On Axiomatization and Inference Complexity over a Hierarchy of Functional Dependencies

Jaroslaw Szlichta\(^1\), Lukasz Golab\(^2\), and Divesh Srivastava\(^3\)

\(^1\) University of Ontario – Institute of Technology, Oshawa, Canada
\texttt{jaroslaw.szlichta@uoit.ca}
\(^2\) University of Waterloo, Waterloo, Canada
\texttt{lgolab@uwaterloo.ca}
\(^3\) AT&T Labs, New Jersey, USA
\texttt{divesh@research.att.com}

Abstract. Functional dependencies (FDs) have recently been extended for data quality purposes with various notions of similarity instead of strict equality. We study these extensions in this paper. We begin by constructing a hierarchy of dependencies, showing which dependencies generalize others. We then focus on an extension of FDs that we call Antecedent Metric Functional Dependencies (AMFDs). An AMFD asserts that if two tuples have \textit{similar} but not necessarily equal values of the antecedent attributes, then their consequent values must be equal. We present a theoretical foundation for AMFDs, including a sound and complete axiomatization as well as an inference algorithm. We compare the axiomatization of AMFDs to those of the other dependencies, and we show that while the complexity of inference for some FD extensions is quadratic or even co-NP complete, the inference problem for AMFDs remains linear, as in traditional FDs. We implemented our inference procedure and experimentally verified its efficiency.

Keywords: Functional Dependency Hierarchy, Axiomatization, Inference

1 Introduction

Poor data quality is a bottleneck to effective business decision-making. Big data initiatives are likely to take longer, cost more, and deliver fewer benefits without clean data. The ability to store data is no longer a problem: according to a survey of 586 senior executives conducted in June 2011 by the Economist Intelligence Unit (EIU) \cite{1}, less than 20\% indicated data storage as a problem; however more than 50\% rated data management tasks such as verifying, cleaning and reconciling data as problematic.

With the interest in data analytics at an all-time high, data quality has become a critical issue in research and practice. Integrity constraints, which specify the intended semantics and attribute relationships, are commonly used to characterize and ensure data quality \cite{6, 20}. In particular, Functional Dependencies (FDs), which have traditionally been used in schema design, have recently been extended for data consistency purposes. An FD asserts that if two tuples agree on the left-hand-side attributes, then they must also agree on the right-hand-side attributes. The idea behind various extensions of FDs is to replace strict equality with some notion of \textit{similarity}, either on the left-hand-side (see, e.g., Matching Dependencies \cite{7, 5, 9}), on the right-hand-side (see, e.g., Metric Functional Dependencies \cite{13} and Sequential Dependencies \cite{11}), or on both sides of the dependency (see, e.g., Differential Dependencies \cite{16}).

In this paper, we study these generalizations of FDs. Our first objective is to construct a hierarchy of dependencies, revealing which ones (strictly) generalize others, and comparing their axiomatization and complexity of inference.

We then introduce a particular generalization of FDs that we call Antecedent Metric FDs (AMFDs). An AMFD asserts that if two tuples have \textit{similar} but not necessarily equal values of the antecedent attributes, then their consequent values must be equal; we will formalize this definition and compare it with related dependencies in Section 2.
Table 1: Movie Relation

<table>
<thead>
<tr>
<th>source</th>
<th>title</th>
<th>length</th>
<th>year</th>
<th>director</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A Beautiful Mind</td>
<td>135</td>
<td>2001</td>
<td>Ridley Scott</td>
</tr>
<tr>
<td>B</td>
<td>A Beaut. Mind</td>
<td>135</td>
<td>2001</td>
<td>Ridley Scott</td>
</tr>
<tr>
<td>C</td>
<td>Beautiful Mind</td>
<td>135</td>
<td>2001</td>
<td>Ridley Scott</td>
</tr>
</tbody>
</table>

To illustrate the utility of AMFDs, consider the movie data set shown in Table 1, which was put together from multiple data sources. In the process of merging data from various sources, it is often the case that small variations occur. For example, one source might report the movie *A Beautiful Mind* to have a running time of 135 minutes, as shown in Table 1, while another source may refer to the same movie as *A Beaut. Mind* and the third one as *Beautiful Mind*. An AMFD \( \{ \text{title, year, director} \} \rightarrow \{ \text{length} \} \) indicates that movies with similar titles, years, and directors (up to some distance threshold, as we will discuss in Section 2) must have equal lengths. Of course, we assume that the semantics are such that two similar movie titles, made in similar years, by similar director names do in fact refer to the same movie.

