Keys with Probabilistic Intervals

Pieta Brown[1], Jeeva Ganesan[1], Henning Köhler[2], and Sebastian Link[1]

[1] Department of Computer Science, University of Auckland, New Zealand
{pieta.brown,j.ganesan,s.link}@auckland.ac.nz
[2] School of Engineering & Advanced Technology, Massey University, New Zealand
h.koehler@massey.ac.nz

Abstract. Probabilistic databases accommodate well the requirements of modern applications that produce large volumes of uncertain data from a variety of sources. We propose an expressive class of probabilistic keys which empowers users to specify lower and upper bounds on the marginal probabilities by which keys should hold in a data set of acceptable quality. Indeed, the bounds help organizations balance the consistency and completeness targets for their data quality. For this purpose, algorithms are established for an agile schema- and data-driven acquisition of the right lower and upper bounds in a given application domain, and for reasoning about these keys. The efficiency of our acquisition framework is demonstrated theoretically and experimentally.

Keywords: Data mining; Data semantics; Integrity Constraint; Probabilistic data; Requirements acquisition

1 Introduction

Background. Keys allow us to understand the structure and semantics of data. In relational databases, a key is a set of attributes that holds on a relation if no two different tuples in the relation have matching values on all the attributes of the key. The ability of keys to uniquely identify entities makes them invaluable in data processing and applications.

Motivation. Relational databases target applications with certain data, such as accounting and payroll. Modern applications, such as data integration and financial risk assessment produce large volumes of uncertain data from a variety of sources. For instance, RFID (radio frequency identification) can track movements of endangered species of animals, such as the Japanese Serow in central Honshu. Here it is sensible to apply probabilistic databases. Table 1 shows two probabilistic relations (p-relation), which are probability distributions over a finite set of possible worlds, each being a relation.

In requirements acquisition the goal is to specify all keys that apply to the application domain, and those keys only. This goal addresses the consistency and completeness dimensions of data quality. Here, consistency means to specify all meaningful keys in order to prevent the occurrence of inconsistent data, while completeness means not to specify any meaningless keys in order to capture any potential meaningful database instance. This situation is exemplified in Figure 1.

Table 1. Probabilistic relations r_1 and r_2

r_1 :

W_1 ($p_1 = 0.35$)		
rfid	*time*	*zone*
j1	2pm	z1
j1	3pm	z2
j2	4pm	z1
j3	2pm	z3

W_2 ($p_2 = 0.3$)		
rfid	*time*	*zone*
j1	2pm	z1
j3	2pm	z3
j1	5pm	z1

W_3 ($p_3 = 0.1$)		
rfid	*time*	*zone*
j1	2pm	z1
j1	5pm	z1
j4	2pm	z1

W_4 ($p_4 = 0.25$)		
rfid	*time*	*zone*
j1	2pm	z1
j1	5pm	z1
j4	2pm	z1
j1	2pm	z4

r_2 :

W_1 ($p_1 = 0.2$)		
rfid	*time*	*zone*
j1	2pm	z1
j2	3pm	z1

W_2 ($p_2 = 0.3$)		
rfid	*time*	*zone*
j1	2pm	z1
j2	3pm	z1
j1	4pm	z2
j3	2pm	z3

W_3 ($p_3 = 0.25$)		
rfid	*time*	*zone*
j1	2pm	z1
j3	2pm	z3
j1	5pm	z1

W_4 ($p_4 = 0.25$)		
rfid	*time*	*zone*
j1	2pm	z1
j1	5pm	z1
j1	2pm	z4
j4	2pm	z1

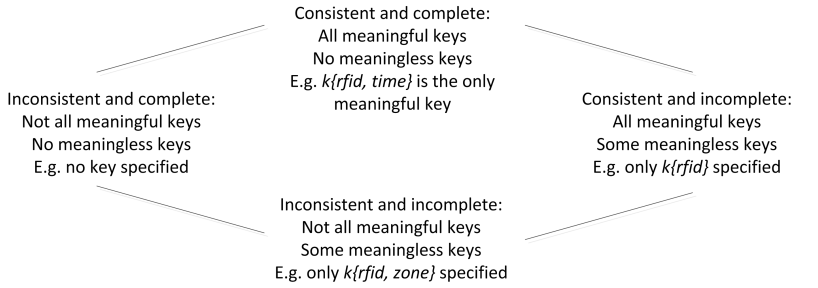


Fig. 1. Consistency and completeness dimensions as controlled by keys

In probabilistic databases, one may speak of a key when it holds in all possible worlds. That is to say that a key holds with marginal probability one, which means that the probabilities of the worlds in which the key holds add up to one. Due to the veracity of probabilistic data and the variety of sources the data originate from, one must not expect to satisfy the completeness criteria with this definition. Neither does such definition make sensible use of probabilities, as one would expect for probabilistic data. In our example, neither r_1 nor r_2 satisfy any non-trivial key with marginal probability one: The key $k\{\mathit{rfid}, \mathit{time}\}$ has marginal probability 0.75 in both r_1 and r_2, while $k\{\mathit{time}, \mathit{zone}\}$ has marginal probability 0.65 in r_1 and marginal probability 0.75 in r_2.

We propose *keys with probabilistic intervals*, or p-keys for short, which stipulate lower and upper bounds on the marginal probability by which a traditional key holds on probabilistic data. For example, we may specify the p-keys $k\{\mathit{rfid}, \mathit{time}\} \in (0.75, 1)$ and $k\{\mathit{time}, \mathit{zone}\} \in (0.65, 0.75)$. In particular, the ability to stipulate lower and upper bounds on the marginal probability of keys is useful for probabilistic data. The p-key $k\{\mathit{time}, \mathit{zone}\} \in (0.65, 0.75)$ reflects our observations that different serows may occur at the same time in the same zone at least with probability 0.25 and at most with probability 0.35. Our main motivation for p-keys is their ability to balance the consistency and completeness

Table 2. Two PC-tables that form an Armstrong PC-base for the p-keys of Figure 2

CD table

rfid	*time*	*zone*	W
j1	2pm	z1	$1, 2, 3, 4$
j1	3pm	z2	1
j2	4pm	z1	1
j3	2pm	z3	$1, 2$
j1	5pm	z1	$2, 3, 4$
j4	2pm	z1	$3, 4$
j1	2pm	z4	4

