Discovery of Genuine Functional Dependencies from Relational Data with Missing Values

Laure Berti-Équille
Aix-Marseille Univ., CNRS, LIS
Marseille, France
laure.berti-equille@univ-amu.fr

Hazar Harmouch
Hasso Plattner Institute
University of Potsdam, Germany
hazar.harmouch@hpi.de

Felix Naumann
Hasso Plattner Institute
University of Potsdam, Germany
felix.naumann@hpi.de

Noël Novelli
Aix-Marseille Univ., CNRS, LIS
Marseille, France
noel.novelli@lis-lab.fr

Saravanan Thirumuruganathan
QCRI, HBKU
Doha, Qatar
sthirumuruganathan@hbku.edu.qa

So maybe they only handle NULL

ABSTRACT

Functional dependencies (FDs) play an important role in maintaining data quality. They can be used to enforce data consistency and to guide repairs over a database. In this work, we investigate the problem of missing values and its impact on FD discovery. When using existing FD discovery algorithms, some genuine FDs could not be detected precisely due to missing values or some non-genuine FDs can be discovered even though they are caused by missing values with a certain NULL semantics. We define a notion of genuineness and propose algorithms to compute the genuineness score of a discovered FD. This can be used to identify the *genuine FDs* among the set of all valid dependencies that hold on the data. We evaluate the quality of our method over various real-world and semi-synthetic datasets with extensive experiments. The results show that our method performs well for relatively large FD sets and is able to accurately capture genuine FDs.

So even if FD holds in current snapshot will it actually exist?

PVLDB Reference Format:
Laure Berti-Équille, Hazar Harmouch, Felix Naumann, Noël Novelli and Saravanan Thirumuruganathan. Discovery of Genuine Functional Dependencies from Relational Data with Missing Values. *PVLDB*, 11(8): 880-892, 2018.
DOI: https://doi.org/10.14778/3204028.3204032

1. INTRODUCTION

Functional dependencies (FDs) are one of the most important types of integrity constraints and have been extensively studied by the research community [7,27]. FDs have a number of applications, such as maintaining data quality [15], capturing schema semantics [16], schema normalization [31], data integration [34], repairing of data inconsistencies, and data cleaning [4,6]. An FD $X \to A$ states that the tuples of attribute set X uniquely determine the value of attribute (set) A. Traditional FDs are typically defined for correct and complete data and there are many efficient algorithms to discover FDs from a given clean dataset [30].

However, many real-world datasets are neither correct nor complete. Traditional FDs often have trouble with incomplete data, such as NULL values, that routinely exist in massive datasets. A common, implicit assumption is that the proportion of incomplete data is assumed to be relatively low and with no significant impact on the set of discovered dependencies. However, it is well-known that data error rates may vary from 20% up to 80% in many real-world datasets [12,18] with dramatic consequences and significant costs [11,19]. Not surprisingly, the database community has come up with a number of workarounds to handle this issue. We provide a representative list of approaches and briefly mention why they are not satisfactory. The experimental results are based on Glass dataset, a benchmark dataset with 10 attributes and only 214 tuples, that we used in our experiments (see Section 6).

Strategy 1: Skipping tuples with NULLs. The simplest strategy is to ignore the set of tuples with NULLs and use the remaining subset of the relation to discover FDs. This approach suffers from two problems. First, as mentioned above, large parts of the dataset can contain NULL values. Second, this approach also discovers a number of spurious FDs that do not hold on the entire relation. We illustrate this issue in Figure 1 (bars on left). There are 11,263 FDs on the correct and complete version of Glass dataset. However, when we injected missing values randomly into 5% of the tuples and skipped those incomplete tuples, the number of discovered FDs jumped to 12,736 (on average over 10 runs).

Strategy 2: NULL Semantics. An alternative approach is to propose definitions of FDs over relations with NULLs. The two commonly used NULL semantics are `NULL-EQ` or `NULL-NOT-EQ` that treat all missing values as identical or distinct, respectively. For example, if two tuples t_i and t_j have NULL for an attribute A_k,

Proceedings of the VLDB Endowment, Vol. 11, No. 8
Copyright 2018 VLDB Endowment 2150-8097/18/04.
DOI: https://doi.org/10.14778/3204028.3204032

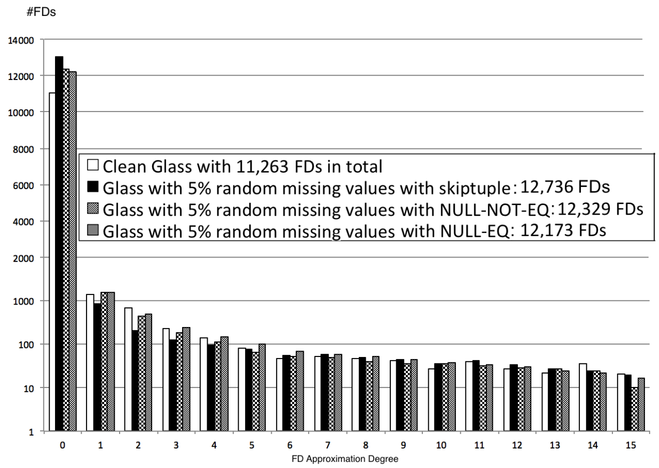

Figure 1: Number of FDs discovered from the clean version of Glass dataset and a polluted version.

then NULL-EQ assumes that both tuples have the same indeterminate value. NULL-NOT-EQ assumes that t_i and t_j have different but still indeterminate value. However, as we shall show in Section 3.1, this approach also does not control the discovery of spurious FDs. This can also be seen in Figure 1 where there are 11,263 FDs on the clean dataset but almost 12,329 and 12,173 FDs for NULL-NOT-EQ and NULL-EQ respectively.

Strategy 3: Approximate FDs. Another strategy is to relax the requirement that the FD holds on all the tuples. Instead, one can aim to discover approximate (a.k.a. partial) FDs that are violated by at most a small fraction of the tuples. This approach also suffers from multiple issues whereby identifying a correct threshold is not straightforward and it does not prevent from discovering a large number of spurious FDs. This is illustrated in Figure 1 where the number of approximate FDs discovered over the complete dataset and the incomplete dataset are different for various approximation degrees (the number of tuples that must be removed from the dataset so that the FD holds). For example, there are 1,156 FDs that are violated by exactly 1 tuple (approximation degree 1). However, there are 379, 1328, and 1404 FDs that violate exactly 1 tuple when we skip incomplete tuples or apply NULL-EQ, NULL-NOT-EQ semantics, respectively.

Strategy 4: Data Imputation. Data imputation refers to the process of replacing missing data with substituted values [13]. One can use probabilistic or other statistical imputation techniques to fill the missing data, such as [17,32], and run FD discovery on the imputed data. However, the FDs discovered are tightly tethered to the imputation strategy. Further, in the case of probabilistic imputation, it can happen that the FDs discovered are valid only in a small fraction of all possible imputed worlds.

Taxonomy of FDs discovered over NULLs. We investigate the effect of incomplete data on the discovery of functional dependencies and compare the FDs that are discovered from the dirty dataset and its corresponding clean version. It is clear that the FD discovery result from the incomplete data includes spurious FDs that do not hold on the clean dataset and also misses some FDs that do.

In order to systematically study this phenomenon, we define three types of FDs: *genuine, ghost*, and *fake* FDs. A *genuine* FD is an FD that would be valid if the dataset contained no missing values and no other errors. When the data is incomplete, a traditional FD discovery could discover false positive (*fake*) FDs that do not hold on the complete version of data, or miss discovering some true FDs (*ghost*) that actually hold on the complete data. In other words, most current FD discovery techniques do not provide any guarantee regarding the genuineness of the discovered dependencies. Further, these methods neither detect nor remove the false dependencies that are supported by incomplete data and they do not consider the true dependencies that have actually disappeared due to missing values. Note that we do *not* address the quite different problem of judging whether a valid FD is in fact semantically correct. This latter decision can, in principle, be made only by a human expert.

Impact of Fake and Ghost FDs. As mentioned earlier, functional dependencies have been used in a number of applications, such as data cleaning and query optimization. Use of non-genuine FDs for such scenarios could have a deleterious effect. For example, FDs that were not identified due to incomplete data could be considered as missed opportunities for schema normalization and data cleaning. On the other hand, false positive FDs could cause issues when they are used in query optimization. Finally, false positive FDs, when used as data integrity constraints, prevent the insertion of valid tuples. In this paper, we define the notion of genuineness of FDs and develop algorithms to estimate it.

Our contributions. Despite its importance, no previous work has focused on these critical aspects of FD discovery over incomplete data. We make the following contributions.

- We formally and experimentally show the phenomenon caused by missing values over FD discovery;
- We formalize the definitions of *genuine*, *ghost*, and *fake* FDs and study their impact under various NULL semantics and imputation strategies;
- Given a dataset with missing values, we propose a probabilistic approach for estimating the genuineness score of FDs and provide an efficient method for enumerating and pruning irrelevant possible worlds;
- We propose two efficient algorithms to approximate the genuineness score of discovered FDs;
- We perform an extensive set of experiments of our methods on real-world and semi-synthetic datasets that show the effectiveness and efficiency of our approach.

The rest of the paper is organized as follows. In Section 2, we introduce key definitions and notations. In Section 3, we provide an illustrative example to motivate the need for this study and then formally define the notions of *fake*, *ghost*, and *genuine* FDs. In Sections 4 and 5, we propose a series of algorithms to quantify the genuineness of an FD. In Section 6, we present our experimental evaluation. Section 7 presents related work, and finally, Section 8 concludes our paper and suggests future work.