Note that a standard FD \( \{ \text{title, year, director} \} \rightarrow \{ \text{length} \} \) would not require the three length values in Table 1 to be equal, even though they refer to the same movie. Thus, AMFDs generalize (subsume) FDs and can express the additional semantics of similarity.

The inference problem is to determine whether a new dependency is logically entailed by a set of given dependencies. For FDs, axiomatization and inference have been studied [3, 4]. In this paper, we prove that while AMFDs are more expressive than FDs and have a more complex axiomatization, their complexity of inference remains linear.

The contributions of this paper are as follows.

1. **Hierarchy:** we construct a hierarchy of dependencies, showing which ones (strictly) generalize others and comparing their complexity of reasoning. Our hierarchy shows which dependencies are practical and which are hard to reason about, and suggests further research on identifying tractable extensions of FDs.
2. **New FD extension:** we introduce AMFDs, which describe integrity constraints on tuples with similar attribute values and are useful in data cleaning.
3. **Axiomatization:** we present a sound and complete axiomatization for AMFDs. Axiomatization is a first necessary step to designing an efficient inference procedure. Our axiomatization reveals interesting insights about inference rules over AMFDs. For instance, the Reflexivity and Augmentation axioms, which hold for traditional FDs, are not necessary true for AMFDs.
4. **Inference Procedure:** we develop an inference procedure for AMFDs that runs in linear time in the complexity of the schema. We implemented the inference algorithm and experimentally verified its efficiency.

The remainder of this paper is organized as follows. In Section 2, we review previous work, we formally define AMFDs, and we present a hierarchy of dependencies. In Sections 3 and 4, we present a sound and complete axiomatization and an inference procedure for AMFDs, respectively, and we compare the axiomatization to those of other related dependencies. We conclude the paper in Section 5.

## 2 Fundamentals

### 2.1 AMFDs

We provide notational conventions in Table 2. To accommodate small variations in the attribute values on the left-hand-side of the dependency, we define AMFDs (Definition 4)}

---

4 Our inference procedure is efficient because it is done at the schema level, which is much smaller than the size of the data.
Table 2: Notational Conventions

Relations
- Capital letters in italic near the beginning of the alphabet represent single attributes: for example, $A$, $B$ and $C$.
- A bold capital letter represents a relation schema: $R$.
- A small bold capital letter in italic represents a relation (a table): $t$.
- Small italic letters near the end of the alphabet denote tuples: $s$ and $t$.
- Small italic letters near the beginning of the alphabet denote attribute values: for example, $a$, $b$ and $c$.
- A small italic letter $m$ denotes a similarity metric.

Sets
- Italic capital letters near the end of alphabet stand for sets of attributes: $X$ & $Y$.
- $XY$ is shorthand for $X \cup Y$. Likewise, $AX$ or $XA$ stand for $X \cup \{ A \}$. $\{ \}$ denotes an empty set.

2). This is a departure from traditional FDs which enforce equality on both sides. Before we define AMFDs, we first define a similarity operator with a distance threshold.

**Definition 1.** (similarity) For every attribute $A$ in a relational schema $R$, we assume a binary similarity relation $(=_{m, \theta})$ w.r.t. some similarity metric $m$ and a threshold parameter $\theta \geq 0$. Specifically, for two tuples $s$ and $t$, $s[A] \approx_{m, \theta} t[A]$ iff $m(s[A], t[A]) \leq \theta$. Metric $m$ satisfies standard properties; it is symmetric, satisfies the triangle inequality and identity of indiscernibles, i.e., $m(a, b) = 0$ iff $a = b$. For two tuples $s$, $t$ in relation $R$, we write $s[X] \approx_{m, \Theta} t[X]$ to mean $s[A_1] \approx_{m_1, \theta_1} t[A_1], \ldots, s[A_n] \approx_{m_n, \theta_n} t[A_n]$, where $X = \{ A_1, \ldots, A_n \}$, $m = [m_1, \ldots, m_n]$ and $\Theta = [\theta_1, \ldots, \theta_n]$.

Next, we define antecedent metric FDs. By definition, AMFDs generalize FDs.

**Definition 2.** (AMFD) Let $X$ and $Y$ be two sets of attributes, and let $m_X$ and $\Theta_X$ be metrics and thresholds for attributes $X$. Then, $X \rightarrow Y$ denotes an antecedent metric FD (AMFD), read as $X$ metrically functionally determines $Y$. Let $R$ be a relation schema that contains the attributes that appear in $X$ and $Y$, and let $t$ be a relation instance of $R$. Relation $t$ satisfies $X \rightarrow Y$ ($t \models X \rightarrow Y$), iff for all tuples $s$, $t \in t$, $s[X] \approx_{m_X, \Theta_X} t[X]$ implies $s[Y] = t[Y]$. An AMFD $X \rightarrow Y$ is said to hold for $R$, written as $R \models X \rightarrow Y$, iff for each admissible relational instance $t$ of $R$, relation $t$ satisfies $X \rightarrow Y$. An AMFD $X \rightarrow Y$ is trivial iff for all $t$, $t \models X \rightarrow Y$.

