Referential Integrity under Uncertain Data

Sebastian Link[0000−0002−1816−2863] and Ziheng Wei

School of Computer Science,
The University of Auckland, New Zealand
{s.link|z.wei}@auckland.ac.nz

Abstract. Together with domain and entity integrity, referential integrity embodies the integrity principles of information systems. While relational databases address applications for data that is certain, modern applications require the handling of uncertain data. In particular, the veracity of big data and the complex integration of data from heterogeneous sources leave referential integrity vulnerable. We apply possibility theory to introduce the class of possibilistic inclusion dependencies. We show that our class inherits good computational properties from relational inclusion dependencies. In particular, we show that the associated implication problem is PSPACE-complete, but fixed-parameter tractable in the input arity. Combined with possibilistic keys and functional dependencies, our framework makes it possible to quantify the degree of trust in entities and relationships.

Keywords: Computational complexity· Inclusion dependency· Possibility theory· Reasoning· Referential integrity

1 Introduction

Big data has given our community big opportunities and challenges. One of these challenges is to build information systems that accommodate different dimensions of big data, including its veracity. According to an IBM study, one in three managers distrust the data that they use to make decisions[1]. The ability to quantify the degree of uncertainty in data would enable us to found decision making on data that is perceived to be sufficiently trustworthy.

In [15, 17] the authors presented a design framework for relational databases with uncertain data. Based on possibility theory [10], records are assigned a discrete degree of possibility (p-degree) with which they occur in a relation. Intuitively, the p-degree quantifies the level of trust an organization is prepared to assign to a record. The assignment of p-degrees can be based on many factors, specific to applications and irrelevant for developing the framework. In addition, an integrity constraint is assigned a degree of certainty (c-degree) that quantifies to which records it applies. Intuitively, the higher the c-degree of a constraint the lower the minimum p-degree of records to which the constraint applies. For example, a constraint is assigned the highest c-degree to affect all records, and the lowest c-degree to affect only records with the highest p-degree.

[1] http://www-01.ibm.com/software/data/bigdata/

The design frameworks of [17] were developed for possibilistic functional dependencies (pFDs) and possibilistic multivalued dependencies (pMVDs) [28].

The work from [17, 28] has therefore shown one way of extending Codd's principle of entity integrity from certain to uncertain data. The other principle is that of referential integrity, which ensures that references between data across tables are maintained soundly. So far, referential integrity has not been investigated for applications that accommodate uncertain data. Hence, we currently lack the ability to guarantee that uncertain data are appropriately referenced across tables. While it would be possible to record all data in one table, this would violate other principles of data management, such as the minimization of data redundancy and sources of inconsistency. This strongly motivates research on possibilistic variants of referential integrity constraints. The most expressive class of referential integrity constraints are inclusion dependencies (INDs), which subsume the important special case of foreign keys. It is therefore the main goal of this paper to introduce the class of possibilistic inclusion dependencies (pINDs) as a fundamental notion that extends the principle of referential integrity to the veracity dimension of big data. Previous classes of constraints that have been extended to the possibilistic setting were downward-closed. Here, a class of integrity constraints is downward-closed whenever every constraint in the class that is satisfied by a database instance will also be satisfied by every subset of that instance. Unfortunately, the class of inclusion dependencies is not downward-closed, which raises the challenge of introducing a suitable notion. In addition, we would like pINDs to cover traditional INDs as a special case, but inherit the computational properties of this special case. Hence, we are aiming for a notion that is adequate for uncertain data, while still being computationally attractive.

From a perspective of information systems engineering, the main contribution of our framework is quantifying the degree of trust in entities and relationships. Technically, we can summarize the contributions of the current work as follows.

- We introduce the class of possibilistic inclusion dependencies as a notion fundamental to extending the principle of referential integrity to the veracity of big data, and quantifying the degree of trust in relationships between data elements.
- While pINDs capture traditional inclusion dependencies as the special case where only one degree of uncertainty is permitted, we show that pINDs still inherit the good computational behavior of this special case. More specifically, we establish an algorithm that decides the implication problem in deterministic quadratic space. While we show that the implication problem is PSPACE-complete, it is also fixed-parameter tractable in the arity.

Organization. Section 2 introduces our running application scenario. We discuss related work in Section 3. We define a possibilistic data model in Section 4. We propose the class of pINDs in Section 5. Section 6 establishes the computational properties for our new class of pINDs. Section 7 concludes and comments briefly on future work.

2 Application Scenario

As a running example consider the following database schema that catalogs which parts are available from which supplier at what price.

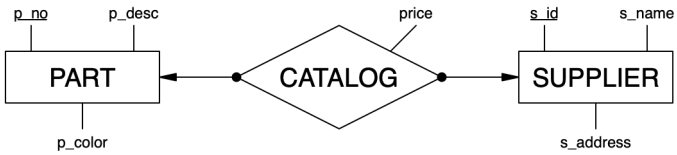

Fig. 1: Entity-relationship diagram for application scenario

- PART={p_no,p_desc,p_color} with key {p_no}
- SUPPLIER={s_id,s_name,s_address} with key {s_id}
- CATALOG={p_no,s_id,price} with key {p_no,s_id} and foreign keys
  - [p_no] $\subseteq$ PART[p_no] and
  - [s_id] $\subseteq$ SUPPLIER[s_id] .

The corresponding Entity-Relationship diagram is shown in Figure 1. Table 1 shows the result of integrating data from three legacy systems of the organization.

Table 1: Data integrated from legacy systems

PART

p_no	*p_desc*	*p_color*
p1	lever	red
p2	knob	yellow
p3	disc	green

CATALOG

p_no	*s_id*	*price*
p1	s1	2 frogs
p1	s2	1 bat
p2	s1	2 toads
p3	s3	2 snails

SUPPLIER

s_id	*s_name*	*s_address*
s1	Rumpel	Witchery
s2	Pumpel	Wizyard

As we can see, the database satisfies the keys and foreign keys defined by the schema. However, the integrated database does not contain any information about the level of trust associated with the records, based on the sources they have been integrated from. As an example use case of the framework we are proposing, we will now illustrate how the integration process can embed information about the different degrees of uncertainty that might be associated with the records. Note that this is just one specific way of using our framework.