2. PRELIMINARIES

Let R be a relation with schema $\{A_1, \dots, A_m\}$ with n tuples and m attributes. The domain of an attribute A_i is denoted as $dom(A_i)$. Let $X \subseteq R$ be a subset of attributes. The projection of a tuple t to a set of attributes X is denoted as $t[X]$.

Functional dependency (FD). An FD $X \rightarrow Y$ over a set of attributes $X, Y \subseteq R$ states that X functionally determines Y. X is the determinant (LHS) and Y is the dependent (RHS). The FD is said to hold on R when $\forall t_i, t_j \in R$, if $t_i[X] = t_j[X]$ then $t_i[Y] = t_j[Y]$. The FD is violated when there exists at least one pair of tuples (t_i, t_j) such that $t_i[X] = t_j[X]$ but $t_i[Y] \neq t_j[Y]$. An FD $X \rightarrow Y$ is said to be *minimal* if no subset of X determines Y. In other words, removing any attribute from X renders the FD invalid. An FD is said to be *non-trivial* if $X \cap Y = \emptyset$. An FD is said to be *normalized* if the RHS is a single attribute. In this paper, we consider only the set of minimal, non-trivial, and normalized FDs as they can be used to infer all other FDs that hold on R using Armstrong's axioms.

Approximate functional dependency (AFD). An approximate (or partial) functional dependency holds on most of the tuples and is violated by a small number of tuples. We can quantify the approximation through the notion of satisfaction error [21, 27]. Given an AFD $X \rightarrow A$, the G_3 metric measures the number of violating tuples that must be deleted from R such that the AFD holds exactly (no violation). We use G_3 as the approximation degree measure for the rest of the paper. We represent an AFD with an approximation degree of α as $X \rightharpoondown^{\alpha} A$. Note that $X \rightharpoondown^{0} A$ is the same as $X \rightarrow A$.

NULL semantics. In the real-world, data is often incomplete, possibly due to the integration of multiple relations. The missing values are represented as NULL values, which we denote with $\perp$. As mentioned earlier, when dealing with NULL values, two semantics have been traditionally proposed: the first interpretation, NULL-EQ, denoted ($\perp = \perp$) considers that all missing values are identical. The second semantics, NULL-NOT-EQ, denoted ($\perp \neq \perp$) considers that every missing value is distinct. The two semantics have diverse motivations and lead to the discovery of different sets of functional dependencies.

We would like to note that from the perspective of FD discovery, the impact of NULL values and the impact of other erroneous values, such as typos and outliers, are very similar. For example, one could use an orthogonal mechanism to identify typos or outliers and simply set those erroneous cells to NULL that have to be fixed later. However, in the rest of the paper, we consider only the case of missing values in FD discovery. Regardless, our method can be naturally extended to handle outliers and typos.

3. GENUINE FD DISCOVERY

In this section, we first provide an illustrative example of the impact of missing values on the discovery of FDs. Next, we formalize the definition of *genuine*, *fake*, and *ghost* FDs in the case of exact and approximate FDs.

3.1 Illustrative example

Let us consider the relation R_0 with schema $R_0(A, B, C)$ in Figure 2. As we show in Section 6.4 with a Sensor dataset, A, B and C could represent attributes such as `sensorId`, `temperature`, and `humidity`, collected by sensors monitoring room-conditions in a lab. FDs reflect the relationships between certain sensors and certain environmental conditions and missing values are frequent due to dysfunctional sensors. In this example, we compute the set of exact and approximate FDs from R_0. Table F_0 (below R_0) gives a sample of the FDs discovered, with their corresponding approximation degree. For non-zero approximation degree, we also list the identifiers of the tuples that need to be changed or removed, using "|" as OR operator and "," as AND operator on the same line of the approximate FD. For instance, in the last line of table F_0, the notation of the AFD $C \rightharpoondown^{2} A$ $\{(t_1|t_2), (t_3|t_4)\}$ means that two tuples (t_1 or t_2) and (t_3 or t_4) have to be removed or updated so that $C \rightarrow A$ becomes valid.

Now, suppose we randomly inject missing values (denoted by $\perp$) in the original dataset R_0 to create three polluted versions of the dataset, denoted R_1, R_2, and R_3 containing one, two, and three missing values, respectively. We recompute the set of exact and approximate FDs for each dataset version, noted F_1, F_2 (in Figure 2), and for F_3 (in Figure 3) with the two NULL semantics. In R_1 and R_2, both NULL-EQ or NULL-NOT-EQ are identical as there is only one missing value per attribute in Figure 2. Then, we compare the original set of FDs reported in F_0 with the sets of FDs discovered from each polluted version. We can observe some interesting differences in the discovered FD sets. For example, when we compare F_0 to F_1, and F_1 to F_2, the more the number of missing values increases, the more FDs are lost for a fixed approximation degree (e.g., the exact FDs $A \rightarrow C$ and $B \rightarrow C$ disappear from F_1 to F_2), and new FDs appear (e.g., $B \rightarrow A$ appears from F_0 to F_1).

Actually, the original FDs do not completely disappear, they lose one approximation degree and "fade out". For example, $A \rightarrow C$ and $B \rightarrow C$ in F_0 and F_1 become $A \rightharpoondown^{1} C$ and $B \rightharpoondown^{1} C$ in F_2 respectively; another example is $C \rightharpoondown^{1} B$ in F_0, which becomes $C \rightharpoondown^{2} B$ in F_2.

In this example we deliberately injected missing values, but an integration scenario with data from multiple, heterogeneous sources may well introduce such missing values, which similarly affects FD discovery. As seen in Figures 2 and 3, missing value injection produces some FDs that were not discovered in R_0. Another interesting phenomenon is illustrated in Figure 3, where three missing values were injected in R_0. In the case where NULL values have the NULL-EQ semantics ($F_3^{=}$), exact FDs are no longer discovered but some FDs appear (e.g., $C \rightharpoondown^{1} A$), even though they were not present in the original FD set F_0 with the same approximation degree.

Interestingly, the two FD sets, $F_3^{=}$ and $F_3^{\neq}$ obtained using the two NULL-EQ and NULL-NOT-EQ semantics are very different for the first two approximation degrees. Which one should be selected? Which FD set is the closest to F_0, the set obtained from the original, clean dataset? What if the three missing values were injected differently? Obviously, removing all tuples with missing values would also lead to another quite different FD set, even further apart from the one obtained from the original dataset.

This phenomenon has neither been identified nor studied by previous work: Either FDs are computed from a supposedly complete and error-free dataset, where records with missing values do not exist or are excluded from the FD

R_0	A	B	C
t_1	0	1	1
t_2	0	1	1
t_3	1	1	1
t_4	1	0	1

F_0	Deg.	FDs discovered from R_0
	0	$A \rightarrow C$
		$B \rightarrow C$
	1	$A \rightharpoondown^1 B\ \{(t_3\|t_4)\}$
		$B \rightharpoondown^1 A\ \{t_3\}$
		$C \rightharpoondown^1 B\ \{t_4\}$
	2	$C \rightharpoondown^2 A\ \{(t_1,t_2)\|(t_3,t_4)\}$

R_1	A	B	C
t_1	0	1	1
t_2	0	1	1
t_3	1	$\perp$	1
t_4	1	0	1

F_1	Deg.	FDs discovered from R_1
	0	$A \rightarrow C$
		$B \rightarrow C$
		$B \rightarrow A$ (fake)
	1	$A \rightharpoondown^1 B\ \{(t_3\|t_4)\}$
	2	$C \rightharpoondown^2 A\ \{(t_1,t_2)\|(t_3,t_4)\}$
		$C \rightharpoondown^2 B\ \{(t_3,t_4)\}$ (ghost)

R_2	A	B	C
t_1	0	1	1
t_2	0	1	$\perp$
t_3	1	$\perp$	1
t_4	1	0	1

F_2	Deg.	FDs discovered from R_2
	0	$B \rightarrow A$ (fake)
	1	$A \rightharpoondown^1 B\ \{(t_3\|t_4)\}$
		$C \rightharpoondown^1 A\ \{t_1\}$ (fake)
		$B \rightharpoondown^1 C\ \{(t_1\|t_2)\}$ (ghost)
		$A \rightharpoondown^1 C\ \{(t_1\|t_2)\}$ (ghost)
	2	$C \rightharpoondown^2 B\ \{(t_1,t_3)\|(t_1,t_4)\|(t_3,t_4)\}$ (ghost)

Figure 2: R_0 is a relation of binary values for attributes A, B, and C; R_1 is the same relation but with one missing value ($\perp$) randomly injected; similarly, R_2 has two missing values randomly injected. Exact and approximate FDs are computed from R_0, R_1, and R_2 and reported in tables F_0, F_1, and F_2 below their respective relations.