**Example 1.** (AMFD) Assume metrics $m_{title}$ and $m_{director}$ are edit distances with thresholds $\theta_{title} = 6$ and $\theta_{director} = 0$, respectively, in Table 1 (movie relation). Also assume that metric $m_{year}$ is an integer distance with a threshold $\theta_{year} = 0$. Therefore, Table 1 satisfies the AMFD $\{ title, year, director \} \rightarrow \{ length \}$.

2.2 Related Work

AMFDs and traditional FDs are specified over a single relation. However, AMFDs replace strict equality on the left-hand-side of the dependency with similarity. Dependencies defined over a single relation with similarity on the right-hand-side, called Metric FDs, were proposed by Koudas et al. [13]. We call them Consequent Metric FDs (CMFDs) to distinguish them from AMFDs. The verification problem over CMFDs was studied in [13], which is to decide whether the instance satisfies a prescribed set of dependencies. However, axiomatization and inference were not considered.5

5 Some of the authors of this paper solved the axiomatization and inference problems for CMFDs in a paper currently under submission.
Bertossi et al. [5], Fan [7] and Fan et al. [9] studied Matching Dependencies (MDs), which are object-identification constraints across multiple relations. MDs also enforce similarity rather than equality on the left-hand-side, but allow arbitrary Boolean similarity functions. (These similarity functions only need to satisfy reflexivity, symmetry and subsumption of equality.) On the other hand, AMFDs are defined over a single relation and only allow a restricted notion of similarity, namely thresholds over similarity metrics (recall Definition 1). Fan et al. presented a sound and complete axiomatization\(^6\) and a quadratic-time inference procedure for MDs.

Pointwise Order Dependencies (PODs), which consider order relationship rather than equality of attribute values, were introduced in Ginsburg et al. [10]. A POD \( X \rightarrow Y \) holds if the order over the values of each attribute of set \( X \) implies order over the values of each attribute of set \( Y \). Formally, a relation satisfies a POD \( X \rightarrow Y \) if, for all tuples \( s, t \), for every attribute \( A \) in \( X \), \( s_A \) op \( t_A \) implies that for every attribute \( B \) in \( Y \), \( s_B \) op \( t_B \), where \( \epsilon \in \{<,>,\le,\ge,=\} \). For example, in relation TimePolls (Table 3), the POD \(#date\rightarrow\{year=,month=,day\}\) holds; however, the POD \(#date\rightarrow\{year=,month=,day\}\) does not hold. Ginsburg et al. present a sound and complete axiomatization for PODs and show that the inference problem for them is co-NP-complete.

PODs are defined over sets of attributes. On the other hand, Lexicographical Order Dependencies (LODs) are defined over lists of attributes [17, 19]. LODs describe the relationship among lexicographical orderings of sets of tuples. This is the notion of order used in SQL and in query optimization, as per the order by operator (nested sort). A relation satisfies a LOD \( X \rightarrow Y \) if any list of its tuples that satisfy order by \( X \) also satisfies order by \( Y \); however, not necessarily vice versa. (\( X \) and \( Y \) denote lists of attributes.) For instance, in relation TimePolls, the LODs \( \text{timestamp} \rightarrow \text{date} \) and \( \{\text{date}\} \rightarrow \{\text{year}, \text{month}, \text{day}\} \) are true. The default direction of the SQL order by is ascending.

This can be generalized to order-by’s that mix asc and desc directions, e.g., order by name asc, age desc. For example, in relation TimePolls, the LOD \([\text{sequential_id} \text{desc}] \rightarrow [\text{timestamp asc}]\) holds. Szlichta et al. present a sound and complete axiomatization for lexicographical order dependencies and show that the inference problem for LODs is co-NP-complete [17, 19].

Another constraint for ordered data, sequential dependencies (SDs), was introduced in Golab et al. [11]. For example, the SD \( \text{sequential_id} \rightarrow^{[4,5]} \text{timestamp} \) means that after sorting the data by the attribute \( \text{sequential_id} \), the gaps between consecutive timestamps are between 4 and 5. This particular SD holds in the TimePolls relation; however, the SD \( \text{sequential_id} \rightarrow^{[6,7]} \text{timestamp} \) does not hold. Golab et al. present a framework for discovering which subsets of the data obey a given SD, but axiomatization and inference were not considered.