The data shown in Table 1 is the result of integrating three different legacy systems as given in Table 2. While entity integrity in the form of the keys on the schemata is valid on all tables, there are issues with referential integrity.

These issues are simply hidden away in the integrated data set, which lacks a representation of the degrees of trust we should associate with the data. In an attempt to overcome this challenge, we assign possibility degrees (p-degrees) to records. In this example, we do this in the following intuitive way: we assign the highest degree *universal* when a record appears in all three relations of the same relation schema, the second highest degree *common* when a record appears in two of the three relations of the same relation schema, the third highest degree *isolated* when a record only appears in one of the three relations of the same relation schema, and the bottom degree *impossible*

Table 2: Legacy databases

PART		
p_no	*p_desc*	*p_color*
p1	lever	red
p2	knob	yellow

CATALOG		
p_no	*s_id*	*price*
p1	s1	2 frogs
p1	s2	1 bat
p2	s1	2 toads

SUPPLIER		
s_id	*s_name*	*s_address*
s1	Rumpel	Witchery
s2	Pumpel	Wizyard

Legacy database 1

PART		
p_no	*p_desc*	*p_color*
p1	lever	red
p3	disc	green

CATALOG		
p_no	*s_id*	*price*
p1	s1	2 frogs
p1	s2	1 bat
p3	s3	2 snails

SUPPLIER		
s_id	*s_name*	*s_address*
s1	Rumpel	Witchery
s2	Pumpel	Wizyard

Legacy database 2

PART		
p_no	*p_desc*	*p_color*
p1	lever	red
p3	disc	green

CATALOG		
p_no	*s_id*	*price*
p1	s1	2 frogs
p2	s1	2 toads

SUPPLIER		
s_id	*s_name*	*s_address*
s1	Rumpel	Witchery

Legacy database 3

Table 3: Data integrated from legacy systems with information about uncertainty

PART			
p_no	*p_desc*	*p_color*	**trust**
p1	lever	red	**universal**
p2	knob	yellow	**common**
p3	disc	green	**isolated**

CATALOG			
p_no	*s_id*	*price*	**trust**
p1	s1	2 frogs	**universal**
p1	s2	1 bat	**common**
p2	s1	2 toads	**common**
p3	s3	2 snails	**isolated**

SUPPLIER			
s_id	*s_name*	*s_address*	**trust**
s1	Rumpel	Witchery	**universal**
s2	Pumpel	Wizyard	**common**

when the record does not occur in any relation. In relational databases, the closed world assumption states that any record that is not explicitly listed in a relation is not part of it. The bottom p-degree *impossible* extends the closed world assumption to possibilistic databases, since it is assigned to every record that does not occur in the possibilistic database instance. Table 3 shows the integrated database instance inclusive of the levels of trust associated with the various records.

P-degrees quantify the level of trust we associate with records. There are different methods to assign such degrees. For example, each of the legacy systems may have some associated level of trust, and we simply assign the highest degree of trust among the systems in which the tuple occurs. Another method may assign p-degrees according to the recency of tuples.

Apart from quantifying uncertainty, p-degrees enable us to assign degrees of certainty to integrity constraints. For example, we can assign the highest degree of certainty to a constraint when it holds on the set of all records that are higher than the bottom p-degree, and the bottom degree of certainty when the constraint does not even hold on the set of records that have the highest p-degree. For instance, the key {p_no} holds with the highest degree of certainty on PART, and the key {s_id} holds with the highest degree of certainty on SUPPLIER. Similarly, the key {p_no,s_id} holds with the highest degree of certainty on CATALOG.

The foreign key $[p\_no] \subseteq$ PART$[p_no]$ on CATALOG holds on the database that only considers records with the *universal* degree of trust, but not on the database that considers records with the *universal* or *common* degree of trust. Hence, it can only be assigned the second lowest degree of certainty.

Similarly, the foreign key $[s\_id] \subseteq$ SUPPLIER$[s_id]$ on CATALOG holds on the database that considers only records with the *universal* degree of trust, and on the database that considers records with the *universal* or *common* degree of trust, but not on the database that considers records with the *universal*, *common* or *isolated* degree of trust. Hence, it can be assigned the second highest degree of certainty.

Denoting the degrees of certainty in this example by $\beta_1 > \beta_2 > \beta_3 > \beta_4$, we can revise our classical database schema as follows:

- PART={p_no,p_desc,p_color} with p-key $(\{p\_no\}, \beta_1)$
- SUPPLIER={s_id,s_name,s_address} with p-key $(\{s\_id\}, \beta_1)$
- CATALOG={p_no,s_id,price} with p-key $(\{p\_no, s\_id\}, \beta_1)$ and foreign keys
  - $([p\_no] \subseteq$ PART$[p_no], \beta_3)$ and
  - $([s\_id] \subseteq$ SUPPLIER$[s_id], \beta_2)$.

Figure 2 shows a corresponding Entity-Relationship diagram. Here, we augment some of the edges with indices of certainty degrees that apply to either attributes that form a key or directed edges that represent a foreign key. For example, we have attached the index 1 to attributes *p_no* of PART and *s_id* of SUPPLIER to indicate that they form a p-key for these entity types of c-degree β_1. Similarly, the label 2 of the edge from CATALOG to SUPPLIER represents the possibilistic foreign key $([s\_id] \subseteq$ SUPPLIER$[s_id], \beta_2)$, and the label 3 of the edge from CATALOG to PART represents the possibilistic foreign key $([p\_no] \subseteq$ PART$[p_no], \beta_3)$.