R_3	A	B	C
t_1	0	1	1
t_2	0	$\perp$	$\perp$
t_3	1	$\perp$	1
t_4	1	0	1

$F_3^{=}$	Deg.	FDs discovered from R_3
	0	$\emptyset$
	1	$B \rightharpoondown^1 A\ \{(t_2\|t_3)\}$
		$A \rightharpoondown^1 C\ \{(t_1\|t_2)\}$ (ghost)
		$B \rightharpoondown^1 C\ \{(t_2\|t_3)\}$ (ghost)
		$C \rightharpoondown^1 A\ \{t_1\}$ (fake)
	2	$A \rightharpoondown^2 B\ \{(t_1\|t_2),(t_3\|t_4)\}$ (ghost)
		$C \rightharpoondown^2 B\ \{(t_1,t_3)\|(t_1,t_4)\|(t_3,t_4)\}$(ghost)

$F_3^{\neq}$	Deg.	FDs discovered from R_3
	0	$B \rightarrow A$ (fake)
		$B \rightarrow C$
	1	$A \rightharpoondown^1 C\ \{(t_1\|t_2)\}$ (ghost)
		$C \rightharpoondown^1 A\ \{t_1\}$ (fake)
	2	$A \rightharpoondown^2 B\ \{(t_1\|t_2),(t_3\|t_4)\}$(ghost)
		$C \rightharpoondown^2 B\ \{(t_1,t_3)\|(t_1,t_4)\|(t_3,t_4)\}$ (ghost)

Figure 3: R_3 is the original relation R_0 with three missing values. Exact and approximate FDs are computed from R_3 and reported for the two cases of NULL semantics: `NULL-EQ` and `NULL-NOT-EQ`, noted $F_3^{=}$ and $F_3^{\neq}$ respectively. Fake FDs are labeled and represented in red, ghost FDs in green.

discovery process, or the methods assume that one of the two default semantics for handling NULL values is systemically applied. As illustrated in the example, both working assumptions suffer from the discovery of spurious FDs. Understanding the various ways in which spurious FDs appear is extremely important, as FDs have a number of applications in data management. We investigate precisely this phenomenon by first formally defining them and then developing a series of algorithms to quantify the genuineness of an FD.

3.2 Genuine, ghost, and fake exact FDs

For ease of exposition, we first define the notion of genuine, ghost and fake for exact FDs. Consider two versions of the relation R: R_{clean} that is clean/complete and R_{dirty} that is a noisy version of R_{clean} with missing values. Let F_{clean} be the set of FDs discovered over R_{clean} while F_{dirty} be the set of FDs discovered over R_{dirty} under a suitable NULL semantics (such as `skiptuple`, `NULL-NOT-EQ`, or `NULL-EQ`). We can see that the set of FDs will not be identical. We partition the set of FDs in $F_{clean} \cup F_{dirty}$ into the following groups.

Same FDs: These are exact FDs that are present in both FD sets, *i.e.*, $F_{same} = F_{clean} \cap F_{dirty}$;

Fake FDs: These are exact FDs that are discovered from R_{dirty} but not from R_{clean}, *i.e.*, $F_{fake} = F_{dirty} \setminus F_{clean}$. We consider them as false positive FDs – FDs that could be considered valid but are not;

Ghost FDs: These are exact FDs that are discovered in F_{clean} but "disappeared" in F_{dirty}, *i.e.*, $F_{ghost} = F_{clean} \setminus F_{dirty}$. These are false negative FDs – candidate FDs that are considered non-FDs but are indeed valid FDs from F_{clean};

Genuine FDs: Using the notations above, we can see that genuine exact FDs can be reconstructed as $F_{genuine} = F_{same} \cup F_{ghost}$.

3.3 Genuine, ghost, and fake approximate FDs

Let us now extend these definitions for approximate FDs. Recall that approximate FDs are associated with an approximation degree that measures how many tuples need to be removed such that an AFD can become an exact FD. A simplistic approach would be to consider all AFDs with an approximation degree less than a certain threshold as equivalent to exact FDs and reuse the prior definition of genuineness for exact FDs. However, finding an appropriate threshold is very challenging. We provide a generic definition that takes into account both the degree of cleanliness of data and the degree of approximation.

Given a dataset with $x\%$ of missing values, we denote with $F_{x,y}$ the set of valid FDs with approximation degree of exactly y. For example, $F_{0,0}$ denotes the set of *exact* FDs discovered from the *clean* dataset, and $F_{10,3}$ denotes the set of approximate FDs with degree 3 discovered from the dataset containing 10% missing values. Let $F_{x',y}$ be the set of valid FDs discovered from a version of the dataset ($x' > x$) with more missing values with the same approximation degree y. For notational convenience, we denote both the degree of dirtiness and the corresponding dataset with that degree of dirtiness using x.

Definition 1. **Same FD set.** *Given a fixed approximation degree y, $SAME^{x_0}_{x,y}$ is the set of FDs discovered from a dirty dataset x (with $x > 0$) that also appear in the clean version x_0 of the dataset:*

$$SAME^{x_0}_{x,y} = SAME(F_{x_0,y}, F_{x_0,y}) = F_{x_0,y} \cap F_{x,y} \qquad (1)$$

Definition 2. **Fake FD set.** *Given a maximum approximation degree y, $FAKE^{x_0}_{x,y}$ is the set of FDs discovered from a dirty dataset ($x > 0$) that were not valid in the clean dataset x_0:*

$$FAKE^{x_0}_{x,y} = FAKE(F_{x_0,y}, F_{x,y}) = F_{x,y} \setminus \bigcup_{\forall y_0 \leq y} F_{x_0,y_0} \qquad (2)$$

Definition 3. **Ghost FD set.** *$GHOST^{x_0}_{x,y}$ is the set of FDs discovered from a dirty dataset ($x > 0$) that are valid in the clean dataset x_0 with a certain approximation degree $y_0 \geq 0$, but exist only with a higher approximation degree $y > y_0$ in the dirty dataset:*

$$GHOST^{x_0}_{x,y} = GHOST(F_{x_0,y}, F_{x,y}) = F_{x,y} \cap \bigcup_{\forall y_0 < y} F_{x_0,y_0} \qquad (3)$$

For generalization, we denote $F_{x,*}$, the FD set discovered from a dataset with x% of missing values for all approximation degrees. We denote $F^{=}_{x,y}$ and $F^{\neq}_{x,y}$, the FD sets discovered with `NULL-EQ` and `NULL-NOT-EQ` semantics respectively. Finally, in presence of the clean dataset ($x_0 = 0$), we define genuine FDs as follows.

Definition 4. **Genuine FD set.** *Given two versions of the same dataset, one containing x% incomplete values ($x > 0$) and the clean version of the dataset ($x_0 < x$), $GENUINE^{x_0}_{x,y}$ can be computed as the union of FDs from $SAME^{x_0}_{x,y}$ and $GHOST^{x_0}_{x,y}$.*

Intuitively, genuine FDs discovered from a dirty dataset x are the FDs that hold in the clean version x_0 of the dataset. But generally, we do not have access to the clean dataset but rather to a "less dirty" version of the dataset (where $0 < x_0 < x$). Our final goal is then to find the set of genuine FDs, noted $GENUINE_{x,*}$ for all approximation degrees discovered from the dirty dataset x using x'_0, one of the possible "cleaner" versions of the dataset ($0 < x'_0 < x$).

4. ESTIMATING FD GENUINENESS

In this section, we introduce a generic notion to quantify the genuineness of an FD and propose an efficient algorithm to compute it.

4.1 Identifying genuine FDs

As we described previously, the set of FDs that are discovered from an incomplete relation can be *genuine*, *ghost*, or *fake*. Naively using all of the discovered FDs, irrespective of whether they are genuine or not, might be sub-optimal in applications, such as query optimization and data cleaning. A data analyst would prefer to utilize only the FDs that are either genuine or very likely to be genuine. Our objective is to identify a measure that can be used to quantify the "genuineness" of a given FD. Informally, we would expect for a genuine FD to have a higher genuineness score than non-genuine FDs. Assuming the availability of such a measure, the analyst can use the following procedure to identify the set of FDs that are likely to be genuine:

1. Run some exact FD discovery algorithm on the "clean" subset of R that does not have any NULL values; The set of discovered FDs will be a superset of all genuine FDs and can contain both *ghost* and *fake* FDs;

2. Compute the genuineness score for each of the discovered FDs;

3. Prune the list of discovered FDs based on some top-k or a domain-specific threshold whereby all FDs with genuineness score above that threshold are considered *genuine*.

4.2 Genuineness for probabilistic imputation

We now describe an approach to compute genuineness of an FD that subsumes many of the strategies used for handling datasets with incomplete tuples as described in Section 1. A common strategy for handling missing values is *imputation*. Imputation refers to the statistical process that replaces missing data with substituted values. There has been extensive work in statistics to perform imputation in a robust way [33]. Often, imputation strategies seek to replace missing data for a given attribute with an estimated value based on the values of other attributes/tuples. For example, a simple imputation strategy for numeric data is to replace missing data with the median value of all the values of that attribute. Alternatively, one can use a regression-based approach to estimate the value of an attribute given the values of other attributes. This approach also subsumes various NULL semantics, such as `NULL-EQ` and `NULL-NOT-EQ`. To see why, one can simulate `NULL-EQ` by imputing with NULL values for a given attribute to the same value. Alternatively, one can simulate `NULL-EQ` by imputing all NULL values for a given attribute to a different value. ?

We now consider a general probabilistic imputation approach that, for each missing data, gives a probability distribution over the various values that can be taken. This approach generalizes most of the main imputation strategies and allows us to exploit connection to well-studied area of probabilistic databases. Intuitively, in a probabilistic database, each tuple (or an attribute) is associated with a probability distribution such that it can take different values with different probabilities. A possible world is a specific instantiation of the probabilistic database where each tuple takes a value based on the probability distribution associated with the tuple. As an example, consider a probabilistic database with two tuples t_1 and t_2 that can take two and three values respectively. Then there are totally six possible worlds (by Cartesian product). In this paper, we consider the scenario where the probability distribution is defined over the entire tuple. Note that this approach is more general than the one where the probability distribution is defined over attributes as the former can handle correlated attributes. Table 1 shows a probabilistic imputation based on relation R_3 from Figure 3, where the probabilities are chosen arbitrarily for expository purposes. For example, the third line of the table can be interpreted as: B and C values of tuple t_3 will be imputed as $t_3[B] = 0$ and $t_3[C] = 1$ with probability 0.2 and $t_3[B] = 1$ and $t_3[C] = 1$ with probability 0.8.