P table

W	$\mathcal{P}$
1	.35
2	.3
3	.1
4	.25

CD table

rfid	*time*	*zone*	W
j1	2pm	z1	$1, 2, 3, 4$
j2	3pm	z1	$1, 2$
j1	4pm	z2	2
j3	2pm	z3	$2, 3$
j1	5pm	z1	$3, 4$
j1	2pm	z4	4
j4	2pm	z1	4

P table

W	$\mathcal{P}$
1	.2
2	.3
3	.25
4	.25

targets for the quality of probabilistic data. Consistency means that for each key the specified lower (upper) bound on its marginal probability is not too high (low), and completeness means that for each key the specified lower (upper) bound is not too low (high). Once the bounds have been consolidated, p-keys can be utilized to control these data quality dimensions during updates. When new data arrives, p-keys can help detect anomalous patterns of data in the form of p-key violations. That is, automated warnings can be issued whenever data would not meet a desired lower or upper bound of some p-key. In a different showcase, p-keys can be used to infer probabilities that query answers are (non-)unique. In our example, we may wonder about the chance that different serows are in the same zone at the same time, indicating potential mating behavior. We may ask

`SELECT DISTINCT` *rfid* `FROM` TRACKING `WHERE` *zone*='z2' `AND` *time*='2pm' .

P-keys enable us to derive a minimum (maximum) probability of 0.65 (0.75) that a unique answer is returned, because different serows are in zone z2 at 2pm at least with probability 0.25 and at most with probability 0.35. These bounds can be inferred without accessing any data at all, only requiring that $k\{time, zone\}$ has a marginal probability between 0.65 and 0.75 on the given data.

Contributions. Our contributions can be summarized as follows. **Modeling:** We propose p-keys $kX \in (p, q)$ as a natural class of integrity constraints over uncertain data. Their main use is to help organizations balance consistency and completeness targets for the quality of their data, and to quantify bounds on the probability for (non-)unique query answers. **Reasoning:** While sets of p-keys can be unsatisfiable, we establish an efficient algorithm to decide satisfiability. The implication problem is to decide for a given set $\Sigma \cup \{\varphi\}$ of p-keys, whether every p-relation that satisfies all elements of Σ also satisfies φ. We characterize the implication problem of satisfiable sets of p-keys by a finite set of Horn rules, and a linear time decision algorithm. This enables organizations to reduce the overhead of managing p-keys to a minimal level necessary. **Summarization:** For the schema-driven acquisition of the right probabilistic intervals, we show how to perfectly summarize any given satisfiable set of p-keys as an Armstrong PC-base. An Armstrong PC-base consists of two PC-tables: One that

satisfies every key with the exact marginal probability that is the perceived best lower bound for the domain, and one that is the perceived best upper bound. Any flaws with these perceptions are explicitly pointed out: Either as unreasonably high lower bounds, or unreasonably low upper bounds. For example, Table 2 shows an Armstrong PC-base for the p-keys shown in Figure 2. In the CD table, the W column of a tuple shows the identifiers of possible worlds to which the tuple belongs. The P-table shows the probability distribution on the possible worlds. The first PC-table represents the p-relation r_1 and the second PC-table represents the p-relation r_2 from Table 1. While all p-keys that are implied by this p-key set are satisfied by both PC-tables, every non-implied p-key is violated by at least one PC-table. For example, the implied p-key $k\{time,zone\} \in (0.6, 0.8)$ is satisfied by the p-relations the tables represent, while the non-implied p-key $k\{time,zone\} \in (0.7, 0.75)$ is violated by r_1.

Discovery: For the data-driven acquisition of p-keys we compute the probabilistic interval of any key as the smallest and largest marginal probabilities across all given PC-tables. For example, given the two PC-tables in Table 2, our algorithm would discover the profile of p-keys in Figure 2. **Experiments:** Experiments show that our algorithms are efficient and scale linearly *in our acquisition framework*.

Organization. We discuss related work in Section 2. P-keys are introduced in Section 3. Computational problems are characterized in Section 4. The schema- and data-driven acquisition of p-keys is developed in Section 5. Experiment results are presented in Section 6. We conclude in Section 7.

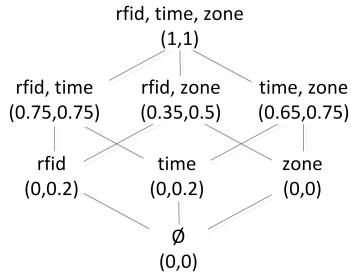


Fig. 2. P-key Profile of Table 2

2 Related Work

Poor data quality inhibits the transformation of data into value [23]. P-keys provide a well-founded, yet simple approach to balance the consistency and completeness targets for the quality of data. Keys are fundamental to most data models [3, 6, 9, 12, 15, 25, 26]. P-keys subsume keys from traditional relations [1, 2], covered by the special case where p-relations consist of one possible world only. There is substantial work on the discovery of "approximate" constraints, see [4, 16, 20] for recent surveys. Approximate means that not all tuples satisfy the given constraint, but exceptions are tolerable. P-keys are not approximate since they are either satisfied or violated by the given p-relation. Future work will investigate approximate p-keys. Possibilistic keys [11] are attributed some degree of certainty saying to which tuples they apply. Possibility theory is a qualitative approach, while probability theory is a quantitative approach to uncertainty. P-keys complements the qualitative approach to possibilistic keys from [11]. Keys

with probabilistic intervals extend our own work on keys with lower bounds only [3]. The extension causes significant differences. Keys with intervals are more expressive as upper bounds smaller than 1 can be specified, addressing better any consistency and completeness targets. Sets of keys with intervals may not be satisfiable by any p-relation, while every set of keys with only lower bounds is satisfiable. While implication and inference problems become more complex for intervals, we succeed in establishing linear time algorithms. While keys with only lower bounds enjoy representations by a single Armstrong PC-table, keys with intervals require generally two PC-tables. This is an interesting novelty for Armstrong databases for which more than one database instance have not been considered in previous research. We also generalize the discovery problem for keys with intervals to a collection of input instances, while only single input instances were considered in [3] for keys with lower bounds.

3 Keys with Probabilistic Intervals

We introduce our notion of keys with probabilistic intervals after some preliminaries on probabilistic databases.