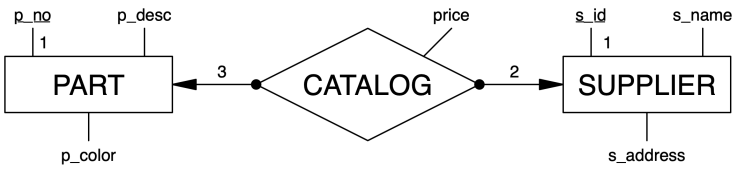

Fig. 2: Entity-relationship diagram representing information about uncertainty

The main aim of this paper is to define possibilistic inclusion dependencies, and to establish axiomatic and algorithmics solutions for their associated implication problem. These provide a foundation for quantifying the levels of trust in data, and hence also for resilience of decision-making in the presence of uncertain data.

3 Related Work

We review results on inclusion dependencies in relational databases, work on relaxed notions of inclusion dependencies, and their impact on data quality in general.

3.1 Classical Inclusion Dependencies

The implication problem is one of the core reasoning problems that is associated with any class of data dependencies [2, 25]. Solutions to this problem provide us with a complete understanding of how the data dependencies in this class interact, but also allow us to minimize the overhead spent on enforcing those dependencies that business analysts and data stewards have selected for modeling the integrity of their database. Indeed, if some dependency is implied by a set of dependencies that is valid on a given database instance, then we know that this dependency is also valid - which means we have already validated it implicitly. Vice versa, if a meaningful dependency is not implied, then we do need to validate it after an update occurs. Typically, solutions to the implication problem also allow us to exhaust all possibilities for optimizing the performance of operations on our data. For example, in order to apply some dependency during query optimization it may suffice to check whether it is implied by a given set of dependencies. Similarly, if we observe some inconsistencies with respect to an implied data dependency, we can conclude that there must be an inconsistency with respect to some data dependency that we are meant to enforce.

For these and other reasons, the implication problem for inclusion dependencies has been studied in many data models, in particular first for the relational model of data. Indeed, Casanova et al. [5] showed that finite and unrestricted implication problem coincide for the class of inclusion dependencies, the problem is PSPACE-complete, and enjoys a binary axiomatization. We will extend these results to the possibilistic case in this article. Here, the extension refers to an arbitrary finite scale of possibility degrees where the relational model occurs as the special case where only two possibility degrees are given, namely a top and a bottom degree. In fact, tuples in the current database instance are assigned the top degree, and tuples that are not in the current database instance are assigned the bottom degree.

3.2 Approximate Inclusion Dependencies

In practice it is often difficult to avoid integrity issues completely. For that purpose, more robust notions of constraints are often useful. These permit violations of the constraints up to some degree. For inclusion dependencies, in particular, different kinds of relaxed notions have been considered. An intuitive approximation is given by upper bounds on the proportion of tuples that need removal from the referencing table to satisfy the given inclusion dependency [18]. In addition, sets of (approximate) inclusion dependencies can also be approximated. Intuitively this makes sense for large sets of constraints that can often not be maintained efficiently, or where some of the constraints are not meaningful [20]. Missing values, often represented in the form of null markers, also cause uncertainty in databases. In fact, SQL supports simple and partial semantics for foreign keys on databases with null markers [12]. Under simple semantics tuples with null markers on some foreign key attributes do not require a match in the referenced table, while partial semantics still requires partial matches. Similarly, possibilistic inclusion dependencies also relax the requirement to hold on the entire instance. However, the scope where they need to hold is precisely given by the dual relationship of their associated possibility and certainty degrees.

3.3 Data Quality and Inclusion Dependencies

More generally, inclusion dependencies and their relaxed notions control referential integrity, with important consequences for the quality of data [22, 24, 29] and schema evolution [8, 26]. For example, data quality problems can be controlled by the use of conditional inclusion dependencies [19] that enable users to customize referential integrity to specific patterns of data. Similarly, the use of smart data samples that show the violation of constraints can draw the attention of human experts to data quality problems [21]. The combination of sampling with the discovery of constraints [11] makes it possible to discover meaningful constraints and data quality problems in unison [27]. These approaches offer opportunities for the application of possibilistic inclusion dependencies in the future. Interestingly, our framework of possibility theory has been used to approach the problem of cleaning data from a different perspective [13]. In that perspective, it is not the data that is viewed to be dirty, but it is the degree of trust in the data that is viewed dirty instead. The problem then is to minimally change the p-degrees associated with tuples in order to satisfy the given possibilistic constraints [13].

3.4 Other Classes of Possibilistic Constraints and Approaches to Uncertainty

Our possibilistic framework has been applied to advance entity integrity for uncertain data using classes of constraints such as keys [3], cardinality constraints [23], functional dependencies [15, 16], and multivalued dependencies [28]. the current article is thus the first to extend the framework towards advancing referential integrity for uncertain data.

Recently, primitive data types in OCL/UML have been extended to model the uncertainty of physical measurements or user estimates [4], and also proposed an algebra of operations to propagate them to complex types.

4 Possibilistic Databases

Previous work introduced the model of uncertain data for single relations [15, 17, 28]. Since our primary interest in the current article is on referential integrity, we will extend the model to actual database schemata and instances, since referential integrity constraints express relationships across different tables.

A relation schema, usually denoted by R, is a finite non-empty set of *attributes*. Each attribute $A \in R$ has a *domain* $dom(A)$ of values. A *tuple* t over R is an element of the Cartesian product $\prod_{A \in R} dom(A)$ of the attributes' domains. For $X \subseteq R$ we denote by $t(X)$ the *projection* of t on X. A *relation* over R is a finite set r of tuples over R. A database schema, usually denoted by D, is a finite non-empty set of relation schemata. A database over D, usually denoted by *db*, assigns to each relation schema $R \in D$ a relation r over R.

Our running example uses the database schema SUPPLY={PART, SUPPLIER, CATALOG} with relation schemata PART={p_no, p_desc, p_color}, SUPPLIER={s_id, s_name, s_address}, and CATALOG={p_no, s_id, price}.