SDs were generalized in Song et al. [16] by introducing gaps (referred to as differential functions) on both sides of the dependency and named Differential Dependencies (DDs). For instance, in the relation TimePolls, the DDs \( \text{sequential_id}^{[1,1]} \rightarrow \text{timestamp}^{[4,5]} \) and \( \{\text{date}^{[0,1]}\} \rightarrow \{\text{year}^{[0,0]}, \text{month}^{[0,0]}, \text{day}^{[0,1]}\} \) hold. However, the DD \( \text{sequential_id}^{[1,1]} \rightarrow \text{timestamp}^{[5,6]} \) does not hold. Song et al. present an axiomatization and show that inference problem for DDs is co-NP-complete.

---

**Table 3: TimePolls Relation**

<table>
<thead>
<tr>
<th>sequential_id</th>
<th>timestamp</th>
<th>date</th>
<th>year</th>
<th>month</th>
<th>day</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2014-02-01 14:23:20</td>
<td>2014-02-01</td>
<td>2014</td>
<td>02</td>
<td>01</td>
</tr>
<tr>
<td>2</td>
<td>2014-02-01 14:23:25</td>
<td>2014-02-02</td>
<td>2014</td>
<td>02</td>
<td>02</td>
</tr>
</tbody>
</table>

\(^6\) It is stated in Fan et al. [9] (without a proof of completeness) that a complete axiomatization for MDs consists of eleven axioms, but only nine sound axioms are presented.
2.3 Hierarchy of Dependencies

Figure 1 illustrates a hierarchy of the dependencies we discussed above as well as a new class: MDDs. MDDs strictly generalize MDs and DDs by allowing differential functions (on the left hand side and the right hand side) with arbitrary similarity functions and allowing multiple tables. Below each dependency name, we point out the complexity of inference. Observe that for the “not studied” dependencies, their complexity of inference is bookended by their immediate ancestors and descendants in the hierarchy.

We say that a dependency class $A$ generalizes dependency class $B$ iff there is a semantically preserving mapping of any dependency of class $B$ into a set of dependencies of class $A$. Class $A$ strictly generalizes class $B$ iff $A$ generalizes $B$, however, $B$ does not generalize $A$. The hierarchy in Figure 1 shows which dependencies strictly generalize others; due to space constraints, proofs will appear in extended version of this paper.

Fig. 1: Hierarchy of dependencies and their complexity of inference.

For example, Differential Dependencies strictly generalize Sequential Dependencies. In our example involving Table 3, with the SD $\text{sequential id} \rightarrow [4, 5] \text{time}$, consecutive sequence numbers can be simulated by using on the left-hand-side of the DD a similarity metric which returns distance one if two numbers are consecutive and zero otherwise. Axiomatization and complexity of inference for SDs are open problems. However, since our hierarchy indicates that DDs strictly generalize SDs, the upper bound on the complexity of inference for SDs is co-NP complete.

Similarly, DDs strictly generalize PODs (which strictly generalize LODs [19]). For instance, a POD $A^\geq \rightarrow B^\leq$ is equivalent to a DD $A^{[0;+\infty]} \rightarrow B^{[-\infty;0]}$. Interestingly, this suggests that the complexity results for PODs can be adapted to DDs and did not have to be re-developed from scratch in [16]. AMFDs and CMFDs are also subsumed by DDs, since DDs allow similarity both on the left-hand-side and right-hand side. (Note that limited AMDs are introduced in Section 3.) Both AMFDs and CMFDs strictly generalize traditional FDs by replacing equality with similarity.

3 Axiomatization

3.1 Soundness and Completeness

We now present an axiomatization for AMFDs, analogous to Armstrong’s axiomatization for FDs [3, 4]. This provides a formal framework for reasoning about AMFDs. The axioms give insights into how AMFDs behave and reveal how dependencies logically follow from others, which is not easily evident when reasoning from first principles. Also,
1. Void
   \( X \mapsto \{ \} \)

2. Transitivity
   \( \text{If } X \mapsto Y \text{ and } Y \mapsto Z \text{ then } X \mapsto Z \)

3. Composition
   \( \text{If } X \mapsto Y \text{ and } Z \mapsto W \text{ then } XZ \mapsto YW \)

4. Decomposition
   \( \text{If } X \mapsto Y \text{ and } Z \subseteq Y \text{ then } X \mapsto Z \)

5. Reduce
   \( \text{If } XZ \mapsto Y \text{ and } X \mapsto Z \text{ then } X \mapsto Y \)

6. Limited Reflexivity
   \( \text{If } Y \subseteq X \text{ and } \theta_Y = 0 \text{ then } X \mapsto Y \)

Fig. 2: Axiomatization for AMFDs.

a sound and complete axiomatization is necessary for an efficient inference procedure (see Section 4).

