples in D on the query $\mathcal{Q}$ asking if there is a path between a and b. Here, $\Phi_{\mathcal{Q}}(D) = X_{t_1} \vee (X_{t_2} \wedge X_{t_3}) \vee (X_{t_4} \wedge X_{t_5} \wedge X_{t_6})$. Assuming all tuples are endogenous, $\mathcal{E}^D_{t_1,\mathcal{Q}} = 0.65625$, $\mathcal{E}^D_{t_2,\mathcal{Q}} = \mathcal{E}^D_{t_3,\mathcal{Q}} = 0.21875$, and $\mathcal{E}^D_{t_4,\mathcal{Q}} = \mathcal{E}^D_{t_5,\mathcal{Q}} = \mathcal{E}^D_{t_6,\mathcal{Q}} = 0.09375$. □

The notion of causal effect can handle non-monotone queries.

Example 4. (ex. 2 cont.) Here, $\Phi_{\mathcal{Q}}(D) = (X_{R(a,b)} \wedge \neg X_{S(b)}) \vee (X_{R(v,b)} \wedge \neg X_{S(b)})$. If all tuples are endogenous, $\mathcal{E}^D_{R(a,b),\mathcal{Q}} = \mathcal{E}^D_{R(c,b),\mathcal{Q}} = 0.25$, and $\mathcal{E}^D_{\neg S(b),\mathcal{Q}} = 0.75$. □

Our next example shows that causal effect can be applied to queries with aggregation. First notice that, in order to compute the effect of intervention, we do not have to materialize and process the lineage. Each intervention specifies an instance to which the query can be posed and evaluated. This allows us to naturally extend causal effect to aggregate queries.

Example 5. Consider an instance D with a unary relation $R = \{450, 150, 100, -100\}$, an the Boolean query

$\mathcal{Q}$: `select true from R having sum(A) > 500`,

asking if the sum of values in R is greater than 500. This non-monotone query (tuple insertions may invalidate a previous answer) has answer *true*, with all numbers in R contributing to it, but with different causal effects: $\mathcal{E}^D_{R(450),\mathcal{Q}} = 0.625$, $\mathcal{E}^D_{R(150),\mathcal{Q}} = 0.375$, $\mathcal{E}^D_{R(100),\mathcal{Q}} = 0.125$, and $\mathcal{E}^D_{R(-100),\mathcal{Q}} = -0.125$. The negative effect of $R(-100)$ means the tuple has a negative causal impact on the query outcome, which is intuitive.

Now consider the query $\mathcal{Q}'$: `select AVG(A) from R`. Here, $\mathcal{E}^D_{R(450),\mathcal{Q}'} = 112.5$, $\mathcal{E}^D_{R(150),\mathcal{Q}'} = 37.5$, $\mathcal{E}^D_{R(100),\mathcal{Q}'} = 25$, and $\mathcal{E}^D_{R(-100),\mathcal{Q}'} = -25$. □

Finally, we point out that causal effect can be applied to Datalog queries, as that in Example 1, where we obtain: $\mathcal{E}^D_{t_1,\mathcal{Q}} = 0.65625$, $\mathcal{E}^D_{t_2,\mathcal{Q}} = \mathcal{E}^D_{t_3,\mathcal{Q}} = 0.21875$, and $\mathcal{E}^D_{t_2,\mathcal{Q}} = \mathcal{E}^D_{t_2,\mathcal{Q}} = \mathcal{E}^D_{t_2,\mathcal{Q}} = 0.09375$.

4. Causal Effect and Pearson Correlation

In Statistics, the Pearson's *correlation coefficient* is a measure of the linear dependence between two random variables X and Y. It is defined by $r_{X,Y} = \frac{Cov(X,Y)}{\sigma_X \sigma_Y}$, where $Cov(X,Y) := E((X - \mu_X)(Y - \mu_Y))$ is the covariance of X, Y, μ_X, μ_Y are the expected values of X, Y, and σ_X, σ_Y their standard deviations.

It turns out that there is a close numerical connection between casual effect as introduced above and Pearson's correlation coefficient. This follows from the fact that the probability of any propositional formula, so as its conditional probability on a given variable, is a multi-linear polynomial in its variables [10].

Proposition 3. Let D be an instance, $\mathcal{Q}$ a Boolean FO query. It holds: (a) If τ is endogenous and X_τ appears positively in $\Phi_\mathcal{Q}(D)$: $\mathcal{E}^D_{\tau,\mathcal{Q}} = r_{\mathcal{Q},X_\tau} \times \frac{\sigma_\mathcal{Q}}{\sigma_{X_\tau}}$. (b) If τ is endogenous and appears negatively in $\Phi_\mathcal{Q}(D)$: $\mathcal{E}^D_{\tau,\mathcal{Q}} = -r_{\mathcal{Q},X_\tau} \times \frac{\sigma_\mathcal{Q}}{\sigma_{X_\tau}}$. Here, $\mathcal{Q}$ and X_τ are treated as Bernouilli random variables on space Ω. □

Unlike causal effect, Pearson's correlation coefficient is a normalized real-valued measure. For monotone queries, it takes values between 0 and 1. Then, we may use this correlation coefficient as a measure of the normalized causal effect of a tuple on a query.

Example 6. (ex. 5 cont.) The Pearson's correlation coefficients between the variables $X_{R(n)}$ and the aggregate $\mathcal{Q}'$ as a random variable are: $r^D_{X_{R(450)},\mathcal{Q}'} = 0.9091373$, $r^D_{X_{R(150)},\mathcal{Q}'} = 0.3030458$, $r^D_{X_{R(100)},\mathcal{Q}'} = 0.2020305$, and $r^D_{X_{R(-100)},\mathcal{Q}'} = -0.2020305$. □

Causal effect accounts only for the "linear interaction" between a tuple and a query answer. More specifically, it computes the shift of the mean of a query answer due to inserting/deleting a tuple into/from a database (on the space of random interventions on the remaining tuples). However, inserting/deleting a tuple might change higher-order moments of the query answer distribution.