Intuitively, the probabilistic imputation associates with each incomplete tuple a set of possible imputed/complete tuple values it can take with the corresponding probability. This is equivalent to an uncertain tuple in a probabilistic database that is associated with a probability distribution.

The probability distribution is defined over a complete tuple. {i.e joint dist. over columns}

Table 1: Tuple-level probability distribution for imputation over relation R_3 of the illustrative example.

	A	**(B , C)**
t_1	0	(1,1)
t_2	0	$\{(0,0) : 0.12; (0,1) : 0.18; (1,0) : 0.28; (1,1) : 0.42\}$
t_3	1	$\{(0,1) : 0.2; (1,1) : 0.8\}$
t_4	1	(0,1)

We also make the tuple independence assumption whereby individual tuples are imputed independently. This is a standard assumption in both probabilistic imputation and probabilistic databases.

Given the setup above, we can now define the genuineness score of an FD as the sum of probabilities of all the possible worlds in which the FD holds. Note that there are 8 possible worlds in the example of Table 1 (four for t_2 and 2 for t_3). Given an FD $X \to A$, one can compute its genuineness by enumerating all possible worlds, evaluating if the FD holds in that world and then simply summing up the probabilities of all worlds where it does. We can see that this definition of genuineness generalizes both the *strong* and *weak* FDs [25]. A strong FD is one that holds in all possible worlds while weak FD holds in at least one possible world. Based on our genuineness score definition, we can see that strong FDs have a genuineness of 1 while weak FDs have a genuineness score > 0. Naturally, our proposed approach provides a granular way to identify the set of genuine FDs. For example, the analyst might use a custom threshold (say .9) and consider only the FDs with a genuineness score above this threshold.

But you can't expand all Null values → 2^n. So they prune if some non possible FD comes, like for B → A if $\boxed{\begin{array}{c|c} 0 & 0 \\ \hline 0 & 1 \end{array}}$ → no prune.

4.3 Estimating probabilistic genuineness

Let us consider how to compute the genuineness score of an FD in a probabilistic imputation setting.

Complete enumeration. The simplest approach basically enumerates all possible worlds. We then sum up the probability of all the worlds where it holds and return it as the genuineness score of the FD. Note that this approach is exact and returns the accurate genuineness score. However, this is a very expensive algorithm as there might be an exponential number of possible worlds in a real-world dataset.

Note that one can generate the possible worlds in a straightforward manner. The deterministic tuples that have no NULL values exist in all the possible worlds while the tuples with NULL exist with appropriate imputation probability. As an example, there are 8 possible worlds for the example in Table 1. Tuples t_1 and t_4 exist in each of them. Tuples t_2 and t_3 take values from the Cartesian product of all possible imputed values. So in possible world w_1, $t_2[B] = 0, t_2[C] = 0$ and $t_3[B] = 0, t_3[C] = 1$. Since this is a tuple-independent probabilistic database, this occurs with probability 0.12×0.2. The last possible world w_8 has $t_2[B] = 1, t_2[C] = 1$ and $t_3[B] = 1, t_3[C] = 1$ with probability 0.42×0.8. One can enumerate other possible worlds and compute its probability in a systematic manner.

Efficient enumeration. One can leverage prior work on efficient inference over probabilistic databases [9,10,22] to propose a more efficient algorithm that can compute the *exact* genuineness score by avoiding the enumeration of irrelevant worlds where the FD does not hold. Consider an FD $X \to A$ and an arbitrary tuple t_i. Intuitively, we perform two major pruning steps. First, we can notice that when considering the possible worlds where we imputed $t_i[X] = V_X$ and $t_i[A] = V_A$ for some $V_X \in Dom(X), V_A \in Dom(A)$, we no longer need to consider all possible worlds where $t_j[X] = V_X$ and $t_j[A] \neq V_A$ where $j > i$. In other words, the entire set of possible worlds where $t_i[X] = t_j[X] = V_X$, $t_i[A] = V_A$ and $t_j[A] \neq V_A$ will have a contribution of 0 to the genuineness score computation and can be readily pruned. Second, if all the values in a given tuple comply with the FD, then the genuineness score computation does not change whether the tuple is picked or not as its contribution is 1.

Algorithm 1 shows the pseudo-code. Given an FD $X \to A$, we use the term constraints loosely to denote the set of (X, A) pairs that are valid in the given partial probable world. For example, consider a tuple t_i with $t_i[X] = V_X$ and $t_i[A] = V_A$ where $V_X \in Dom(X), V_A \in Dom(A)$. Then the pair (V_X, V_A) acts as a constraint (denoted by C in Algorithm 1) whereby all possible worlds where another tuple t_j is imputed with same value for X but different value for A is invalid. Please refer to [10] for additional details.

Algorithm 1 Estimate_Genuineness_Score

```
Input: Imputed database $D$, FD $f$, Set of constraints $C$
Output: Genuineness score $P$ of $f$
$P = 0$
$t$ = Next tuple to process from $D$
if $t$ does not violate $f$ and $C$ then
    return Estimate_Genuineness_Score($D \setminus t$, $f$, $C$)
end if
                    for tuple t, column LHS.
for each distinct possible (t[LHS], t[ RHS]) combination in imputed $t$ do
    if Possible tuple (t[LHS], t[RHS]) does not violate $C$ then
        Add constraint (t[LHS], t[RHS]) to $C$
        result = Estimate_Genuineness_Score($D \setminus t$, $f$, $C$)
        $P = P$ + Prob( t[LHS], t[RHS] ) × result
        Remove t[LHS], t[RHS] from $C$
    end if
end for
return $P$
```

Example 1. *Consider Table 1 and try to estimate the genuineness score of candidate FD $A \to B$. We can see that tuples t_1 and t_4 are deterministic and impose the "constraints" $\{(A = 0, B = 1), (A = 1, B = 0)\}$. Hence, we need to consider only the set of possible worlds where this set of constraints hold. Let us now consider tuple t_2. Since $t_2[A] = 0$, $t_2[B]$ has to be 1 (otherwise it violates the constraints and has a probability of 0). We can see that $t_2[B] = 1$ occurs with probability 0.7 (0.28 + 0.42). Similarly, $t_3[B]$ can take only the value of 0 that occurs with probability 0.2. The assignment for t_2 and t_3 happens independently with probability $0.7 \times 0.2 = 0.14$. Hence the genuineness score of $A \to B$ is 0.14.*

Example 2. *Let us compute the genuineness score of the FD $AB \rightarrow C$. Once again, t_1 and t_4 are deterministic and impose the "constraints" $\{(A = 0, B = 1, C = 1), (A = 1, B = 0, C = 1)\}$. Let us process t_3 next. We can see that both the possible options $(A = 1, B = 0, C = 1)$ and $(A = 1, B = 1, C = 1)$ do not violate any constraints. Hence the entire tuple does not violate any of the current set of constraints. For each option, we add the assignment to the set of constraints and recursively process tuple t_2. Based on the constraints above, we can see that t_2 can take only three of the possible options without violating the constraint. By adding up the respective probabilities, we get the genuineness score of $AB \rightarrow C = 0.12 + 0.18 + 0.42 = 0.72$.*

Complexity Analysis. The efficiency of Algorithm 1 stems from the fact that it avoids enumerating possible worlds where a given FD does not hold. Note that the complete enumeration based approach requires to consider all possible worlds which is exponential in the number of tuples in the worst case. In contrast, Algorithm 1 is only exponential in the cardinality of the domain of the attributes. When the number of attributes involved in the FD is small or when they have low domain cardinality, such as Gender, this approach is orders of magnitude faster than complete enumeration. Nevertheless, in the worst case, both approaches have exponential complexity.

5. EFFICIENTLY APPROXIMATING FD GENUINENESS

In this section, we propose two distinctive approaches based on sampling and likelihood computation to speed up computation of FD genuineness.

5.1 Sampling-based genuineness computation

In a number of real-world applications, the user does not necessarily need to compute the genuineness of an FD exactly and an approximate value suffices for decision making purposes. Under this circumstance, the user can quickly approximate the genuineness score by sampling the possible worlds.

Monte Carlo sampling of possible worlds. In this approach, one can adapt the Karp-Luby algorithm for approximate model counting that is used for inference over probabilistic databases [9, 22]. Intuitively, we generate different possible worlds in proportion to their likelihood. We then compute the genuineness score as the weighted ratio of the likelihood of all the generated worlds where the FD held to the likelihood of all the generated worlds. In contrast to the complete enumeration approach, we do not enumerate all possible worlds. Instead we generate a sample of possible worlds and compute the genuineness score for each FD from the sample.

When the size of the sampled possible worlds is large enough, the ratio converges to the correct genuineness score with high probability. Specifically, [9, 22] showed that if we run the experiment for $N \geq \frac{4n}{\varepsilon^2} \ln \frac{2}{\delta}$, we can guarantee that the probability that the generated estimate being off by more than ε is less than δ. For example, if there are $n = 100$ tuples and we want the genuineness estimate to be within 0.1 of the true value at least 95% of the times, then we need to generate at least 29,778 possible worlds.

Furthermore, one can generate a confidence interval during the execution of the algorithm and can terminate it when the confidence is satisfactory. After sampling N possible worlds with $0 \leq \delta < 1$, we can guarantee [9,22] that the estimated genuineness score $\tilde{p}$ relates with accurate genuineness score p as follows:

$$P(\tilde{p} \leq (1 - \delta)p) \leq \exp\left(\frac{-N \times p \times \delta^2}{2}\right). \quad (4)$$

One can see that the runtime complexity of the algorithm is parameterized by the number of sampled possible worlds N. Since one can evaluate whether a given FD holds in $O(n^2)$, the overall time complexity is $O(N \cdot n^2)$.