The inference rules (axioms) for AMFDs are presented in Figure 2. Recall that \( \{ \} \) denotes an empty set. Two of the axioms generate trivial dependencies that are always true: Void and Limited Reflexivity. Below we introduce additional inference rules that follow from the axioms in Figure 2. These will be used throughout the paper, particularly to prove that our AMFD axioms are complete.

**Lemma 1.** (Left Augmentation) If \( X \mapsto Y \), then \( XZ \mapsto Y \).

*Proof.* We are given \( X \mapsto Y \). By Void and Composition it follows that \( XZ \mapsto Y \). \( \Box \)

**Lemma 2.** (Union) If \( X \mapsto Y \) and \( X \mapsto Z \), then \( X \mapsto YZ \).

*Proof.* We are given \( X \mapsto Y \) and \( X \mapsto Z \). By Composition it follows that \( X \mapsto YZ \). \( \Box \)

Next, we define closure over AMFDs. The closure of a set of attributes \( X \) is the set of attributes that \( X \) logically determines given a set of AMFDs \( F \).

**Definition 3.** (Closure \( X^+ \)) The AMFD-closure of set of attributes \( X \), denoted \( X^+ \), w.r.t. a set of AMFDs \( F \) using axioms \( I = \{1-6\} \) in Figure 2, is defined as, \( X^+ = \{A \mid F \vdash X \mapsto A\} \).

Lemma 3 tells us whether a dependency \( X \mapsto Y \) follows from \( F \) using our axioms.

**Lemma 3.** (Closure for AMFDs) \( F \vdash X \mapsto Y \), if and only if \( Y \subseteq X^+ \).

*Proof.* Let \( Y = \{A_1, ..., A_n\} \). Assume \( Y \subseteq X^+ \). By definition of \( X^+ \), \( X \mapsto A_i \) for all \( i \in \{1, ..., n\} \). Therefore, by the Union axiom, \( X \mapsto Y \). To prove the other direction, suppose \( X \mapsto Y \) follows from the axioms. For each \( i \in \{1, ..., n\} \), \( X \mapsto A_i \) by Decomposition, so \( Y \subseteq X^+ \). \( \Box \)

**Theorem 1.** (Soundness and Completeness) AMFD axioms are sound and complete.

*Proof.* The soundness proof (if \( F \vdash X \mapsto Y \), then \( F \models X \mapsto Y \)) is trivial. We just have to show that each axiom presented Figure 2 is true. Below we present the completeness proof (if \( F \models X \mapsto Y \), then \( F \vdash X \mapsto Y \)). We consider a table \( t \) with two rows, whose template is shown in Table 4. We divide the attributes of \( t \) into three subsets: the closure of \( X \), the set \( N \), consisting of attributes in \( X \) that are not in the closure of \( X^+ \), and all the remaining attributes. All the attributes of the first row have the value \( a \), while for the second row, the attributes in \( X^+ \) are \( a \)'s, the attributes in \( N \) are \( b \)'s and the other attributes are \( c \)'s. Without loss of generality, assume that for all the attributes \( A \) in \( N \) and other attributes over table \( t \) we use the same metric \( m \), and that \( a \) and \( b \) are similar \( (a \approx_{m, \theta_A} b) \) but not equal. Also, assume that the values \( a \) and \( c \) are not similar \( (a \not\approx_{m, \theta_A} c) \), and hence not equal.

We first show that all dependencies in the set of AMFDs \( F \) are satisfied by table \( t \) (\( t \models F \)). Since the AMFD axioms are sound, AMFDs inferred from \( F \) are true. Note that by Void and Limited Reflexivity, all trivial AMFDs are satisfied in table \( t \).

\text{For a traditional FD } X \mapsto Y, \text{ by Reflexivity all the attributes in } X \text{ are also in } X^+. \text{ However, this is not true for AMFDs, as we will show in Example 2.}

\( \Box \)
Assume \( V \rightarrow Z \) is in \( F \) but is not satisfied by table \( t \). Therefore, \( V \subseteq \{ X^+ \cup N \} \) because otherwise two rows of \( t \) are not similar on some attribute of \( V \) since \( a \not\approx_{m,\theta_A} c \), and consequently an AMFD \( V \rightarrow Z \) would not be violated. Moreover, \( Z \) cannot be a subset of \( X^+ \) (\( Z \not\subseteq X^+ \)), or else \( V \rightarrow Z \) would be satisfied by \( t \). Let \( A \) be an attribute of \( Z \) not in \( X^+ \). Since the dependency \( V \rightarrow Z \) is in \( F \), by Decomposition, \( V \rightarrow A \).