A *relation schema* is a finite set R of attributes A. Each attribute A is associated with a domain $dom(A)$ of values. A tuple t over R is a function that assigns to each attribute A of R an element $t(A)$ from the domain $dom(A)$. A *relation* over R is a finite set of tuples over R. Relations over R are also called *possible worlds* of R here. An expression kX over R with $X \subseteq R$ is called a *key*. A key kX is said to hold in a possible world W of R, denoted by $W \models kX$, if and only if there no two tuples $t_1, t_2 \in W$ such that $t_1 \neq t_2$ and $t_1(X) = t_2(X)$. A *probabilistic relation* (p-relation) over R is a pair $r = (\mathcal{W}, P)$ of a finite non-empty set $\mathcal{W}$ of possible worlds over R and a probability distribution $P : \mathcal{W} \to (0, 1]$ such that $\sum_{W \in \mathcal{W}} P(W) = 1$ holds. Table 1 shows two p-relations over relation schema TRACKING=$\{$*rfid*,*time*,*zone*$\}$. World W_2 of r_1, for example, satisfies the keys $k\{$*rfid*, *time*$\}$ and $k\{$*zone*, *time*$\}$, but violates the key $k\{$*rfid*, *zone*$\}$. The *marginal probability* $m_{X,r}$ of a key kX in the p-relation r is the sum of the probabilities of those possible worlds in r which satisfy kX. We will now introduce the central notion of a key with probabilistic intervals.

Definition 1. *A* key with probabilistic intervals*, or* p-key *for short, over relation schema R is an expression $kX \in (l, u)$ where $X \subseteq R$, $l, u \in [0, 1]$, and $l \leq u$. The p-key $kX \in (l, u)$ over R is* satisfied by*, or said to* hold in*, the p-relation r over R if and only if the marginal probability $m_{X,r}$ of kX in r falls into the interval (l, u), that is, $l \leq m_{X,r} \leq u$.*

In our running example over relation schema TRACKING, the p-relation r_2 from Table 1 satisfies the p-keys $k\{$*rfid*, *time*$\} \in (0.7, 0.75)$ and $k\{$*time*, *zone*$\} \in (0.3, 0.8)$, but violates the p-keys $k\{$*rfid*, *time*$\} \in (0.7, 0.7)$ and $k\{$*time*, *zone*$\} \in (0.3, 0.65)$. The reasons are that $m_{\{rfid,time\},r_2} = m_{\{time,zone\},r_2} = 0.75$.

It is useful to separate a p-key into one key that stipulates the lower bound and one key that stipulates the upper bound. This allows users to focus on one

bound at a time, but also allows us to gain a better understanding of their interaction. A key with lower bound, or *l-key*, is of the form $kX \in (l, 1)$, and we write $kX_{\geq l}$. A key with upper bound, or *u-key*, is of the form $kX \in (0, u)$, and written as $kX_{\leq u}$. For example, the p-key $k\{time, zone\} \in (0.65, 0.75)$ can be rewritten as the l-key $k\{time, zone\}_{\geq 0.65}$ and the u-key $k\{time, zone\}_{\leq 0.75}$. It follows directly that a p-relation satisfies a p-key iff it satisfies the corresponding l-key and u-key. L-keys were studied in [3]. First, we will study u-keys, and then combine them with l-keys.

4 Reasoning Tools

When using sets of p-keys to enforce the consistency and completeness targets on the quality of data, their overhead must be reduced to a minimal level necessary. In practice, this requires us to reason about p-keys efficiently. We will now establish tools to reason about the interaction of p-keys. This will help us identify efficiently i) if a given set of p-keys is consistent, and ii) the most concise interval by which a given key is implied from a given set of p-keys. This helps optimize query and update efficiency, but is also essential for developing our acquisition framework later.

4.1 Computational Problems

Let $\Sigma \cup \{\varphi\}$ denote a set of constraints over relation schema R. We say that Σ is *satisfiable*, if there is some p-relation over R that satisfies all elements of Σ; and say that Σ is *unsatisfiable* otherwise. We say Σ *implies* φ, denoted by $\Sigma \models \varphi$, if every p-relation r over R that satisfies Σ, also satisfies φ. We use $\Sigma^* = \{\varphi : \Sigma \models \varphi\}$ to denote the *semantic closure* of Σ. Let $\mathcal{C}$ denote a class of constraints. The *$\mathcal{C}$-satisfiability problem* is to decide for a given relation schema R and a given set Σ of constraints in $\mathcal{C}$ over R, whether Σ is satisfiable. The *$\mathcal{C}$-implication problem* is to decide for a given relation schema R and a given satisfiable set $\Sigma \cup \{\varphi\}$ of constraints in $\mathcal{C}$ over R, whether Σ implies φ. If $\mathcal{C}$ denotes the class of p-keys, then the *$\mathcal{C}$-inference problem* is to compute for a given relation schema R, a given satisfiable set Σ of p-keys, and a given key kX over R the largest probability l and the smallest probability u such that Σ implies $kX \in (l, u)$. We will characterize the computational problems for u-keys first. Subsequently, we then show how to combine these results with our previous findings on l-keys to characterize the computational problems for p-keys.

4.2 Keys with Upper Bounds

Satisfiability. Unsatisfiability is strong evidence that a set of keys has been over-specified. While every set of l-keys is satisfiable, this is not the case for every set of u-keys. However, satisfiable sets are easy to characterize for u-keys: Unsatisfiability can only originate from stipulating an upper bound smaller than one for the trivial key kR.

Table 3. Axiomatization $\mathfrak{U} = \{\mathcal{R}, \mathcal{F}, \mathcal{W}\}$

$\dfrac{}{kR_{\leq 1}}$	$\dfrac{kXY_{\leq p}}{kX_{\leq p}}$	$\dfrac{kX_{\leq p}}{kX_{\leq p+q}}$
(Maximum, $\mathcal{M}$)	(Fragment, $\mathcal{F}$)	(Relax, $\mathcal{R}$)

Proposition 1. *A set Σ of u-keys over relation schema R is satisfiable if and only if Σ does not contain a u-key of the form $kR_{\leq u}$ where $u < 1$. The satisfiability problem for u-keys can thus be decided with one scan over the input.* □

Axioms. We determine the semantic closure by applying *inference rules* of the form $\dfrac{\text{premise}}{\text{conclusion}}$. For a set $\mathfrak{R}$ of inference rules let $\Sigma \vdash_{\mathfrak{R}} \varphi$ denote the *inference* of φ from Σ by $\mathfrak{R}$. That is, there is some sequence $\sigma_1, \ldots, \sigma_n$ such that $\sigma_n = \varphi$ and every σ_i is an element of Σ or is the conclusion that results from an application of an inference rule in $\mathfrak{R}$ to some premises in $\{\sigma_1, \ldots, \sigma_{i-1}\}$. Let $\Sigma^+_{\mathfrak{R}} = \{\varphi : \Sigma \vdash_{\mathfrak{R}} \varphi\}$ be the *syntactic closure* of Σ under inferences by $\mathfrak{R}$. $\mathfrak{R}$ is *sound* (*complete*) if for every satisfiable set Σ over every R we have $\Sigma^+_{\mathfrak{R}} \subseteq \Sigma^*$ ($\Sigma^* \subseteq \Sigma^+_{\mathfrak{R}}$). The (finite) set $\mathfrak{R}$ is a (finite) *axiomatization* if $\mathfrak{R}$ is both sound and complete. The set $\mathfrak{U}$ of inference rules from Table 3 forms a finite axiomatization for the implication of u-keys. Here, R denotes the underlying relation schema, X and Y form attribute subsets of R, and p, q as well as $p + q$ are probabilities.