5.2 Likelihood as genuineness

An alternate approach to speed up genuineness computation is to limit the expressiveness of the imputation. A number of commonly used imputation and repair strategies are often frequency-based (the more frequent a value occurs, more likely it is to be correct). If one adopts such an imputation strategy, one can design a linear time algorithm to efficiently compute the genuineness score.

Given a candidate FD $X \rightarrow A$, we can define its genuineness as the "likelihood" that it is correct. FDs that are more likely would have a higher genuineness score. Note that if the FD $X \rightarrow A$ is indeed genuine, we would like its genuineness score (and hence its likelihood) to be 1. However, due to incomplete data, there might be some tuples that have different values of A. Hence, a natural way to define the likelihood of a FD is to compute the fraction of tuples for each distinct value of X where the FD holds. In the following, we present two approaches adapted from [34] to compute efficiently genuineness scores per value and per tuple.

PerValue approach. Given a FD $X \rightarrow A$ and a NULL semantics, we begin by computing the likelihood that the FD holds for each distinct value of X. Consider an arbitrary value $V_X \in Dom(X)$. For all the tuples that have $t[X] = V_X$, we identify the value V_A that occurs in the maximum number of times. The likelihood that FD $X \rightarrow A$ holds for the value V_X can be computed as

$$Lik(X \rightarrow A, V_X) = \frac{|V_X, V_A|}{|V_X|} \quad (5)$$

where $|V_X, V_A|$, $|V_X|$ are the number of tuples that have $t[X] = V_X, t[A] = V_A$, and $t[X] = V_X$, respectively.

Note that $Lik(X \rightarrow A, V_X)$ is for a specific value V_X. We can compute the likelihood for a FD as the average of the likelihood values for each distinct value V_X. Formally, the genuineness score is computed as

$$Genuineness_{PV}(X \rightarrow A) = \frac{\sum_{V_X \in Distinct(X)} Lik(X \rightarrow A, V_X)}{|Distinct(X)|} \quad (6)$$

where $Distinct(X)$ returns all distinct values of X that occur in the relation R.

PerTuple approach. The PerValue approach, while intuitive, has a subtle issue. Consider two groups of tuples for arbitrary values V_X and V_Y. Let $|V_X| = 1,000$ and $|V_Y| = 10$ and assume that $|V_X, V_A| = 800$ and $|V_Y, V_A| = 8$. Using Equation 5, the likelihood for both V_X and V_Y are 0.8. Intuitively, we might want to give higher weight to V_X than V_Y. This can be achieved by weighting the likelihood by the frequency of each distinct value V_X. This results in a PerTuple definition of genuineness score computed as

$$Genuineness_{PT}(X \rightarrow A) = \frac{\sum_{V_X \in Distinct(X)} |V_X, V_A|}{\sum_{V_X \in Distinct(X)} |V_X|} \quad (7)$$

6. EXPERIMENTS

In this section, we report on our experimental results. First, we expose the phenomenon of fake and ghost FDs when missing values are injected in a clean dataset and we show how missing values can distort the FD discovery result (Section 6.2). Second, we apply our algorithms for computing the genuineness score of FDs from various real-world datasets that we have artificially polluted with missing values, and we report the quality performance evaluation of our approach given the true labels of FD discovered from the clean version of the datasets (Section 6.3). In particular, we observed: (1) Increasing the percentage of missing values in LHS attributes of FDs generates fake FDs both for NULL-NOT-EQ semantics and skiptuple (Section 6.2.B); (2) Increasing the percentage of missing values in RHS attributes of FDs generates fake FDs with NULL-EQ semantics or skiptuple but ghost FDs for NULL-NOT-EQ (Section 6.2.C); (3) PerValue (**PV**) and PerTuple (**PT**) approximations of the genuineness score have the highest quality performance to discover genuine FDs with NULL-EQ semantics. skiptuple is not a good strategy to discover genuine FDs (Section 6.3.A). We show that our approach is robust and can perform well even under a worst case imputation (Section 6.3.B). Experiments on a real-world Sensor dataset show that our FD-scoring methods can find 100% of genuine FDs that would have been obtained by multiple imputation strategies in very reasonable time, which offers a significant gain over pre- and post-processsing efforts for FD discovery (Section 6.4). Due to space limitations we did not include experiments on runtime performance. However, the results directly follow the complexity analysis of various algorithms proposed in Sections 4 and 5.

6.1 Experimental setup

Parameters of the study: In each experiment, we vary (1) the dataset and the characteristics of the discovered FDs in terms of number, set of attributes in LHS and RHS, and approximation degree; (2) the number and distribution of missing values; and (3) the considered NULL semantics: NULL-NOT-EQ, NULL-EQ, or skiptuple, and (4) the threshold to select the top-k genuine FDs ($k = 10, 20, 30$ and 100% of the total number of FDs discovered).

Datasets. We used five real-world datasets: four are selected from the UCI machine learning repository [26] and one Sensor dataset from Intel Berkeley Research lab[1]. The first four datasets (used in Section 6.2) are originally complete, i.e., without any missing values. The Sensor dataset (used in Section 6.4) includes missing values. As shown in Table 2, they vary in the number of columns, rows, and discovered FDs, and cover a wide variety of topics, and they are representative in terms of distributional characteristics and distinctness of attribute sets. We injected a varying percentage of missing value from 5% to 40% in the dataset attributes using one of the three modes: UNIFORM, PARETO, and TARGET. For UNIFORM mode, we distribute the random injection of missing values uniformly over the set of attributes. For PARETO mode, we inject randomly 20% of the missing values in 80% of the attributes and 80% in the remaining 20% of the attributes. Using PARETO mode, we study the impact of a realistically unbalanced distribution of missing values across the attributes and how it can affect the FD discovery (e.g., causing more ghost and fake FDs). Using TARGET mode, we select a subset of attributes involved either in LHS or RHS for a set of targeted FDs. For each originally complete dataset (×4), each missing value percentage (×6), and each distribution mode (×3), we generate 10 polluted versions to finally obtain $4 \times 6 \times 3 \times 10 = 720$ datasets.

FD discovery: The exact and approximate FDs were discovered from all datasets using the original implementation of FUN algorithm [29]. Once missing values have been injected, we re-ran FD discovery for each NULL semantics and for Skiptuple, the case where the tuples containing missing values are skipped in the FD discovery process. We finally analyze $3 \times 720 = 2,160$ FD sets for the experiments of Sections 6.2 and 6.3 and 18 datasets for Section 6.4.

Quality performance evaluation: For the first sets of experiments in Sections 6.2 and 6.3, we used the ground truth obtained by discovering FDs from the originally clean datasets. We compared the set of FDs discovered before and after injection of missing values. We used the true labels of FDs to compute the traditional measures of precision (P), recall (R), and F1-measure for k percent of the discovered FD size defined as: $P = |true\ Genuine\ FDs|/|Top\text{-}k\ FDs|$, $R = |true\ Genuine\ FDs|/|AllFDs|$, and $F1_k = 2PR/(P + R)$. For the case study in Section 6.4, since we do not have access to the ground truth, we used the set of FDs discovered from datasets obtained from three most commonly used imputation strategies as a baseline and report the Jaccard coefficient.

Data Storage and system setup: We store all discovered FDs with their approximation degrees as well as the attribute sets' distinctness in a MySQL database and perform SQL queries to extract the information we report hereafter. We perform all experiments on a Dell XPS machine with an Intel Core i7-7500U quad-core, 2.8 GHz, 16 GB RAM, Windows 10 64-bit with g++ (GNU).

6.2 Ghost and fake FDs phenomenon

A. Impact of NULLs uniformly distributed in LHS and RHS. In this experiment, our goal is to show that ghost and fake FDs exist and have considerable impact on the validity of FD discovery results. First, we randomly injected increasing percentages of missing values uniformly in the attribute list for each clean version of the real-world datasets described in Table 2. We discover the sets of FDs for the full range of approximation degrees in the original, clean version of the dataset as well as from each polluted version in the two NULL semantics and the Skiptuple cases.

Figure 4 shows, when increasing the percentage of missing values (X-axis) in the Abalone dataset, two Jaccard coefficients (in Y-axis) averaged over the 10 polluted versions: The first Jaccard coefficient (same_approxdeg as dashed line) is computed as the fraction of the number of common FDs discovered both in the clean dataset and each polluted dataset version for exactly the same approximation degree over the total number of FDs discovered in both datasets.