Let \( V_1 \) be a maximal set of attributes such that \( V_1 \subseteq V \) and \( V_1 \subseteq X^+ \). Let \( V_2 \) be a maximal set of attributes such that \( V_2 \subseteq V \) and \( V_2 \subseteq N \). By Union and Definition 3 of closure, \( X \rightarrow X^+ \). Therefore, by Left Augmentation and Reduce, \( XV_2 \rightarrow A \). Since \( N = \{ A \mid A \in X \land A \not\in X^+ \} \), \( V_2 \subseteq X \). Hence, \( X \rightarrow A \), which is a contradiction.

Our remaining proof obligation is to show that any AMFD not inferable from the set of AMFDs \( F \) with our axioms (\( F \not\models X \rightarrow Y \)) is not true (\( F \models X \rightarrow Y \)). Suppose it is satisfied (\( F \models X \rightarrow Y \)). It follows by the construction of table \( t \) that \( Y \subseteq X^+ \); otherwise, two rows of table \( t \) agree or are similar on \( X \) but disagree on some attribute \( A \) from \( Y \). Since \( Y \subseteq X^+ \), by Lemma 3 it can be inferred that \( X \rightarrow Y \), which is a contradiction. Thus, whenever \( X \rightarrow Y \) does not follow from \( F \) by the AMFD axioms, \( F \) does not logically imply \( X \rightarrow Y \). That is, the axiom system is complete over AMFDs, which ends the proof of Theorem 1. \( \Box \)

### 3.2 Discussion

The axiomatization for AMFDs is more involved than its FDs counterpart. A sound and complete axiomatization for traditional FDs consists of only three axioms: Reflexivity, Augmentation and Transitivity. Interestingly, Reflexivity (if \( Y \subseteq X \), then \( X \rightarrow Y \)) is not necessary true for AMFDs.

**Example 2.** (lack of Reflexivity) Consider table \( t \) (Table 4). Assume again that the values \( a \) and \( b \) are not equal \( (a \neq b) \) but they are similar \( (a \approx_{m,\theta_A} b) \) for each attribute \( A \) in \( N \). Let attributes \( \{BCD\} \subseteq N \). Therefore, the AMFDs \( BCD \rightarrow BCD \) and \( BCD \rightarrow BC \) are not satisfied in \( t \) because the values are similar on the left hand side of the dependencies, but not equal on their right hand side.

Similarly, the Augmentation inference rule, which is another axiom for FDs, does not necessary hold for AMFDs. (Augmentation states that if \( X \rightarrow Y \) then \( XZ \rightarrow YZ \).)

**Example 3.** (lack of Augmentation) Consider table \( t \) (Table 4). Assume again that the values \( a \) and \( b \) are not equal \( (a \neq b) \) but they are similar \( (a \approx_{m,\theta_A} b) \) for each attribute \( A \) in \( N \). Let an attribute \( B \in N \), and attributes \( \{CD\} \subseteq X^+ \). The AMFD \( C \rightarrow D \) holds in table \( t \). However, the AMFD \( CB \rightarrow DB \) is not true.

We replaced Reflexivity with Void and Limited Reflexivity in the axiomatization for AMFDs. Lack of Augmentation forced us to add Composition and Decomposition to the axiomatization. Note that Left Augmentation (Theorem 1) holds for AMFDs. Since Reflexivity does not hold for AMFDs, we had to add Reduce. Only Transitivity, which is a base axiom for FDs, was preserved in our axiomatization for AMFDs.

We also studied an axiomatization for a simplified version of MDs over a single table, rather than multiple tables as originally defined in [5,7,9]. We call these limited AMDs. The main difference between limited AMDs and AMFDs is that the former allow arbitrary similarity functions while the latter employ thresholds on similarity metrics.

A sound and complete axiomatization for limited AMDs consists of the following five axioms: Void, Transitivity, Composition, Decomposition and Reduce. The proof will

<table>
<thead>
<tr>
<th>( X^+ )</th>
<th>( N = { A \mid A \in X \land A \not\in X^+ } )</th>
<th>other attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a \ldots a )</td>
<td>( a \ldots a )</td>
<td>( a \ldots a )</td>
</tr>
<tr>
<td>( b \ldots b )</td>
<td>( c \ldots c )</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Table template for AMFDs.
Table 5: Comparison of Axiomatizations