Theorem 1. *$\mathfrak{U}$ forms a finite axiomatization for u-keys.* □

It is worth pointing out the soundness of the rules. The maximum rule $\mathcal{M}$ holds trivially, because every marginal probability can at most be one. For the fragment rule $\mathcal{F}$ assume that the marginal probability of kX exceeds p. Since every world that satisfies kX must also satisfy kXY, the marginal probability of kXY exceeds p, too. Finally, for the relax rule $\mathcal{R}$ assume that the marginal probability of kX exceeds $p + q$. Then the marginal probability of kX exceeds p, for sure. Some examples illustrate the use of the inference rule for reasoning about u-keys.

For example, $\Sigma = \{k\{\textit{rfid,time}\}_{\leq 0.75}\}$ implies $\varphi = k\{\textit{time}\}_{\leq 0.8}$, but not $\varphi' = k\{\textit{time}\}_{\leq 0.2}$. Indeed, φ can be inferred from Σ by applying $\mathcal{F}$ to $k\{\textit{rfid,time}\}_{\leq 0.75}$ to infer $k\{\textit{time}\}_{\leq 0.75}$, and applying $\mathcal{R}$ to $k\{\textit{time}\}_{\geq 0.75}$ to infer φ.

If a p-relation satisfies a set Σ of p-keys, then it also satisfies every p-key φ implied by Σ. Consequently, it is redundant to verify that a given p-relation satisfies an implied p-key. In particular, the larger the given p-relation, the more time we save by avoiding such redundant validation checks.

Algorithms. In practice, the semantic closure Σ^* of a finite set Σ is infinite and even though it can be represented finitely, it is often unnecessary to determine all implied constraints. In fact, the implication problem has input $\Sigma \cup \{\varphi\}$ and the question is if Σ implies φ. Computing Σ^* and checking if $\varphi \in \Sigma^*$ is not feasible. We will now establish a linear-time algorithm for computing the smallest

Algorithm 1 Inference

Require: R, Σ, kX with satisfiable set Σ of u-keys
Ensure: $\min\{u : \Sigma \models kX_{\leq u}\}$
1: $p \leftarrow 1$;
2: **for all** $kZ_{\leq q} \in \Sigma$ **do**
3: **if** $X \subseteq Z$ and $q < u$ **then**
4: $u \leftarrow q$;
5: **return** u;

probability u, such that $kX_{\leq u}$ is implied by Σ. The following theorem allows us to reduce the implication problem for u-keys to a single scan of the input.

Theorem 2. *Let $\Sigma \cup \{kX_{\leq u}\}$ denote a satisfiable set of u-keys over relation schema R. Then Σ implies $kX_{\leq u}$ if and only if i) $u = 1$ or ii) there is some $kZ_{\leq q} \in \Sigma$ such that $X \subseteq Z$ and $q \leq u$.* □

Based on Theorem 2, Algorithm 1 returns for a given satisfiable set Σ of u-keys and a given key kX over R, the smallest probability u such that $kX_{\leq u}$ is implied by Σ. Starting with $u = 1$, the algorithm scans all input keys $kZ_{\leq q}$ and sets u to q whenever q is smaller than the current u and X is contained in Z. We use $|S|$ to denote the total number of attributes that occur in set S.

Corollary 1. *On input (R, Σ, kX), Algorithm 1 returns in $\mathcal{O}(|\Sigma| + |R|)$ time the minimum probability u with which $kX_{\leq u}$ is implied by Σ.* □

Given $R, \Sigma, kX_{\leq p}$ as an input to the implication problem for u-keys, Algorithm 1 computes $u := \min\{q : \Sigma \models kX_{\leq q}\}$ and we return an affirmative answer iff $u \leq p$. Hence, the implication problem is linear time decidable in the input.

Corollary 2. *The implication problem of u-keys is decidable in linear time.* □

Given $\Sigma = \{k\{rfid,time\}_{\leq 0.75}\}$ and $k\{time\}$, Algorithm 1 returns $u = 0.75$. For $\varphi' = k\{time\}_{\leq 0.2}$, we conclude that Σ does not imply φ' as $u > 0.2$.

4.3 Keys with Probabilistic Intervals

We will now study p-keys as the combination of l-keys and u-keys. That is, we think of every set Σ of p-keys as the union of the set $\Sigma_l := \{kX_{\leq p} \mid kX \in (p, q) \in \Sigma\}$ of l-keys and the set $\Sigma_u := \{kX_{\geq q} \mid kX \in (p, q) \in \Sigma\}$ of u-keys.
Satisfiability. While satisfiability for l-keys can be decided in constant time [3], and satisfiability for u-keys requires one scan over the input, the satisfiability problem for p-keys requires two scans over the input.

Proposition 2. *A set Σ of p-keys over relation schema R is satisfiable if and only if $\Sigma_l \cup \Sigma_u \cup \{kR_{\geq 1}\}$ does not contain $kX_{\geq p}$, $kXY_{\leq q}$ such that $p > q$. The satisfiability problem for p-keys is decidable with two scans over the input.* □

The set $\Sigma = \{k\{rfid\} \in (0.75, 0.75), k\{rfid,time\} \in (0.6, 0.7)\}$ is unsatisfiable.

No interaction. We reduce the remaining computational problems for p-keys to those of u-keys and l-keys. This is possible since we can show that every satisfiable set of p-keys does not exhibit any interaction between its l-keys and u-keys. Formally, u-keys and l-keys *do not interact* if and only if for every relation schema R, every satisfiable set Σ of p-keys, every l-key $kX_{\geq p}$ and every u-key $kY_{\geq u}$ over R, the following two conditions hold:

- $\Sigma_u \cup \Sigma_l \models kX_{\geq p}$ if and only if $\Sigma_l \models kX_{\geq p}$,
- $\Sigma_u \cup \Sigma_l \models kY_{\leq q}$ if and only if $\Sigma_u \models kY_{\leq q}$.