[1] http://db.csail.mit.edu/labdata/labdata.html

Table 2: Clean versions of real-world datasets with the number of (A)FDs discovered (with approximation degree α)

Datasets	[#]Att.	[#]Rows	[#]Distinct (min;max)	[#]Missing	[#]FDs						
					$\alpha = 0$	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha \geq 5$	[#]Total
Iris	5	150	(3;43)	10%-40%	5	2	1	1	7	59	80
Abalone	9	4,177	(3;2,429)	10%-40%	783	219	122	57	56	1,067	2,313
Computer hardware	9	209	(15;209)	10%-40%	3,046	193	199	168	92	1,422	5,129
Glass identification	10	214	(6;214	10%-40%	8,624	1,156	536	166	84	687	11,263
Sensor	8	2,313,681	(137;10,283)	96,733	`Skiptuple`						
Sensor 10 Bins			→ 10		397	29	10	14	11	563	1,024
Sensor 100 Bins			→ 100		432	40	10	0	3	539	1,024
Sensor 1000 Bins			→ 1000		427	44	7	0	3	543	1,024

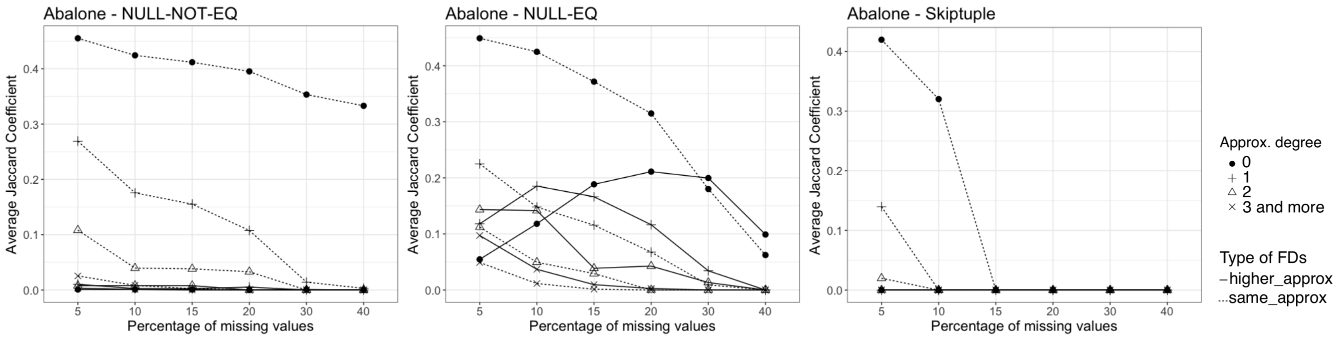

Figure 4: Jaccard coefficients with increasing percentage of missing values for Abalone dataset. It measures the similarity between the FD sets discovered from the original, dirty datasets for the same or higher approximation degrees in the 3 cases of NULL semantics: `NULL-NOT-EQ`, `NULL-EQ`, and `skiptuple`.

It represents the same FDs as defined in Equation (1). The second Jaccard coefficient (`higher_approxdeg` as solid line) represents the fraction of common FDs that have an approximation degree in each dirty version that is higher than in the clean dataset over the total number of FDs discovered in both datasets. In this figure, we can see that non exact FDs are the most impacted by the ghost and fake phenomenon. The more missing values are introduced, the more dissimilar sets of FDs with same approximation degree are obtained. Skipping tuples with missing values for FD discovery is clearly not a good option to preserve genuine FDs as the Jaccard coefficients tend to 0 when increasing the percentage of NULL values. We made the same observations of the phenomenon on the other datasets.

B. Impact of NULLs in LHS. To better understand the phenomenon at the attribute set level, we injected missing values with the PARETO mode. We obtained very similar figures to Figure 4 in the two cases of NULL semantics (not shown due to space limitation). To grasp the phenomenon at a finer grain, we used the TARGET mode over the least and most distinct attributes of the datasets. Figure 5 (Right) shows the approximation degree variation (Y-axis) with respect to the percentage of missing values (X-axis) injected in attribute $A10$ (the least distinct attribute) of Glass dataset for each FD having $A10$ in its `LHS` for each NULL semantics cases (similarly Figure 5 (Right) for `RHS`). In Figure 5(Left), an increasing percentage of missing values in `LHS` causes a dramatic drop of the approximation degree of all FDs both for `NULL-NOT-EQ` and `Skiptuple` (thus generating fake FDs), whereas for `NULL-EQ` semantics, targeted injection in `LHS` leaves the FDs' approximation degree intact irrespectively of the number of missing values injected.

C. Impact of NULLs in RHS. In Figure 5 (Right), 150 FDs (not listed due to space limitation) having $A10$ in the RHS are plotted for each NULL semantics. We observe clearly that the increase of their approximation degree is proportional to the increasing of the percentage of missing values for `NULL-NOT-EQ`, whereas it decreases significantly in the two other cases with a much steeper slope when the tuples are skipped than for `NULL-EQ`. In this case when more missing value are injected in `RHS`, depending on the NULL semantics, we can see either the generation of fake FDs (with approximation degrees getting lower for `NULL-EQ` and `Skiptuple`) or the disappearance of FDs (becoming ghost with approximation degrees getting higher for `NULL-NOT-EQ`).

Conclusions. We observe the same phenomenon in all polluted versions of all datasets. This corroborates our conclusions: (1) increasing the distinctness of `LHS` attribute set or adding more distinct missing values decreases the approximation degree of the corresponding FDs and more fake FDs will appear for `NULL-NOT-EQ` and `Skiptuple`; (2) increasing the distinctness of `RHS` attribute set makes the corresponding FDs become more and more approximate: more genuine FDs will disappear and become ghost FDs for `NULL-NOT-EQ`; (3) decreasing the distinctness of `RHS` or adding missing values with `NULL-NOT-EQ` and `Skiptuple` makes the corresponding FDs become less and less approximate and more fake FDs will appear.

6.3 Quality evaluation

In this set of experiments, we compute the genuineness scores proposed in Sections 4 and 5 and report quality performance as accuracy, recall, precision, and F1-measure.

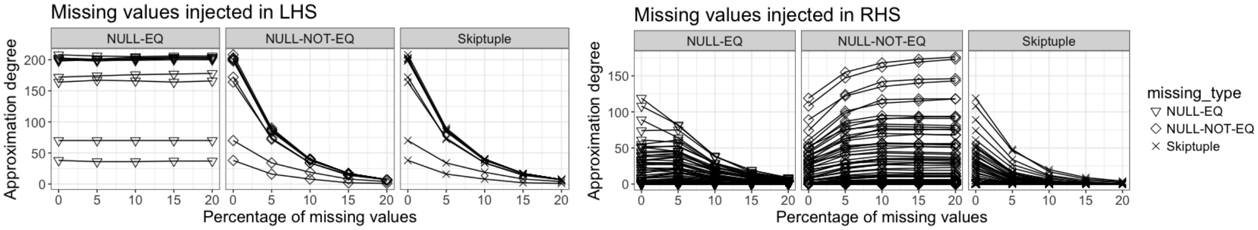

Figure 5: Variation of FD approximation degree w.r.t. the NULL semantics when missing values are injected in the least distinct attribute in LHS of the FDs (Left) and in RHS of FDs (Right) for the Glass dataset.

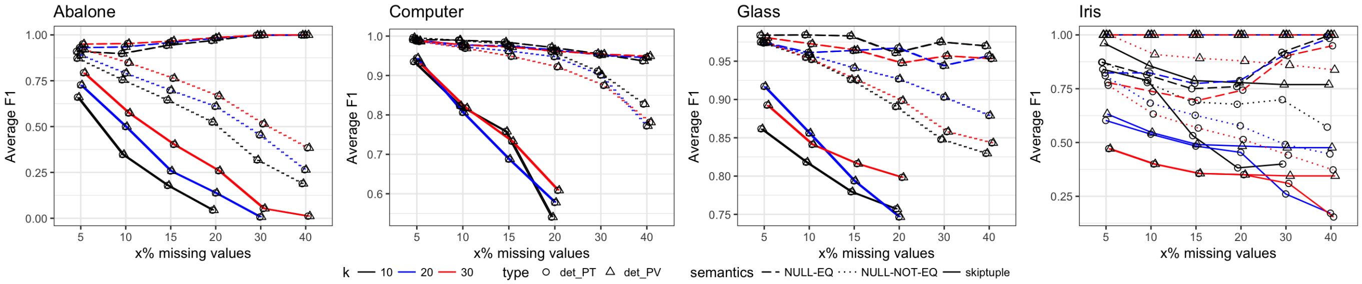

Figure 6: Average F1-measures of genuineness scores per value (PV) and per tuple (PT) over 10 runs for `Skiptuple`, `NULL-EQ`, and `NULL-NOT-EQ` semantics and 3 thresholds $k = 10, 20, 30\%$ of the number of FDs discovered from dirty dataset.

A. PerValue (PV) and PerTuple (PT) genuineness scores. For each polluted version of each dataset, precision, recall, and F1-measure are computed as follows: we select as genuine FDs, the ones having PV and PT scores greater than a predefined top-k threshold and we compare them with the true genuine FDs discovered from the clean version of each dataset. This procedure is repeated ten times for averaging the quality metrics. In Figure 6, we report averaged F1-measure of PV and PT scores for top-k genuine FDs discovered from the dirty datasets with $k = 10, 20$, and 30%. Precision and recall averages are presented in Figure 7. Overall, we observe that `Skiptuple` is the worst performer across all the datasets. All PV and PT scores obtained with the two NULL semantics outperform the scores obtained from `Skiptuple` to a significant extent across all datasets. PV and PT genuineness scores have very close F1-measures except for Iris where PV score reaches 1 despite the increasing percentage of missing values. We observe that `NULL-EQ` is consistently the best performer, regardless of the missing values percentage. With high percentages of missing values, the difference between scores obtained from `NULL-NOT-EQ` and `NULL-EQ` is greater by more than 10 to 20%. In conclusion, our PV score combined with `NULL-EQ` semantics can approximate correctly the genuineness of FDs across all datasets. We note that for all datasets, the computation time of PV and PT per FD is negligible (linearly with the dataset size: around 1 second for 100,000 tuples).