<table>
<thead>
<tr>
<th>Dependency Class</th>
<th>Axioms</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDs [16]</td>
<td>Extended Reflexivity, Extended Augmentation, Extended Transitivity, Impropriety</td>
</tr>
<tr>
<td>SDs [11]</td>
<td>N/A</td>
</tr>
<tr>
<td>PODs [10]</td>
<td>Reflexivity, Augmentation, Transitivity, Reversal, Disjunction, Total Order, Impropriety</td>
</tr>
<tr>
<td>LODs [17, 19]</td>
<td>Reflexivity, Transitivity, Augmentation, Suffix, Normalization, Chain</td>
</tr>
<tr>
<td>MDs [7, 5, 9]</td>
<td>9 sound axioms (out of 11) appear in [9]</td>
</tr>
<tr>
<td>limited AMDs [this paper]</td>
<td>Void, Transitivity, Composition, Decomposition, Reduce</td>
</tr>
<tr>
<td>AMFDs [this paper]</td>
<td>Void, Transitivity, Composition, Decomposition, Reduce, Limited Reflexivity</td>
</tr>
<tr>
<td>CMFDs [13]</td>
<td>Footnote 5</td>
</tr>
<tr>
<td>FDs [3, 4, 12]</td>
<td>Reflexivity, Augmentation, Transitivity</td>
</tr>
</tbody>
</table>

appear in the extended version of this paper; it is a simplified version of the proof of Theorem 1. In comparison to AMFDs, Limited Reflexivity does not hold for limited AMDs. A sound and complete axiomatization for a full class of MDs is more complex (Footnote 6), as it incorporates axioms that allow us to reason over multiple relations.

Table 5 compares the axiomatization of AMFDs and AMDs with other dependency classes. We point out several interesting observations below.

In contrast to MDs and AMFDs, the distance (gap) functions for DDs are defined at the dependency level for each attribute instead of the schema level. Therefore, the Transitivity axiom for DDs additionally requires an order relation over differential functions. For instance, if we have the dependency “if the date difference for two tuples is \( \leq 30 \) days, then price \( \geq 50 \)”, then the dependency “if the date difference for two tuples is \( \leq 30 \) days, then price \( \geq 40 \)” must also hold.

There is an extra inference rule for DDs–Impropriety–that accommodates inconsistencies between dependencies (the inconsistency problem does not arise in AMFDs and MDs). For example, the following two dependencies are inconsistent since it is not possible to instantiate a relation that satisfies both of them: a) if the date difference for two tuples is \( \leq 30 \) days, then price = 50; and b) if the date difference for two tuples is \( \leq 30 \) days, then price > 50. Similarly, Augmentation and Reflexivity have to be modified for DDs to accommodate different distance functions used by different dependencies on the same attribute. For instance, different distance functions for the same attribute may result in the same actual distance.

Interestingly, as we traverse the hierarchy of dependencies, the number of axioms does not necessarily decrease. There are: six axioms for AMFDs, seven axioms for PODs and six axioms for LODs versus four axioms for DDs; however, there are three axioms for FDs at the bottom of hierarchy. There are two reasons for this. First, the axioms for DDs are quite complex. Second, as we go down the hierarchy, the dependencies become more specialized and therefore we may need more axioms to express their restricted semantics, e.g., lack of Reflexivity. As the dependencies become more generalized, some axioms must be weakened, e.g., Limited Reflexivity.

4 Inference Procedure

A goal of a dependency theory is to develop algorithms for the inference problem. Inference for DDs is co-NP-complete [16] and for MDs it is quadratic [7], as shown in Figure 1. Since DDs and MDs generalize AMFDs, this sets an upper bound for the complexity of inference for AMFDs. However, computing closure, \( X^+ \), for AMFDs can be done more efficiently. It takes time proportional to the length of the dependencies in \( F \), written out (linear time), which is as efficient as for FDs. (The complexity of inference for limited AMDs is also linear; the proof will appear in the extended version of
Algorithm 1 Inference procedure for AMFDs

**Input:** A set of AMFDs $F$, and a set of attributes $X$.

**Output:** The closure of $X$ with respect to $F$.

1: $F_{\text{unused}} \leftarrow F$
2: $X^0 \leftarrow W$ where $W = \{A \mid A \in X \text{ and } \theta_A = 0\}$
3: $n \leftarrow 0$
4: loop
5: if $\exists V \rightarrow Z \in F_{\text{unused}}$ and $V \subseteq \{X^n \cup X\}$ then
6: $X^{n+1} \leftarrow X^n \cup Z$
7: $F_{\text{unused}} \leftarrow F_{\text{unused}} - \{V \rightarrow Z\}$
8: $n \leftarrow n + 1$
9: else
10: return $X^n$
11: end if
12: end loop

this paper.) Algorithm 1 presents an inference procedure for AMFDs. Our experiments have shown that it is very efficient. For 10 AMFDs prescribed over a synthetic dataset generated by the UIS Database [2], the algorithm runs in time $\leq 1\text{ms}$.