We define possibilistic relations as relations where each tuple is associated with some confidence. The confidence of a tuple expresses up to which degree of possibility

a tuple occurs in a relation. Formally, we model the confidence as a *scale of possibility*, that is, a finite, strictly linear order $\mathcal{S} = (S, <_k)$ with $k+1$ elements where k is some positive integer, which we denote by $\alpha_1 >_k \cdots >_k \alpha_k >_k \alpha_{k+1}$, and whose elements $\alpha_i \in S$ we call *possibility degrees* (p-degrees). We sometimes simply write $<_k$ to refer to $\mathcal{S} = (S, <_k)$, and omit the subscript k from $<_k$ when it is fixed. The top p-degree α_1 is reserved for tuples that are 'fully possible' to occur in a relation, while the bottom p-degree α_{k+1} is reserved for tuples that are 'impossible' to occur in the relation at the moment. The use of the bottom p-degree α_{k+1} in our model is the counterpart of the classical closed world assumption. Humans like to use simple scales in everyday life, for instance to communicate, compare, or rank. Simple usually means to classify items qualitatively, rather than quantitatively by putting a precise value on it. Note that classical relations use a scale with two elements, that is, where $k = 1$.

In our running example, we use four different p-degrees that we label $\alpha_1 =$ *universal* for the top degree, $\alpha_2 =$ *common*, $\alpha_3 =$ *isolated*, and $\alpha_4 =$ *absent* for the bottom degree. A tuple is assigned p-degree α_i when it occurs in $4 - i$ legacy instances. For simplicity, we will use the same linear order on all relation schemata. We can also use different orders, but these can either be fused or more involved definitions can be given for our possibilistic referential constraints.

Formally, a *possibilistic relation schema* (p-schema) $(R, <_k)$ consists of a relation schema R and a possibility scale $<_k$. A *possibilistic relation* (p-relation) over $(R, <_k)$ consists of a relation r over R, together with a function Poss_r that maps each tuple $t \in r$ to a p-degree $\mathit{Poss}_r(t)$ in the possibility scale $<_k$. Sometimes, we simply refer to a p-relation (r, Poss_r) by r, assuming that Poss_r has been fixed. For example, Table 3 shows p-relations (r, Poss_r) over $(\text{PART}, <_3)$, $(\text{SUPPLIER}, <_3)$, and $(\text{CATALOG}, <_3)$ where $<_3 =$ *universal* $>_3$ *common* $>_3$ *isolated* $>_3$ *absent*.

A *possibilistic database schema* (pdb-schema) $(D, <_k)$ consists of a set D of relation schemata R, each of which forms a p-schema $(R, <_k)$. A *possibilistic database* (pdb) over $(D, <_k)$, usually denoted by $pdb = (db, \mathit{Poss})$, assigns to each p-schema $(R, <_k)$ of $(D, <_k)$ a p-relation (r, Poss_r). Again, Table 3 shows a pdb over pdb-schema $(\text{SUPPLY}, <_3)$ with $<_3 =$ *universal* $>_3$ *common* $>_3$ *isolated* $>_3$ *absent*.

Possibilistic databases enjoy a well-founded semantics in terms of possible worlds. In fact, every possible world is itself a classical database. For $i = 1, \ldots, k$ let $db_i = \{r_i \mid r \in db\}$ denote the database that consists of all tuples in db that have a p-degree of at least α_i, that is, $r_i = \{t \in r \mid \mathit{Poss}_r(t) \geq \alpha_i\}$ for all p-relations (r, Poss_r). Indeed, we have $r_1 \subseteq r_2 \subseteq \cdots \subseteq r_k$ for all of the p-relations (r, Poss_r) that constitute pdb. Hence, the p-degree associated with the world db_i is α_i. In particular, db_{k+1} is not a possible world since it includes tuples that are *impossible* to occur. Vice versa, the possibility $\mathit{Poss}_r(t)$ of a tuple $t \in r$ is the possibility of the smallest possible world in which t occurs. If $t \notin db_k$, then $\mathit{Poss}_r(t) = \alpha_{k+1}$. The top p-degree α_1 takes on a distinguished role: every tuple that is 'fully possible' occurs in every possible world - and is thus - 'fully certain'. This confirms our intuition that pdbs subsume databases (of fully certain tuples) as a special case. Table 4 shows the possible worlds of databases db_1, db_2, and db_3 of our running example.

Table 4: Chain of Possible Database Worlds

PART			CATALOG			SUPPLIER		
p_no	*p_desc*	*p_color*	*p_no*	*s_id*	*price*	*s_id*	*s_name*	*s_address*
p1	lever	red	p1	s1	2 frogs	s1	Rumpel	Witchery

Possible World $db1$

PART			CATALOG			SUPPLIER		
p_no	*p_desc*	*p_color*	*p_no*	*s_id*	*price*	*s_id*	*s_name*	*s_address*
p1	lever	red	p1	s1	2 frogs	s1	Rumpel	Witchery
p3	disc	green	p1	s2	1 bat	s2	Pumpel	Wizyard
			p2	s1	2 toads			

Possible World $db2$

PART			CATALOG			SUPPLIER		
p_no	*p_desc*	*p_color*	*p_no*	*s_id*	*price*	*s_id*	*s_name*	*s_address*
p1	lever	red	p1	s1	2 frogs	s1	Rumpel	Witchery
p3	disc	green	p1	s2	1 bat	s2	Pumpel	Wizyard
			p2	s1	2 toads			
			p3	s3	2 snails			

Possible World $db3$

5 Possibilistic Inclusion Dependencies

We recall the concepts of possibilistic keys and possibilistic functional dependencies from previous work [3, 15]. These form primary mechanisms to address entity integrity for uncertain data. We then introduce the new concept of possibilistic inclusion dependencies as the primary mechanism to address referential integrity for uncertain data.

An FD $X \rightarrow Y$ is satisfied by a relation r whenever every pair of tuples in r that have matching values on all the attributes in X have also matching values on all the attributes in Y [2, 25]. If $X \cup Y = R$, we call X a *key* because this case entails that there are no different tuples that match on X. For example, the FD *p_no*, *s_id* $\rightarrow$ *price* is satisfied by all the relations over CATALOG in Table 2, but the FD *s_id* $\rightarrow$ *price* is only satisfied by the relations over CATALOG in the second legacy database of Table 2. In particular, $\{p\_no, s\_id\}$ is a key but $\{s\_id\}$ is not a key.