In other words, the non-interaction between u-keys and l-keys enables us to reduce the implication problem for p-keys to the implication problems for u-keys and l-keys. That is, a p-key $kX \in (p, q)$ is implied by a satisfiable set Σ of p-keys if and only if i) $kX_{\geq p}$ is implied by Σ_l, and ii) $kX_{\leq q}$ is implied by Σ_u.

Theorem 3. *U-keys and l-keys do not interact.*

Proof (Sketch). The non-trivial direction is to show the following: if $\Sigma_l \not\models kX_{\geq p}$, then $\Sigma_u \cup \Sigma_l \not\models kX_{\geq p}$. If $\Sigma_l \not\models kX_{\geq p}$, then any Armstrong p-relation for Σ_l satisfies Σ_l and violates $kX_{\geq p}$. Since $\Sigma_u \cup \Sigma_l$ is satisfiable, it follows that the Armstrong p-relation for Σ_l also satisfies Σ_u. Consequently, $\Sigma_u \cup \Sigma_l \not\models kX_{\geq p}$. The arguments works similarly when we know that $\Sigma_u \not\models kY_{\leq q}$. We can create an Armstrong p-relation for Σ_u, which must also satisfy Σ_l because $\Sigma_u \cup \Sigma_l$ is satisfiable. □

Theorem 3 allows us to reduce the implication and inference problems for p-keys to the implication and inference problems for l-keys and u-keys. As a first consequence, combining our axiomatizations for u-keys and l-keys yields an axiomatization for p-keys.

Corollary 3. *Axiomatization $\mathfrak{U}$ for u-keys from Theorem 1 together with axiomatization $\mathfrak{P}$ for l-keys from [3] form a finite axiomatization for p-keys.* □

As a second consequence, we can also combine our inference algorithms for u-keys and l-keys to obtain an efficient inference algorithm for p-keys.

Corollary 4. *Given relation schema R, a satisfiable set Σ of p-keys, and a key kX over R, we can return in $\mathcal{O}(|\Sigma| + |R|)$ time the maximum probability l and the minimum probability u such that $kX \in (l, u)$ is implied by Σ.* □

Thirdly, the implication problem of p-keys can be decided efficiently.

Corollary 5. *The implication problem of p-keys is decidable in linear time.* □

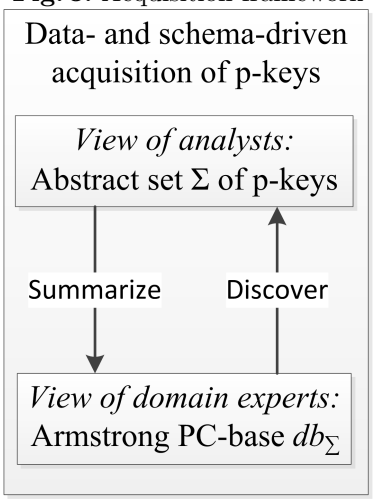
To decide if $k\{time,zone\} \in (0.6, 0.7)$ is implied by the set Σ of p-keys from Figure 2, we check if $k\{time,zone\}_{\geq 0.6}$ is implied by Σ_l and if $k\{time,zone\}_{\leq 0.7}$ is implied by Σ_u. As the second condition fails, the p-key is not implied by Σ.

Our results show that it takes quadratic time in the input to keep the enforcement of p-key sets to a minimal level necessary: For a given p-key set, we can remove successively all p-keys from the set that are implied by the remaining set. More validation time is saved the bigger the underlying p-relations grow.

5 Tools for Acquiring Probabilistic Key Intervals

The main inhibitor to the uptake of p-keys is the difficulty to determine the right interval for the marginal probabilities by which keys hold in the underlying application domain. For that purpose, analysts should communicate with domain experts. We establish two major computational tools that help analysts communicate effectively with domain experts. We follow the framework in Figure 3. Here, analysts use our algorithm to summarize abstract sets Σ of p-keys in the form of some Armstrong PC-base, which is then inspected jointly with domain experts. In particular, the two PC-tables that form together the PC-base represent simultaneously for every key kX their lowest and highest marginal probabilities that quality data sets in the target domain should exhibit. Domain experts may change the PC-tables or supply new PC-tables to the analysts. For that case we establish an algorithm that discovers p-keys from sets of PC-tables. That is, the algorithm computes the lowest and highest marginal probabilities of each key across all the given PC-tables. Such profiles are also useful for query optimization, for example.

Fig. 3. Acquisition framework



5.1 Summarizing Abstract Sets of P-Keys as Armstrong PC-bases

Our results will show that every satisfiable set Σ of p-keys can be summarized in the form of two PC-tables such that all given p-keys are satisfied by the two p-relations the PC-tables represent, and all those p-keys not implied by Σ are violated by at least one of the p-relations. This notion generalizes the concept of an *Armstrong database*, which is a *single* database instance that satisfies a constraint if and only if it is implied by the given constraint set [5]. The reason why p-keys require two database instances is simple: Each instance can only represent one marginal probability, but p-keys generally require a lower and an upper bound on the marginal probability. So, unless every given key has the same lower and upper bounds, we require two database instances. The formal definition is therefore as follows.

Definition 2. *Let Σ be a satisfiable set of p-keys over a given relation schema R. A pair of p-relation r_1, r_2 over R is Armstrong for Σ if and only if for all p-keys φ over R it holds that r_1 and r_2 satisfy φ if and only if Σ implies φ.*

For example, the p-relations r_1, r_2 from Table 1 are Armstrong for the set Σ of p-keys in Figure 2. It is worth emphasizing the effectiveness of the definition:

Knowing that r_1, r_2 are Armstrong for a given Σ enables us to reduce *every* instance $\Sigma \cup \{\varphi\}$ of the implication problem to simply checking if both r_1 and r_2 satisfy φ. Knowing that u-keys and l-keys do not interact, we can compute r_1, r_2 such that *every* instance $\Sigma \cup \{kX\}$ of the inference problem is reduced to simply computing the lower (upper) bound l (u) in $kX \in (l, u)$ as the marginal probability m_{X,r_1} (m_{X,r_2}) of kX in r_1 (r_2). For example, the $k\{time,zone\} \in (0.6, 0.7)$ is not implied by Σ from Figure 2 as the given upper bound 0.7 is smaller than the marginal probability 0.75 of $k\{time,zone\}$ in r_2.