B. Sampling-based probabilistic genuineness score. The qualitative performance of sampling based approach is guaranteed to be identical to the PerTuple approach when frequency based probabilistic imputation is used. Instead, we highlight the robustness of the sampling based approach by performing a worst-case uniform imputation and show that it still achieves meaningful results. In uniform imputation, each value in the domain of an attribute is equally likely to be imputed. For example, if the attribute has a domain cardinality of 100, then each of the possible values have 1% probability of being imputed. We now study, how our approach fares under this imputation for Glass dataset. Applying Eq. (4) with $\delta = .95$ and $\varepsilon = .2$ and $\varepsilon = .1$ requires sampling at least 15,931 and 63,724 possible worlds respectively. In Table 3, we report the quality metrics of top-10% FDs based on the probabilistic genuineness score (**GS**) computed using 10,000 to 70,000 worlds from exact minimal FDs from `Skiptuple` Glass dataset. As expected, both precision and recall decreases with increasing missing values. However, our approach has high precision but low recall - whereby we return few FDs but most of the returned FDs are genuine. Given the preponderance of database applications of FDs, returning FDs that are very likely to be genuine is indeed desirable. Once again, we caution that this result is for the absolute worst case of uniform imputation. If the imputation is reasonably accurate, then the precision/recall will comparable to the PerValue and PerTuple approaches.

Table 3: Averaged precision, recall, F1-measure of GS@10% for Glass dataset over 10 runs with 10,000 to 70,000 possible worlds.

Missing (%)	Nb Worlds	Precision	Recall	F1
5	10000	0.758	0.082	0.149
	20000	0.783	0.085	0.154
	70000	0.866	0.094	0.170
10	10000	0.642	0.078	0.139
	20000	0.642	0.078	0.139
	70000	0.556	0.068	0.122
20	10000	0.580	0.033	0.062
	20000	0.559	0.032	0.060
	70000	0.152	0.009	0.017

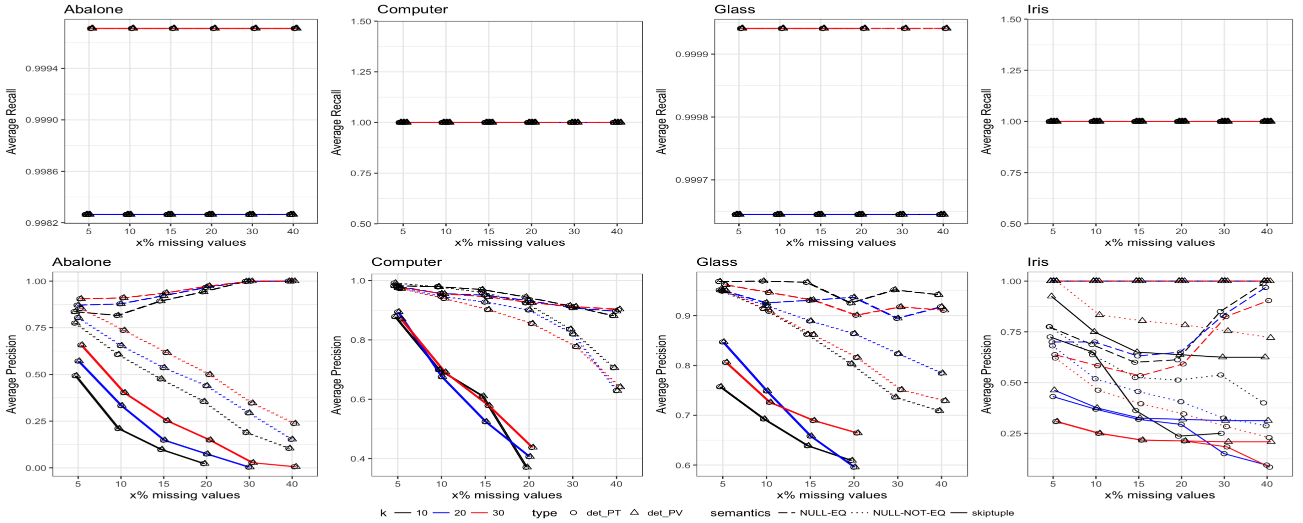

Figure 7: Average recall and precision of genuineness scores per value (PV) and per tuple (PT) over 10 runs for Skiptuple, NULL-EQ, and NULL-NOT-EQ semantics and 3 thresholds $k = 10, 20, 30\%$ of the number of FDs discovered from dirty dataset.

6.4 Case study on a real-world Sensor dataset

In this set of experiments, we use real-world data collected from 54 sensors deployed including 2,313,681 records identified by a timestamp and five relevant numerical attributes describing the conditions of the monitored rooms such as (date, hour, epoch, sensorId, voltage, temperature, humidity, light). The dataset includes 96,733 missing values with the distribution given in Table 4. A "1" in the table indicates a non-missing value and a "0" indicates a missing value. There are 2,219,802 observations with non-missing values, and for example, 3 observations with non-missing values except for the variables humidity and light (line 5 of the table). The original number of distinct values per attribute is given in parenthesis. We do not consider the spatio-temporal dimension of the dataset and focus on FD discovery in presence of missing values.

To study the effect of attribute cardinality on FD discovery, we transformed the dataset into three binned versions with 10, 100, and 1000 bins, respectively, for the five numerical attributes. We discovered FDs from each binned versions. Table 2 (three last lines) reports the numbers of FDs discovered from Skiptuple binned version. We can observe the overlaps of common FDs across various NULL semantics in the Venn diagram for 10 bins in Figure 8 (Left). Similar overlaps are observed for 100 and 1000 bins. Moreover, our intuition about the phenomenon of fake and ghost FDs is confirmed as we observed exact FDs that are present in Skiptuple but "disappear" with another NULL semantics, such as the FD epoch, sensorId,temperature,humidity → voltage exact in Skiptuple and NULL-NOT-EQ versions but with approximation degree 18 in NULL-EQ.

Next, we computed PV, PT, and GS scores and selected the top-k FDs with $k = 10, 20, 30$, and 100%. As in many similar application scenarios, we do not have access to the ground truth related to missing values, but a common technique is to apply statistical imputation methods. Therefore, we applied three imputation strategies to the original Sensor dataset, namely PMM, RI, and QUAD. PMM calculates imputations by predictive mean matching [33]. RI (Random Indicator) estimates an offset between the distribution of the observed and missing data using an algorithm that iterates over the response and imputation models. QUAD is a multivariate imputation technique based on estimating the squared terms [33]. We also applied the same binning strategies to the imputed datasets and discovered three FD sets respectively. In the absence of ground truth, we can reasonably take the assumption that common FDs across all imputed datasets can be considered as genuine FDs for our comparison purposes. Figure 8 (right) represents the overlaps and the set of 486 genuine FDs for imputed Bin 10 Sensor dataset (similar figures for Bin 100 and 1000 are observed). Finally, Figure 9 reports the Jaccard coefficient between FDs discovered using our top-k scoring-based methods for various NULL semantics and the set of genuine FDs as $F_{PMM} \cap F_{RI} \cap F_{QUAD}$. Our results show that, with only top-30% of PT- and PV-scoring results obtained from the FD set size of any NULL semantics, around 60% of the set of imputed-genuine FDs can be discovered.

PV and PT scores are computed simultaneously for the full Sensor dataset in approximately 21 seconds for 10 Bins, 20 seconds for 100 Bins, and 20 seconds for 1000 Bins for the three cases of NULL semantics and Skiptuple. GS score computation times are: 11 min and 35s, 3 hours 32 min, and 3 hours 26 min, respectively. As a conclusion, the user may choose many different ways to impute missing values and then discover FDs from imputed datasets. However, using our method and in particular PV score regardless of the NULL semantics, the user can obtain, with reasonable execution time, the set of genuine FDs instead of carefully selecting the imputation strategies, spending time for imputation and screening the FDs discovered from multiple imputed datasets.

7. RELATED WORK

Functional dependency is one of the most important type of integrity constraints and has been extensively studied by the research community. It has a number of applications,

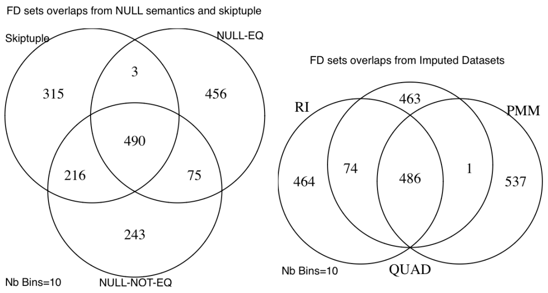

Figure 8: Venn diagrams representing the FD sets and their intersections in Sensor 10 Bins dataset between `Skiptuple`, `NULL-EQ` and `NULL-NOT-EQ` versions (Left) and imputation strategies (Right).

Table 4: Missing values distribution in Sensor data.

[#]Records	SensorId (61)	voltage (137)	temp. (10,283)	hum. (1,990)	light (143)
2,219,802	1	1	1	1	1
1	1	1	0	1	1
92,975	1	1	1	1	0
1	1	1	0	1	0
3	1	1	1	0	0
373	1	1	0	0	0
526	0	0	0	0	0
2,313,681	526	526	901	902	93,878

such as maintaining data quality [14], schema normalization [31], repair data inconsistencies [4, 6], etc. There has been extensive work on discovering exact FDs from complete/correct data [27, 30]. There has been a number of different formalisms to extend FDs to handle erroneous data inherent in real-world applications. Common approaches include approximate FDs [5] and conditional FDs [20, 21] whereby a FD holds on a subset of data instead of the entire dataset (please refer to [7] for a detail survey of relaxed definitions of FDs). There has been some work on probabilistic FDs that might hold on the data with some probability [10, 34]. Recently, there has been some proposals to extend the semantics of FDs under NULL markers [1, 2]. However, none of the FD mining algorithms questions whether the discovered FDs are genuine FDs or not. The usual working assumption is that FD discovery operates from a clean dataset. As an unfortunate consequence, existing FD discovery-based frameworks for data cleaning rely on the correctness of the discovered dependencies; cleaning rules based on matching dependencies [3] and constant or variable CFDs [15] may actually be erroneous (fake) and skip relevant dependencies (ghost).