For a given FD σ, the marginal certainty with which σ holds in a p-relation corresponds to the p-degree of the smallest possible world in which σ is violated. Therefore, dually to a scale $\mathcal{S}$ of p-degrees for tuples we use a scale $\mathcal{S}^T$ of certainty degrees (c-degrees) for constraints. We use positive integers as indices of the Greek letter β to denote c-degrees. Formally, the duality between p-degrees in $\mathcal{S}$ and c-degrees in $\mathcal{S}^T$ is defined by the mapping $\alpha_i \mapsto \beta_{k+2-i}$, for $i = 1, \ldots, k+1$. Since the impossible world r_{k+1} violates every FD, the marginal certainty $C_{(r,Poss_r)}(\sigma)$ with which the FD σ holds on the p-relation $(r, Poss_r)$ is the c-degree β_{k+2-i} for the smallest world r_i in which σ is violated. In particular, if r_k satisfies σ, then $C_{(r,Poss_r)}(\sigma) = \beta_1$.

We can now define the syntax and semantics of pFDs. A pFD over a p-schema $(R, \mathcal{S})$ is an expression $(X \rightarrow Y, \beta)$ where $X, Y \subseteq R$ and $\beta \in \mathcal{S}^T$. A p-relation $(r, Poss_r)$ over $(R, \mathcal{S})$ *satisfies* the pFD (σ, β) if and only if $C_{(r,Poss_r)}(\sigma) \geq \beta$. In our running example we use $\beta_1 >_3^S \beta_2 >_3^S \beta_3 >_3^S \beta_4$, with the interpretations of *certain* for

β_1, *quite certain* for β_2, *kind of certain* for β_3 and *not certain at all* for β_4. The marginal certainty of the FD *p_no*, *s_id* $\rightarrow$ *price* for the p-relation over CATALOG in Table 3 is *certain*, since the FD holds even in the largest possible world having p-degree α_3. The FD *s_id* $\rightarrow$ *price* is *kind of certain* since the smallest possible world that violates it ($db2$) has p-degree α_2.

We denote by $R, S \in D$ relation schemata in a database schema D and by $X = [A_1, \ldots, A_n]$ and $Y = [B_1, \ldots, B_n]$ sequences of distinct attributes in R and S, respectively, such that for all $m = 1, \ldots, n$, $dom(A_m) = dom(B_m)$ holds. The expression $R[X] \subseteq S[Y]$ a called an *inclusion dependency* (IND) over D. A database db over D with relations r over R and s over S is said to satisfy the IND $R[X] \subseteq S[Y]$ over D if and only if for every tuple $t_r \in r$ there is some tuple $t_s \in s$ such that $t_r[X] = t_s[Y]$. In our running example, the expressions CATALOG$[p_no] \subseteq$ PART$[p_no]$ and CATALOG$[s_id] \subseteq$ SUPPLIER$[s_id]$ denote INDs over SUPPLY. According to Table 4, the first IND is satisfied by $db1$ and $db3$ but not by $db2$, while the second IND is satisfied by $db1$ and $db2$ but not by $db3$.

We will now introduce the new concept of possibilistic inclusion dependencies.

Definition 1. *Let $(D, <_k)$ denote a pdb-schema and let $R[X] \subseteq S[Y]$ denote an IND over D. For $i \in \{1, \ldots, k+1\}$, we call the expression $(R[X] \subseteq S[Y], \beta_i)$ a* possibilistic inclusion dependency *(pIND) over $(D, <_k)$. Let $pdb = (db, Poss)$ denote a pdb over $(D, <_k)$ such that $(r, Poss_r)$ and $(s, Poss_s)$ denote p-relations over $(R, <_k)$ and $(S, <_k)$, respectively. The marginal certainty $C_{pdb}(R[X] \subseteq S[Y])$ with which the IND $R[X] \subseteq S[Y]$ holds on $pdb = (db, Poss)$ is the c-degree β_{k+2-i} for the smallest world db_i in which $R[X] \subseteq S[Y]$ is violated. In particular, if db_k satisfies $R[X] \subseteq S[Y]$, then $C_{pdb}(R[X] \subseteq S[Y]) = \beta_1$. We say that pdb* satisfies the pIND *$(R[X] \subseteq S[Y], \beta_i)$, denoted by $\models_{pdb} (R[X] \subseteq S[Y], \beta_i)$, if and only if $C_{pdb}(R[X] \subseteq S[Y]) \geq_k^S \beta_i$.*

The pdb from Table 3 shows that the smallest possible world that violates the IND CATALOG$[p_no] \subseteq$ PART$[p_no]$ is *db2*. Consequently, the marginal certainty of CATALOG$[p_no] \subseteq$ PART$[p_no]$ is β_3. Similarly, the smallest possible world that violates the IND CATALOG$[s_id] \subseteq$ SUPPLIER$[s_id]$ is *db3*. Hence, the marginal certainty of CATALOG$[s_id] \subseteq$ SUPPLIER$[s_id]$ is β_2. We conclude that pdb satisfies the pINDs

- (CATALOG$[p_no] \subseteq$ PART$[p_no]), \beta_3)$ and
- (CATALOG$[s_id] \subseteq$ SUPPLIER$[s_id], \beta_2)$,

but satisfies none of the pINDs

- (CATALOG$[p_no] \subseteq$ PART$[p_no]), \beta_2)$
- (CATALOG$[s_id] \subseteq$ SUPPLIER$[s_id], \beta_1)$.

Following Definition 1, pINDs enjoy a possible world semantics. Indeed, for every pdb pdb over every pdb-schema $(D, <_k)$, and for every $i = 1, \ldots, k$, we have that $\models_{pdb}$ $(R[X] \subseteq S[Y], \beta_i)$ if and only if $\models_{db_j} R[X] \subseteq S[Y]$ holds for all $j = 1, \ldots, k+1-i$.