Instead of computing Armstrong p-relations we compute PC-tables that are more concise representations. We call these *Armstrong* PC-bases. Recall the following standard definition from probabilistic databases [24]. A *conditional table* or *c-table*, is a tuple $CD = \langle r, W \rangle$, where r is a relation, and W assigns to each tuple t in r a finite set W_t of positive integers. The set of *world identifiers* of CD is the union of the sets W_t for all tuples t of r. Given a world identifier i of CD, the possible world associated with i is $W_i = \{t | t \in r \text{ and } i \in W_t\}$. The semantics of a c-table $CD = \langle r, W \rangle$, called *representation*, is the set $\mathcal{W}$ of possible worlds W_i where i denotes some world identifier of CD. A *probabilistic conditional database* or *PC-table*, is a pair $\langle CD, P \rangle$ where CD is a c-table, and P is a probability distribution over the set of world identifiers of CD. The set of possible worlds of a PC-table $\langle CD, P \rangle$ is the representation of CD, and the probability of each possible world W_i is defined as the probability of its world identifier. For example, the PC-tables from Table 2 form an Armstrong PC-base for the set Σ of p-keys from Figure 2.

Algorithm 2 in [3] computes a single Armstrong PC-table for every given set Σ of l-keys. In the construction, the number of possible worlds is given by the number of distinct lower bounds that occur in Σ. Indeed, for every given set Σ of l-keys over R and every $p \in (0, 1]$, $\Sigma_p = \{kX : \exists kX_{\geq q} \in \Sigma \wedge q \geq p\}$ denotes the *p-cut* of Σ. If Σ does not contain a p-key $kX_{\geq p}$ where $p = 1$, an Armstrong PC-table for Σ is computed that contains one more possible world than the number of distinct lower bounds in Σ. Processing the bounds in Σ from smallest p_1 to largest p_n, the algorithm computes as possible world with probability $p_i - p_{i-1}$ a traditional Armstrong relation for the p_i-cut Σ_{p_i}. For this purpose, the anti-keys are computed for each p_i-cut, and the set W of those worlds i is recorded for which X is an anti-key with respect to Σ_{p_i}. The CD-table contains one tuple t_0 which occurs in all worlds, and for each anti-key X another tuple t_j that occurs in all worlds for which X is an anti-key and that has matching values with t_0 in exactly the columns of X.

For example, applying this construction to the lower bounds of p-keys in Figure 2 produces the PC-table on the left of Table 2. Indeed, let Σ consist of $k\{rfid, time\}_{\geq .75}$, $k\{rfid, zone\}_{\geq .35}$, and $k\{time, zone\}_{\geq .65}$. Then $\Sigma_{.35}$ consists of $k\{rfid, time\}$, $k\{rfid, zone\}$, and $k\{time, zone\}$; $\Sigma_{.65}$ consists of $k\{rfid, time\}$, and $k\{time, zone\}$; $\Sigma_{.75}$ consists of $k\{rfid, time\}$; and Σ_1 is empty. The world W_1 has thus probability $p_1 = 0.35$, and is an Armstrong relation for $\Sigma_{0.35}$. Here, we have the three singleton anti-keys $\{rfid\}$, $\{time\}$, and $\{zone\}$. W_1 has four tuples, the first and second tuple have matching values on $\{rfid\}$, the first and

third tuple have matching values on $\{zone\}$, and the first and fourth tuple have matching values on $\{time\}$. This gives us the Armstrong relation W_1 of r_1 shown in Table 1. The world W_2 has probability $p_2 = 0.3$, and is an Armstrong relation for $\Sigma_{0.65}$. Here, we have the two anti-keys $\{time\}$ and $\{rfid, zone\}$. The world W_3 has probability $p_3 = 0.1$, and is an Armstrong relation for $\Sigma_{0.75}$. Here, we have the two anti-keys $\{rfid, zone\}$ and $\{time, zone\}$. Finally, W_4 has probability $p_4 = 0.25$, and is an Armstrong relation for Σ_1. Here, we have the three anti-keys $\{rfid, zone\}$, $\{time, zone\}$, and $\{rfid, time\}$. Similar to W_1 it is easy to see how W_2, W_3, and W_4 of r_1 in Table 1 constitute the corresponding Armstrong relations. Finally, we simply record the identifiers of those worlds in which a tuple appears to obtain the CD-table. This results in the CD-table shown in Table 2.

The outlined algorithm can also compute an Armstrong PC-table for every satisfiable set Σ of u-keys. The reason is that the algorithm is independent of whether we view the given probabilities as lower or upper bounds. The only necessary change concerns the definition of Σ_p which becomes $\Sigma_p = \{kX : \exists kX_{\leq q} \in \Sigma \wedge q \geq p\}$ in this case. For example, applying this construction to the upper bounds of p-keys in Figure 2 results in the PC-table on the right of Table 2.

The use of this algorithm can be taken further when we consider Theorem 3, which states that u-keys and l-keys do not interact in satisfiable sets. This means, given a satisfiable set Σ of p-keys, we can compute an Armstrong PC-base for Σ by applying Algorithm 2 of [3] to compute an Armstrong PC-table for the set Σ_l, and by applying Algorithm 2 of [3] to compute an Armstrong PC-table for the set Σ_u. Indeed, we obtain an Armstrong PC-base for Σ by simply pairing the outputs of both applications together.