There has been some research considering the effect of NULL values on constraints, namely on FDs [25] and on keys [23]. As in our work, the authors first acknowledge the presence of NULL values in typical datasets and explain their detrimental effects on enforcing constraints. They then introduce the notions of possible and certain FDs/keys (weak and strong FDs in [25]). A possible FD/key is one for which a possible world exists (*i.e.*, some instantiation of all NULL values with any non-NULL values). A certain FD/key is one that holds for all possible worlds. Both works then construct sound and complete axiom systems for such dependencies and the authors of [23] suggest an algorithm for the discovery of certain keys. In [24] the authors go a step further, by proposing an algorithm to discover approximate certain keys, *i.e.*, keys with NULL values that are still sufficient to identify tuples (certain), but may have some violating values (approximate). In a similar vein, certain FDs (with some violations) might be good candidates for genuine FDs. In contrast, we are interested in the behavior of FDs under changing cleanliness to then determine genuine FDs. Finally, some attempts have been made to solve the problem of inconsistency between data and their respect set of FDs. In [8], the authors developed an algorithm for FD repair and maintenance without overfitting the potentially erroneous data. But they did not consider NULL semantics within their cost model for both data and constraint repairs. Another method to maintain FD set was proposed in [28]. This method adds one or more attributes to an FD to repair it instead of changing the data. It estimates to what extent an FD is violated by the data using measures of confidence and goodness of an FD. However, the authors excluded attributes with NULL values from being involved in FDs.

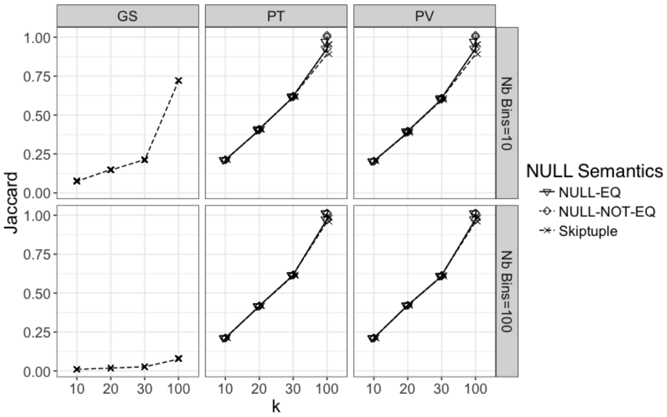

Figure 9: Jaccard coefficient between top-k FD sets based on GS, PV, and PT scores and the set of common FDs discovered from imputed datasets using PMM, RI, and QUAD imputation techniques for Sensor dataset with 10 and 100 Bins.

8. CONCLUSION AND FUTURE WORK

In this work, we studied how missing values may impair the final FD discovery results by causing the generation of spurious FDs and the omission of valid FDs in the same time. We formalized the notions of ghost, fake, and genuine functional dependencies. We proposed a probabilistic approach to quantify the genuineness of FDs and provide an efficient sampling-based computation of genuineness score with accuracy guarantee. We also proposed two algorithms to approximate the genuineness score of FDs based on per value and per tuple granularity levels that could be used by analysts to identify most promising FDs. Experimental results on real-world and semi-synthetic data show high accuracy and efficiency of our scoring model. For future work, we plan to extend and apply our technique to the particular case of "disguised" missing values when incorrect default values are misused in replacement of missing values and hardly detectable, which adds complexity into the detection of genuine FDs and anomaly semantics interpretation.

9. REFERENCES

[1] A. Badia and D. Lemire. Functional dependencies with null markers. *Comput. J.*, 58(5):1160–1168, 2015.

[2] A. Badia and D. Lemire. On desirable semantics of functional dependencies over databases with incomplete information. *arXiv preprint arXiv:1703.08198*, 2017.

[3] L. E. Bertossi. *Database Repairing and Consistent Query Answering.* Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.

[4] G. Beskales, I. F. Ilyas, and L. Golab. Sampling the repairs of functional dependency violations under hard constraints. *Proc. of the VLDB Endowment*, 3(1):197–207, 2010.

[5] T. Bleifuß, S. Bülow, J. Frohnhofen, J. Risch, G. Wiese, S. Kruse, T. Papenbrock, and F. Naumann. Approximate discovery of functional dependencies for large datasets. In *Proc. of the International Conference on Information and Knowledge Management (CIKM)*, 2016.

[6] P. Bohannon, W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis. Conditional functional dependencies for data cleaning. In *Proc. of the International Conference on Data Engineering (ICDE)*, pages 746–755, 2007.

[7] L. Caruccio, V. Deufemia, and G. Polese. Relaxed functional dependencies: A survey of approaches. *IEEE Transactions on Knowledge and Data Engineering*, 28(1):147–165, 2016.

[8] F. Chiang and R. J. Miller. A unified model for data and constraint repair. In *Proc. of the International Conference on Data Engineering (ICDE)*, pages 446–457, 2011.

[9] N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. *VLDB Journal*, 16(4):523–544, 2007.

[10] S. De and S. Kambhampati. Defining and mining functional dependencies in probabilistic databases. *arXiv preprint arXiv:1005.4714*, 2010.

[11] R. D. De Veaux and D. J. Hand. How to lie with bad data. *Statist. Sci.*, 20(3):231–238, 08 2005.

[12] DemandGen. Assessing the impact of dirty data on sales & marketing performance, `https://www.zoominfo.com/business/mktg/ebooks/dirtydataebook.pdf`, 2017.

[13] B. Efron. Missing data, imputation, and the bootstrap. *Journal of the American Statistical Association*, 89(426):463–475, 1994.

[14] W. Fan and F. Geerts. Uniform dependency language for improving data quality. *IEEE Data Engineering Bulletin*, 34(3):34–42, 2011.

[15] W. Fan and F. Geerts. *Foundations of Data Quality Management.* Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2012.

[16] W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis. Conditional functional dependencies for capturing data inconsistencies. *ACM Transactions on Database Systems (TODS)*, 33(2):6:1–6:48, 2008.

[17] W. Fan, J. Li, S. Ma, N. Tang, and W. Yu. Towards certain fixes with editing rules and master data. *VLDB Journal*, 21(2):213–238, 2012.

[18] Gartner. Dirty data is a business problem, not an it problem, `http://www.gartner.com/newsroom/id/501733`, 2007.

[19] A. Haug, F. Zachariassen, and D. van Liempd. The costs of poor data quality. *Journal of Industrial Engineering and Management*, 4(2):168–193, 2011.

[20] Y. Huhtala, J. Kärkkäinen, P. Porkka, and H. Toivonen. Efficient discovery of functional and approximate dependencies using partitions. In *Proc. of the International Conference on Data Engineering (ICDE)*, pages 392–401, 1998.

[21] J. Kivinen and H. Mannila. Approximate inference of functional dependencies from relations. *Theor. Comput. Sci.*, 149(1):129–149, 1995.

[22] C. Koch and D. Olteanu. Conditioning probabilistic databases. *Proceedings of the VLDB Endowment*, 1(1):313–325, 2008.

[23] H. Köhler, U. Leck, S. Link, and X. Zhou. Possible and certain keys for SQL. *VLDB Journal*, 25(4):571–596, 2016.

[24] H. Köhler, S. Link, and X. Zhou. Discovering meaningful certain keys from incomplete and inconsistent relations. *IEEE Data Engineering Bulletin*, 39(2):21–37, 2016.

[25] M. Levene and G. Loizou. Axiomatisation of functional dependencies in incomplete relations. *Theoretical Computer Science*, 206(1-2):283–300, 1998.

[26] M. Lichman. UCI machine learning repository `http://archive.ics.uci.edu/ml`, 2013.

[27] J. Liu, J. Li, C. Liu, and Y. Chen. Discover dependencies from data – a review. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 24(2):251–264, 2012.

[28] M. Mazuran, E. Quintarelli, L. Tanca, and S. Ugolini. Semi-automatic support for evolving functional dependencies. In *Proc. of the International Conference on Extending Database Technology (EDBT)*, pages 293–304, 2016.

[29] N. Novelli and R. Cicchetti. Fun: An efficient algorithm for mining functional and embedded dependencies. In *Proc. of the International Conference on Database Theory (ICDT)*, pages 189–203, 2001.

[30] T. Papenbrock, J. Ehrlich, J. Marten, T. Neubert, J.-P. Rudolph, M. Schnberg, J. Zwiener, and F. Naumann. Functional dependency discovery: An experimental evaluation of seven algorithms. *Proc. of the VLDB Endowment*, 8(10):1082–1093, 2015.

[31] T. Papenbrock and F. Naumann. Data-driven schema normalization. In *Proc. of the International Conference on Extending Database Technology (EDBT)*, 2017.

[32] S. Song, A. Zhang, L. Chen, and J. Wang. Enriching data imputation with extensive similarity neighbors. *Proc. of the VLDB Endowment*, 8(11):1286–1297, 2015.

[33] S. Van Buuren. CRC/Chapman & Hall, 2012.

[34] D. Z. Wang, X. L. Dong, A. D. Sarma, M. J. Franklin, and A. Y. Halevy. Functional dependency generation and applications in pay-as-you-go data integration systems. In *Proc. of the ACM SIGMOD Workshop on the Web and Databases (WebDB)*, 2009.