An important difference to pFDs is that the equivalence requires us to check all possible worlds from $j = 1, \ldots, k+1-i$, while pFDs only require us to check the largest possible world db_{k+1-i}. The reason for the latter is that FDs are closed downwards in

the sense that every FD that is satisfied by a relation will also be satisfied by any sub-relation of the relation. This is not true for inclusion dependencies, which is why we specifically need to require that property in our semantics. This requirement, however, is very natural since values of tuples associated with some p-degree α should never reference tuples that are associated with a p-degree lower than α. Hence, this requirement is a natural extension of Codd's principle of referential integrity to uncertain data.

6 Reasoning about Possibilistic Inclusion Dependencies

The significance of (p)INDs results from their applicability to the most fundamental processing tasks for (uncertain) data. For example, we need to validate that all INDs that govern our data are still satisfied after updates are processed. This validation should impose a minimum overhead in resources. Computing a minimal set of INDs that *imply* all the INDs that govern the data makes it possible to minimize resources. For queries we want to generate a query plan that is likely to return the answer set as efficiently as possible. In attempting to find an optimal query plan, we may need to check whether some candidate IND holds on the given database. Deciding whether this candidate is implied by the set of INDs that are enforced on the data, the resources required to validate the candidate are minimized. Hence, INDs are useful when they can be reasoned about efficiently. We will show that our definition of pINDs cannot only express referential integrity for uncertain data, but also inherits the good computational behaviour from the well-known special case where $k = 1$.

6.1 Correspondence to INDs

We establish a correspondence between instances of the implication problems for pINDs and INDs. Let $\Sigma \cup \{\varphi\}$ denote a set of pINDs over a pdb-schema $(D, <_k)$. We say that Σ *implies* φ, denoted by $\Sigma \models \varphi$, if every pdb $(db, Poss)$ over $(D, <_k)$ that satisfies every pIND in Σ also satisfies φ.

Example 1. Let Σ consist of the pINDs $(\text{CATALOG}[p\_no] \subseteq \text{PART}[p\_no]), \beta_3)$ and $(\text{CATALOG}[s\_id] \subseteq \text{SUPPLIER}[s\_id], \beta_2)$. Let φ_1 denote the pIND $(\text{CATALOG}[p\_no] \subseteq \text{PART}[p\_no]), \beta_2)$ and let φ_2 denote the pIND $(\text{CATALOG}[s\_id] \subseteq \text{SUPPLIER}[s\_id], \beta_1)$. Then Σ implies neither φ_1 nor φ_2. Indeed, the p-database from Table 3 satisfies the two pINDs in Σ but satisfies neither φ_1 nor φ_2.

For a set Σ of pINDs on some pdb-schema $(D, <_k)$ and c-degree $\beta > \beta_{k+1}$, let $\Sigma_\beta = \{\sigma \mid (\sigma, \beta') \in \Sigma \text{ and } \beta' \geq \beta\}$ be the β-*cut* of Σ. The strength of our framework is engraved in the following result. It says that a pIND (σ, β) with c-degree β is implied by a set Σ of pINDs if and only if the IND σ is implied by the β-cut Σ_β of Σ.

Theorem 1. *[β-cuts] Let $\Sigma \cup \{(\varphi, \beta)\}$ denote a set of pINDs over p-db schema $(D, <_k)$ and let $\beta > \beta_{k+1}$. Then $\Sigma \models (\varphi, \beta)$ if and only if $\Sigma_\beta \models \varphi$.*

Proof. Let $\beta = \beta_i$ for some $1 \leq i \leq k$.

If $\Sigma \not\models (\varphi, \beta)$, then there is some p-db over $(D, <_k)$ that satisfies all pINDs in Σ but does not satisfy (φ, β). By definition there must exist a smallest possible world that satisfies all INDs in Σ_β but does not satisfy φ.

Vice versa, let db denote a database instance over D such that db satisfies all INDs in Σ_β but violates φ. For $\varphi = R[X] \subseteq S[Y]$ there must exist some tuple $t_r \in r$ such that for all tuples $t_s \in s$ we have $t_r[X] \neq t_s[Y]$. For $\beta = \beta_i$ with $1 \leq i \leq k$ we assign to $t_r \in r$ the p-degree α_{k+1-i} and to all other tuples in db the p-degree α_1. Then the resulting pdb (db, Poss) will satisfy Σ and violate (φ, β). □

The following example illustrates Theorem 1.

Example 2. For Σ, φ_1, and φ_2 from Example 1 we know that Σ does neither imply φ_1 nor φ_2. This is evident from the pdb in Table 3. Indeed, the smallest possible world which satisfies Σ_{β_2} and violates CATALOG$[p_no] \subseteq$ PART$[p_no]$ is the possible world $db2$. Vice versa, if we take the possible world $db2$ and assign the p-degree α_2 to the tuple (p2, s1, 2 toads) over CATALOG, and assign the p-degree α_1 to all the other tuples, then the resulting pdb will satisfy Σ but violate φ_1.

A similar argument can be made for Σ and φ_2, with the only difference being that the possible world $db3$ satisfies Σ_{β_1} and violates CATALOG$[s_id] \subseteq$ SUPPLIER$[s_id]$ due to the tuple (p3, s3, 2 snails). Vice versa, assigning p-degree α_3 to this tuple in $db3$ and assigning p-degree α_1 to any other tuple in this database, results in a pdb that satisfies Σ and violates φ_2.

6.2 Algorithmic Characterization

We would like an algorithm that can decide the implication problem for pINDs efficiently. Using Theorem 1, we can extend the decision procedure for classical INDs to decide the implication problem for pINDs. Algorithm 1 directly returns an affirmative answer whenever the candidate pIND φ has bottom c-degree β_{k+1}. Otherwise, it uses the chase procedure for INDs applied to the β-cut Σ_β. Algorithm 1 runs in nondeterministic linear space. According to Savitch (PSPACE = NPSPACE) the algorithm can be implemented to run in determinstic quadratic space.