Causal effect can properly deal with FO queries (due to the multi-linearity of their lineages) and linear aggregate queries. To deal with non-linear aggregate queries, we plan to use information theoretic approaches to quantify causal influence [9].

5. Conclusions and Related Work

In [11] it is argued that people use something similar to the intuition behind degree of responsibility (in the sense of [7]) to ascribe responsibilities. In [23], it is pointed out that people take into account not only the number of changes required to make A a counterfactual cause for B, but also the number of ways to reach a situation where B counterfactually depends on A. In [12] it is claimed that, while causal responsibility (in the sense of [7]) does capture some natural intuitions, still alternative definitions might be more appropriate for some applications.

Not surprisingly, much research on causal responsibility can be found in law literature [22, 2]. However, in no numerical quantification has been proposed, except for the work of [2].

In [18], the notion of degree of causal contribution has been introduced in the context of databases. This notion is defined based on the number of contingency sets associated to a tuple and shown to be closely related to the proposal in [2] and confirms the intuition behind [23]. It is not difficult to show that the notion of causal effect as introduced in this paper generalizes that of [18].

References

[1] Abiteboul, S., Hull, R. and Vianu, V. *Foundations of Databases*. Addison-Wesley, 1995.

[2] Braham, M. and Van Hees, M. Degrees of Causation. *Erkenntnis*, 2009, 71:323-344.

[3] Buneman, P., Khanna, S. and Tan, W. C. Why and Where: A Characterization of Data Provenance. *Proc. ICDT*, 2001, pp. 316-330.

[4] Buneman, P. and Tan, W. C. Provenance in Databases. *Proc. SIGMOD*, 2007, pp. 1171-1173.

[5] Cibele, F., Gatterbauer, W., Immerman, N. and Meliou A. A Characterization of the Complexity of Resilience and Responsibility for Conjunctive Queries. *PVLDB*, 2016, 9(3).

[6] Cheney, J. Causality and the Semantics of Provenance. *Proc. of the Workshop on Developments in Computational Models*, 2010, pp. 63-74.

[7] Chockler, H. and Halpern, J. Y. Responsibility and Blame: A Structural-Model Approach. *Journal of Artificial Intelligence Research (JAIR)*, 2004, 22:93-115.

[8] Cui, Y., Widom, J. and Wiener, J. L. Tracing the Lineage of View Data in a Warehousing Environment. *ACM Transactions on Database Systems (TODS)*, 2000, 25(2):179-227.

[9] Janzing, D., Balduzzi, D., Grosse-Wentrup, M. and Schlkopf, B., Quantifying causal influences. *The Annals of Statistics* , 2013, 41(5):2324-2358.

[10] Suciu, D., Olteanu, D., Re, C. and Koch, C. *Probabilistic Databases.* Synthesis Lectures on Data Management, Morgan & Claypool, 2011.

[11] Gerstenberg, T. and Lagnado, D. Spreading the Blame: The Allocation of Responsibility Amongst Multiple Agents. *Cognition*, 2010, 115(1):166-171

[12] Halpern, J. Y. Cause, Responsibility and Blame: A Structural-Model Approach. *Law, Probability and Risk*, 2015, 14 (2): 91-118.

[13] Halpern, J. Y. and Pearl, J. Causes and Explanations: A Structural-Model Approach: Part 1. *The British Journal for the Philosophy of Science*, 2005, 56:843-887.

[14] Karvounarakis, G. Ives, Z. G. and Tannen, V. Querying Data Provenance. *Proc. SIGMOD*, 2010, pp. 951-962.

[15] Meliou, A., Gatterbauer, W. and Suciu, D. Bringing Provenance to its Full Potential Using Causal Reasoning. *Proc. Theory and Practice of Provenance (TaPP)*, 2011.

[16] Meliou, A., Gatterbauer, W. Moore, K. F. and Suciu, D. The Complexity of Causality and Responsibility for Query Answers and Non-Answers. *Proc. VLDB*, 2010, pp. 34-41.

[17] Pearl, J. *Causality: Models, Reasoning and Inference.* Second ed., Cambridge University Press, 2009a.

[18] Salimi, B. Query-Answer Causality in Databases and its Connections with Reverse Reasoning Tasks in Data and Knowledge Management, PhD thesis, 2015. Posted at: `http://people.scs.carleton.ca/~bertossi/papers/Official08.pdf`.

[19] Salimi, B. and Bertossi, L. From Causes for Database Queries to Repairs and Model-Based Diagnosis and Back. *Proc. ICDT*, 2015a, pp. 342-362.

[20] Salimi, B. and Bertossi, L. Query-Answer Causality in Databases: Abductive Diagnosis and View-Updates. *Proc. UAI'15 Causal Inference Workshop*. CEUR WS Proc. Vol-1504, 2015b.

[21] Salimi, B. and Bertossi, L. Causes for Query Answers from Databases, Datalog Abduction and View-Updates: The Presence of Integrity Constraints. *To appear in Proc. FLAIRS*, 2016.

[22] Wright, R. W. Once More Into the Bramble Bush: Duty, Causal Contribution, and the Extent of Legal Responsibility. *Vanderbilt Law Review*, 2001, 54(3):1071-1132.

[23] Zultan, R., Gerstenberg, T. and Lagnado, D. Finding Fault: Causality and Counterfactuals in Group Attributions. *Cognition*, 2013, 125(3):429-440.