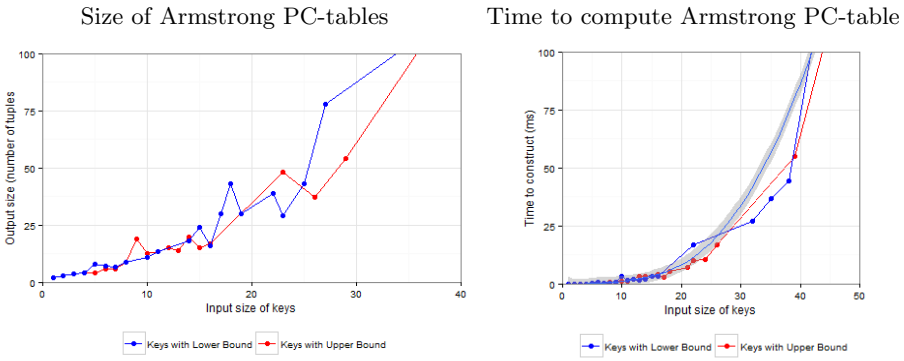
Theorem 4. *For every satisfiable set Σ of p-keys over relation schema R, applications of Algorithm 2 in [3] to Σ_l and Σ_u, respectively, result in an Armstrong PC-base for Σ in which the total number of possible worlds coincides with the sum of the distinct non-zero lower bounds in Σ'_l and the distinct non-zero upper bounds in Σ'_u. Here, Σ'_l (Σ'_u) denotes Σ_l (Σ_u) if there is some $X \subseteq R$ such that $kX_{\geq 1} \in \Sigma_l$ ($kX_{\leq 1} \in \Sigma_u$), and $\Sigma_l \cup \{kR_{\geq 1}\}$ ($\Sigma_u \cup \{kR_{\leq 1}\}$) otherwise.* □

The PC-tables of Table 2 form an Armstrong PC-base for the set Σ of p-keys in Figure 2. Indeed, the number of possible worlds in both PC-tables is 4, which is the number of distinct non-zero lower bounds in Σ and also the number of distinct non-zero upper bounds in Σ. Finally, we derive some bounds on the time complexity of finding Armstrong PC-tables. Additional insight is given by our experiments in Section 6.

Theorem 5. *The time complexity to find an Armstrong PC-base for a given set Σ of p-keys over relation schema R is precisely exponential in $|\Sigma|$.*

Here, precisely exponential means that there is an algorithm which requires exponential time and that there are cases in which the number of tuples in the output is exponential in the input size. Nevertheless, there are also cases where

Fig. 4. Results of experiments with visualization



the number of tuples in some Armstrong PC-base for Σ over R is logarithmic in $|\Sigma|$. Such a case is given by $R_n = \{A_1, \ldots, A_{2n}\}$ and $\Sigma_n = \{k(X_1 \cdots X_n) \in (1,1) : X_i \in \{A_{2i-1}, A_{2i}\}$ for $i = 1, \ldots, n\}$ with $|\Sigma_n| = n \cdot 2^n$.

5.2 Discovery of P-Keys from Collections of PC-tables

The discovery problem of p-keys from a collection of PC-tables over a relation schema R is to determine for all $X \subset R$, the smallest marginal probability $l_{X,r}$ and the largest marginal probability $u_{X,r}$ of kX across all given p-relations $r = (\mathcal{W}, P)$ represented by some given PC-table. The problem of computing the marginal probability $m_{X,r}$ can be solved as follows: For each $X \subset R$, initialize $m_{X,r} \leftarrow 0$ and for all worlds $W \in \mathcal{W}$, add the probability p_W of W to $m_{X,r}$, if X contains some minimal key of W. The set of minimal keys of a world W is given by the set of minimal transversals over the disagree sets of W (the complements of agree sets) [21]. For example, applying this algorithm to the PC-tables from Table 2 returns the p-keys shown in Figure 2.

6 Experiments

In this section we report on some experiments regarding the computational complexity of our algorithms for the summarization and discovery of p-keys.

Summarization. While the worst-case time complexity of generating Armstrong PC-tables is exponential, these cases occur rarely in practice and at random. In our experiment we simulated average case behavior by generating sets of keys with upper/lower bounds. For each key, the set of attributes, the associated probability, and the type (either upper or lower) were randomly selected. First, we checked whether the created set of p-keys was satisfiable. If it was, one Armstrong PC-table was computed for the set of keys with upper bounds and

Fig. 5. GUI for summarization and times for discovering p-keys

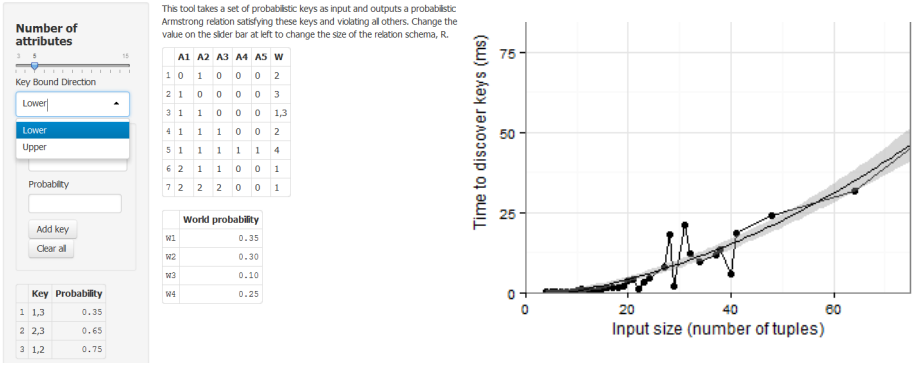
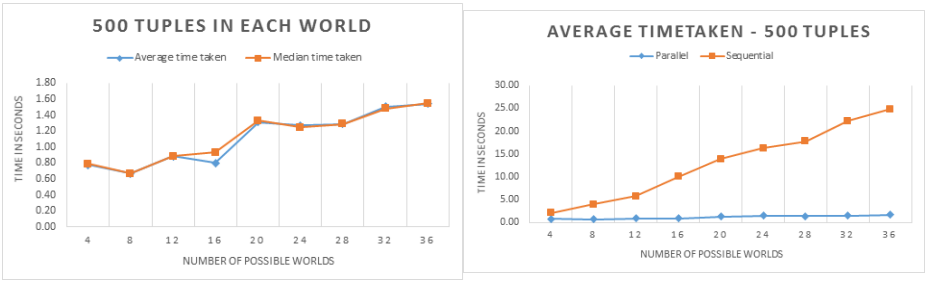


Fig. 6. Results on "Car" data set for MapReduce Implementation



one for the set of keys with lower bounds. Overall, 24% of the p-key sets created were unsatisfiable. The average sizes and times to create the Armstrong PC-tables are shown in Figure 4. The results demonstrate that Armstrong PC-bases exhibit small sizes on average, which makes them a practical tool to acquire keys with meaningful probabilistic intervals in a joint effort with domain experts. A screenshot of our graphical user interface is shown on the left of Figure 5.

Discovery. The right of Figure 5 shows the discovery times of p-keys from two given PC-tables. The input size is the total number of tuples in the input. We also applied a MapReduce implementation on a single node machine with 40 processors to the "Car" data set[3] of the UCI Machine Learning Repository. We converted "Car" into a p-relation with rising numbers of possible worlds and 500 tuples in each world. Figure 6 shows that our algorithm for the discovery of p-keys scales linearly in the number of possible worlds, considering this number is relatively low in our acquisition framework.