Corollary 1. *Algorithm 1 decides pIND implication in deterministic quadratic space.*

Proof. The correctness and complexity of Algorithm 1 follow from that of the algorithm for deciding INDs in the special case $k = 1$, and Theorem 1. □

Example 3. Let Σ consist of the following two pINDs over the extended pdb-schema SUPPLY that we have been using as a running example:

- (SALES$[s_id] \subseteq$ CATALOG$[s_id], \beta_1$)
- (CATALOG$[s_id] \subseteq$ SUPPLIER$[s_id], \beta_2$) .

We use φ to denote (SALES$[s_id] \subseteq$ CATALOG$[s_id], \beta_2$) and we use φ' to denote (SALES$[s_id] \subseteq$ CATALOG$[s_id], \beta_1$) as two candidate pINDs for which we wonder whether they are implied by Σ.

Let us apply Algorithm 1 to both inputs $\Sigma \cup \{\varphi\}$ and $\Sigma \cup \{\varphi'\}$. For neither of the two inputs does β represent the bottom c-degree β_4. For φ we obtain the β_2-cut Σ_{β_2} of Σ that consists of the two INDs:

Algorithm 1 pIND-chase

Require: Set $\Sigma \cup \{(R_a[A_1, \ldots, A_n] \subseteq R_b[B_1, \ldots, B_n], \beta)\}$ of pINDs over $(D, <_k)$
Ensure: Yes, if Σ implies $(R_a[A_1, \ldots, A_n] \subseteq R_b[B_1, \ldots, B_n], \beta)$, and No, otherwise
1: **if** $\beta = \beta_{k+1}$ **then return** ('Yes')
2: **end if**;
3: $\mathcal{E} := \{R_a[A_1, \ldots, A_n]\}$;
4: **repeat**
5: **if** $R_i[C_1, \ldots, C_n] \in \mathcal{E}$ and $R_i[C_1, \ldots, C_n] \subseteq R_j[D_1, \ldots, D_m]$ can be inferred from
6: Σ_β by a single application of $\mathcal{P}'$ **then**
7: $\mathcal{E} := \mathcal{E} \cup \{R_j[D_1, \ldots, D_m]\}$;
8: **end if**
9: **until** $R_b[B_1, \ldots, B_n] \in \mathcal{E}$ or no change possible
10: **if** $R_b[B_1, \ldots, B_n] \in \mathcal{E}$ **then return** 'Yes'
11: **else**
12: **return** ('No');
13: **end if**

- $\sigma_1 = \text{SALES}[s\_id] \subseteq \text{CATALOG}[s\_id]$
- $\sigma_2 = \text{CATALOG}[s\_id] \subseteq \text{SUPPLIER}[s\_id]$.

Starting with $\mathcal{E} = \{\text{SALES}[s\_id]\}$ and applying first σ_1 and then σ_2 we obtain $\mathcal{E} = \{\text{SALES}[s\_id], \text{CATALOG}[s\_id], \text{SUPPLIER}[s\_id]\}$, which means that Algorithm 1 returns an affirmative answer. Starting with $\mathcal{E}' = \{\text{SALES}[s\_id]\}$ and β_1-cut $\Sigma_{\beta_1} = \{\sigma_1\}$, we can only apply σ_1 to obtain $\mathcal{E}' = \{\text{SALES}[s\_id], \text{CATALOG}[s\_id]\}$. This means Algorithm 1 returns a negative answer since $\text{SUPPLIER}[s\_id] \notin \mathcal{E}'$.

6.3 PSPACE-completeness and Fixed-parameter Tractability

Corollary 1 shows that the implication problem of pINDs is in PSPACE. In the relational model the implication problem of INDs is also PSPACE-hard [5], and INDs form the special case of pINDs for $k = 1$. Hence, the implication problem of pINDs is also PSPACE-complete. Following [12] the implication problem for INDs is even fixed-parameter tractable (FPT) [9] in the arity of the input. That is, there is a deterministic algorithm that runs in polynomial time when the arity of the input is fixed.

Corollary 2. *The implication of pINDs is* PSPACE*-complete and* FPT *in their arity.*

It also follows from a result about INDs [7] that the implication problem for pINDs with bounded arity is NLOGSPACE-complete.

7 Conclusion and Future Work

Using possibility theory we have proposed a class of inclusion dependencies for uncertain data. Our proposal can express different degrees of certainty by which INDs hold. Since the degrees can be customized to the needs of data owners, our possibilistic INDs are able to enforce referential integrity according to the requirements of applications.

This should provide organizations with the ability to quantify the level of trust they have in their data and the relationships between them. This trust will facilitate more confident decision-making under uncertainty. Our proposal inherits the good computational behavior from its special case of certain data.

The research opens up several new questions. When studying the interaction of entity and referential integrity for uncertain data, and their extension to other data models with missing values, such as JSON. It will be important to extend conceptual, logical, and physical design approaches from certain to uncertain data, such as [6, 14]. While a headstart has been made [17], inclusion dependencies have not been taken into account yet. Another core reasoning problem is the discovery of possibilistic constraints for a given class that hold on a given possibilistic database. This problem has received much attention in the relational model [1], but not yet for models of uncertain data.