Example 4. (inference) Let $F = \{AB \rightarrow C, ABC \rightarrow EG, EG \rightarrow H\}$ denote the set of AMFDs. Also, let $\theta_C = 0$ and $\theta_D > 0$ for all attributes $D$ in $ABEGH$. Let us calculate the closure of set of attributes $AB$ with Algorithm 1:

1) $X^0 = \{\}$;
2) $X^1 = C$;
3) $X^2 = CE$;
4) $X^3 = CEGH$.

The closure of $AB$ is $CEGH$. For traditional FDs, the closure of $AB$ is $ABCEGH$.

Theorem 2. (Correctness) Algorithm 1 correctly computes the closure $X^+$ over AMFDs.

Proof. First we show by induction on $k$ that if $Z$ is placed in $X^k$ in Algorithm 1, then $Z$ is in $X^+$.

**Base case:** $k = 0$. By Limited Reflexivity, $X \rightarrow W$, where $W = \{A \mid A \in X \text{ and } \theta_A = 0\}$.

**Induction step:** $k > 0$. Assume that $X^{k-1}$ only consists of the attributes in $X^+$. Suppose $Z$ is placed in $X^k$ because $VW \rightarrow Z \in F_{\text{unused}}$, such that $V \subseteq X^{k-1}$ and $W \subseteq X$. Since $V \subseteq X^{k-1}$, we know by the induction hypothesis that $V \subseteq X^+$. Hence, by Lemma 3, $X \rightarrow V$. Therefore, since $XV \rightarrow VZ$ by Composition, then by Reduce and Decomposition $X \rightarrow Z$. Thus, $Z$ is in $X^+$.

Now we prove the opposite: if $Z$ is in $X^+$, then $Z$ is in the set returned by Algorithm 1. Suppose $Z$ is in $X^+$ but $Z$ is not in the set returned by Algorithm 1. Consider table $t$ similar to that in Table 4. Table $t$ has two tuples that agree on attributes in $X^n$, are similar but not equal on attributes $X$ that are not subset of $X^n$, and disagree on all other attributes. We claim that $t$ satisfies $F$. If not, let $P \rightarrow Q$ be a dependency in $F$ that is violated by $t$. Then $P \subseteq X^n \cup X$ and $Q$ cannot be a subset of $X^n \cup X$, if the violation happens. We used a similar argument in the proof of Theorem 1. Thus, by Algorithm 1, Lines 5–8, there exists $X^{n+1}$, which is a contradiction. $\square$

5 Conclusions and Future Work

In this paper, we developed a hierarchy of dependency classes and laid out the theoretical foundations for AMFDs, which generalize traditional FDs. In future work, we plan to investigate the following problems.

- Determining whether a given AMFD holds on a given relation, and using AMFDs for data cleaning, similarly to how FDs were employed in previous data cleaning work[6, 7].
FDs have recently been extended to Conditional FDs (CFDs) [8], and we plan to study conditional extensions of AMFDs. A conditional AMFD can be represented as a pair \((X \rightarrow \rightarrow Y, T_r)\), where \(X \rightarrow Y\) is the embedded AMFD, and \(T_r\) is a pattern tableau defining over which rows the dependency applies.

Algorithms for automatic discovery of dependencies have been proposed for some dependencies, such as FDs and CFDs [8]. Similarly, we plan to study algorithms for discovering AMFDs.

We plan to explore an axiomatization and inference framework for multiple dependencies. For example, the following inference rules hold:

\begin{itemize}
  \item[a)] if an AMFD \(X \rightarrow \rightarrow Y\), then a CMFD \(X \rightarrow \rightarrow Y\),
  \item[b)] if an AMFD \(X \rightarrow \rightarrow Y\), then a FD \(X \rightarrow Y\),
  \item[c)] if an FD \(X \rightarrow Y\), then a CMFD \(X \rightarrow \rightarrow Y\).
\end{itemize}

The above rules, along with the axioms for AMFDs (Figure 2) and CMFDs (Footnote 5), are sound for the integrated inference problem with FDs, AMFDs and CMFDs. However, an open question is if this rule set is complete and what is the complexity of the integrated inference problem.

Integrity constraints have been widely used in query optimization. For instance, FDs and LODs have been shown to be useful in simplifying queries with group by and order by [14,18,17,19] We believe that AMFDs can be used in similar ways to simplify SQL queries with similarity operators [15].

References