[3] `http://archive.ics.uci.edu/ml/datasets/Car+Evaluation`

7 Conclusion and Future Work

We introduced keys with probabilistic intervals, which stipulate lower and upper bounds on the marginal probability by which keys shall hold on large volumes of uncertain data. Keys with probabilistic intervals provide a principled, yet simple enough mechanism to control the consistency and completeness targets for the quality of an organization's uncertain data. Similar to how lower bounds say that a key is satisfied with some minimum probability, upper bounds provide us with means to say that a key is violated with a minimum probability. Our axiomatic and algorithmic reasoning tools minimize the overhead in using the keys for data quality management and query processing. Our findings for the visualization and discovery of these keys provide effective support for the efficient acquisition of the right probabilistic intervals that apply in a given application domain.

In future research we will apply our algorithms to investigate empirically the usefulness of our framework for acquiring the right probabilistic intervals of keys in a given application domain. This will require us to extend empirical measures from certain [17] to probabilistic data. Particularly intriguing is the question whether PC-bases or their p-relations are more useful. It is also interesting to investigate probabilistic variants of other useful constraint sets, such as functional, multivalued, and inclusion dependencies [7, 8, 10, 13, 14, 18, 19]. However, we have shown that such variants are not finitely axiomatizable. In this sense, our results for p-keys are rather special. An extension that seems feasible is to add lower bounds to the probabilistic cardinality constraints from [22].

References

1. Abedjan, Z., Golab, L., Naumann, F.: Profiling relational data: a survey. VLDB J. 24(4), 557–581 (2015)
2. Beeri, C., Dowd, M., Fagin, R., Statman, R.: On the structure of Armstrong relations for functional dependencies. J. ACM 31(1), 30–46 (1984)
3. Brown, P., Link, S.: Probabilistic keys for data quality management. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) Advanced Information Systems Engineering - 27th International Conference, CAiSE 2015, Stockholm, Sweden, June 8-12, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9097, pp. 118–132. Springer (2015)
4. Caruccio, L., Deufemia, V., Polese, G.: Relaxed functional dependencies - A survey of approaches. IEEE Trans. Knowl. Data Eng. 28(1), 147–165 (2016)
5. Fagin, R.: Horn clauses and database dependencies. J. ACM 29(4), 952–985 (1982)
6. Hannula, M., Kontinen, J., Link, S.: On the finite and general implication problems of independence atoms and keys. J. Comput. Syst. Sci. 82(5), 856–877 (2016)
7. Hartmann, S., Link, S.: Multi-valued dependencies in the presence of lists. In: Beeri, C., Deutsch, A. (eds.) Proceedings of the Twenty-third ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 14-16, 2004, Paris, France. pp. 330–341. ACM (2004)
8. Hartmann, S., Link, S.: On a problem of Fagin concerning multivalued dependencies in relational databases. Theor. Comput. Sci. 353(1-3), 53–62 (2006)

9. Hartmann, S., Link, S.: Efficient reasoning about a robust XML key fragment. ACM Trans. Database Syst. 34(2) (2009)
10. Hartmann, S., Link, S., Schewe, K.: Functional and multivalued dependencies in nested databases generated by record and list constructor. Ann. Math. Artif. Intell. 46(1-2), 114–164 (2006)
11. Köhler, H., Leck, U., Link, S., Prade, H.: Logical foundations of possibilistic keys. In: Fermé, E., Leite, J. (eds.) Logics in Artificial Intelligence - 14th European Conference, JELIA 2014, Funchal, Madeira, Portugal, September 24-26, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8761, pp. 181–195. Springer (2014)
12. Köhler, H., Leck, U., Link, S., Zhou, X.: Possible and certain keys for SQL. The VLDB Journal 25(4), 571–596 (2016)
13. Köhler, H., Link, S.: Inclusion dependencies reloaded. In: Bailey, J., Moffat, A., Aggarwal, C.C., de Rijke, M., Kumar, R., Murdock, V., Sellis, T.K., Yu, J.X. (eds.) Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015. pp. 1361–1370. ACM (2015)
14. Köhler, H., Link, S.: SQL schema design: Foundations, normal forms, and normalization. In: Özcan, F., Koutrika, G., Madden, S. (eds.) Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016. pp. 267–279. ACM (2016)
15. Köhler, H., Link, S., Zhou, X.: Possible and certain SQL keys. PVLDB 8(11), 1118–1129 (2015)
16. Köhler, H., Link, S., Zhou, X.: Discovering meaningful certain keys from incomplete and inconsistent relations. IEEE Data Eng. Bull. 39(2), 21–37 (2016)
17. Langeveldt, W., Link, S.: Empirical evidence for the usefulness of Armstrong relations in the acquisition of meaningful functional dependencies. Inf. Syst. 35(3), 352–374 (2010)
18. Link, S.: Charting the completeness frontier of inference systems for multivalued dependencies. Acta Inf. 45(7-8), 565–591 (2008)
19. Link, S.: Characterisations of multivalued dependency implication over undetermined universes. J. Comput. Syst. Sci. 78(4), 1026–1044 (2012)
20. Liu, J., Li, J., Liu, C., Chen, Y.: Discover dependencies from data - A review. IEEE Trans. Knowl. Data Eng. 24(2), 251–264 (2012)
21. Mannila, H., Räihä, K.J.: Algorithms for inferring functional dependencies from relations. Data Knowl. Eng. 12(1), 83–99 (1994)
22. Roblot, T., Link, S.: Probabilistic cardinality constraints. In: Johannesson, P., Lee, M., Liddle, S.W., Opdahl, A.L., López, O.P. (eds.) Conceptual Modeling - 34th International Conference, ER 2015, Stockholm, Sweden, October 19-22, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9381, pp. 214–228. Springer (2015)
23. Sadiq, S.: Handbook of Data Quality. Springer (2013)
24. Suciu, D., Olteanu, D., Ré, C., Koch, C.: Probabilistic Databases. Synthesis Lectures on Data Management, Morgan & Claypool Publishers (2011)
25. Thalheim, B.: On semantic issues connected with keys in relational databases permitting null values. Elektronische Informationsverarbeitung und Kybernetik 25(1/2), 11–20 (1989)
26. Toman, D., Weddell, G.E.: On keys and functional dependencies as first-class citizens in description logics. J. Autom. Reasoning 40(2-3), 117–132 (2008)