References

1. Ziawasch Abedjan, Lukasz Golab, Felix Naumann, and Thorsten Papenbrock. *Data Profiling*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2018.
2. Paolo Atzeni and Valeria De Antonellis. *Relational Database Theory*. Benjamin/Cummings, 1993.
3. Nishita Balamuralikrishna, Yingnan Jiang, Henning Koehler, Uwe Leck, Sebastian Link, and Henri Prade. Possibilistic keys. *Fuzzy Sets and Systems*, 376:1–36, 2019.
4. Manuel F. Bertoa, Loli Burgueño, Nathalie Moreno, and Antonio Vallecillo. Incorporating measurement uncertainty into OCL/UML primitive datatypes. *Softw. Syst. Model.*, 19(5):1163–1189, 2020.
5. Marco A. Casanova, Ronald Fagin, and Christos H. Papadimitriou. Inclusion dependencies and their interaction with functional dependencies. *J. Comput. Syst. Sci.*, 28(1):29–59, 1984.
6. Peter P. Chen. The entity-relationship model - toward a unified view of data. *ACM Trans. Database Syst.*, 1(1):9–36, 1976.
7. Stavros S. Cosmadakis, Paris C. Kanellakis, and Moshe Y. Vardi. Polynomial-time implication problems for unary inclusion dependencies. *J. ACM*, 37(1):15–46, 1990.
8. Konstantinos Dimolikas, Apostolos V. Zarras, and Panos Vassiliadis. A study on the effect of a table's involvement in foreign keys to its schema evolution. In Gillian Dobbie, Ulrich Frank, Gerti Kappel, Stephen W. Liddle, and Heinrich C. Mayr, editors, *Conceptual Modeling - 39th International Conference, ER 2020, Vienna, Austria, November 3-6, 2020, Proceedings*, volume 12400 of *Lecture Notes in Computer Science*, pages 456–470. Springer, 2020.
9. Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.
10. Didier Dubois and Henri Prade. Possibility theory. In Robert A. Meyers, editor, *Computational Complexity: Theory, Techniques, and Applications*, pages 2240–2252. Springer New York, 2012.
11. Falco Dürsch, Axel Stebner, Fabian Windheuser, Maxi Fischer, Tim Friedrich, Nils Strelow, Tobias Bleifuß, Hazar Harmouch, Lan Jiang, Thorsten Papenbrock, and Felix Naumann. Inclusion dependency discovery: An experimental evaluation of thirteen algorithms. In Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu, editors, *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 219–228, 2019.

12. Henning Köhler and Sebastian Link. Inclusion dependencies and their interaction with functional dependencies in SQL. *J. Comput. Syst. Sci.*, 85:104–131, 2017.
13. Henning Köhler and Sebastian Link. Possibilistic data cleaning. *IEEE Trans. Knowl. Data Eng.*, in press.
14. Mark Levene and Millist W. Vincent. Justification for inclusion dependency normal form. *IEEE Trans. Knowl. Data Eng.*, 12(2):281–291, 2000.
15. Sebastian Link and Henri Prade. Possibilistic functional dependencies and their relationship to possibility theory. *IEEE Trans. Fuzzy Systems*, 24(3):757–763, 2016.
16. Sebastian Link and Henri Prade. Relational database schema design for uncertain data. In Snehasis Mukhopadhyay, ChengXiang Zhai, Elisa Bertino, Fabio Crestani, Javed Mostafa, Jie Tang, Luo Si, Xiaofang Zhou, Yi Chang, Yunyao Li, and Parikshit Sondhi, editors, *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, pages 1211–1220. ACM, 2016.
17. Sebastian Link and Henri Prade. Relational database schema design for uncertain data. *Inf. Syst.*, 84:88–110, 2019.
18. Stéphane Lopes, Jean-Marc Petit, and Farouk Toumani. Discovering interesting inclusion dependencies: application to logical database tuning. *Inf. Syst.*, 27(1):1–19, 2002.
19. Shuai Ma, Wenfei Fan, and Loreto Bravo. Extending inclusion dependencies with conditions. *Theor. Comput. Sci.*, 515:64–95, 2014.
20. Fabien De Marchi and Jean-Marc Petit. Approximating a set of approximate inclusion dependencies. In *Intelligent Information Processing and Web Mining, Proceedings of the International IIS: IIPWM'05 Conference held in Gdansk, Poland, June 13-16, 2005*, pages 633–640, 2005.
21. Fabien De Marchi and Jean-Marc Petit. Semantic sampling of existing databases through informative armstrong databases. *Inf. Syst.*, 32(3):446–457, 2007.
22. Carlos Ordonez and Javier García-García. Referential integrity quality metrics. *Decis. Support Syst.*, 44(2):495–508, 2008.
23. Tania Roblot and Sebastian Link. Cardinality constraints and functional dependencies over possibilistic data. *Data Knowl. Eng.*, 117:339–358, 2018.
24. Shazia Wasim Sadiq, Tamraparni Dasu, Xin Luna Dong, Juliana Freire, Ihab F. Ilyas, Sebastian Link, Renée J. Miller, Felix Naumann, Xiaofang Zhou, and Divesh Srivastava. Data quality: The role of empiricism. *SIGMOD Rec.*, 46(4):35–43, 2017.
25. Bernhard Thalheim. *Dependencies in relational databases*, volume 126 of *Teubner-Texte zur Mathematik*. Teubner, 1991.
26. Panos Vassiliadis, Michail-Romanos Kolozoff, Maria Zerva, and Apostolos V. Zarras. Schema evolution and foreign keys: Birth, eviction, change and absence. In Heinrich C. Mayr, Giancarlo Guizzardi, Hui Ma, and Oscar Pastor, editors, *Conceptual Modeling - 36th International Conference, ER 2017, Valencia, Spain, November 6-9, 2017, Proceedings*, pages 106–119, 2017.
27. Ziheng Wei and Sebastian Link. DataProf: Semantic profiling for iterative data cleansing and business rule acquisition. In Gautam Das, Christopher M. Jermaine, and Philip A. Bernstein, editors, *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 1793–1796, 2018.
28. Ziheng Wei and Sebastian Link. A fourth normal form for uncertain data. In Paolo Giorgini and Barbara Weber, editors, *Advanced Information Systems Engineering - 31st International Conference, CAiSE 2019, Rome, Italy, June 3-7, 2019, Proceedings*, volume 11483 of *Lecture Notes in Computer Science*, pages 295–311. Springer, 2019.
29. Ruojing Zhang, Marta Indulska, and Shazia W. Sadiq. Discovering data quality problems - the case of repurposed data. *Bus. Inf. Syst. Eng.*, 61(5):575–593, 2019.