# Reasoning with Examples: Propositional Formulae and Database Dependencies

# Reasoning with Examples: Propositional Formulae and Database Dependencies

Roni Khardon
Heikki Mannila
and
Dan Roth

TR-15-95

Center for Research in Computing Technology
Harvard University
Cambridge, Massachusetts

# Reasoning with Examples:
# Propositional Formulae and Database Dependencies

**Roni Khardon**[*]
Harvard University
Cambridge, MA 02138

**Heikki Mannila**[†]
University of Helsinki

**Dan Roth**[‡]
Harvard University
Cambridge, MA 02138

June 13, 1995

## Abstract

For humans, looking at how concrete examples behave is an intuitive way of deriving conclusions. The drawback with this method is that it does not necessarily give the correct results. However, under certain conditions example-based deduction can be used to obtain a correct and complete inference procedure. This is the case for Boolean formulae (reasoning with models) and for certain types of database integrity constraints (the use of Armstrong relations). We show that these approaches are closely related, and use the relationship to prove new results about the existence and sizes of Armstrong relations for Boolean dependencies. Further, we study the problem of translating between different representations of relational databases, in particular we consider Armstrong relations and Boolean dependencies, and prove some positive results in that context. Finally, we discuss the close relations between the questions of finding keys in relational databases and that of finding abductive explanations.

[*] e-mail: roni@das.harvard.edu. Research supported by Center for Intelligent Control Systems under ARO contract DAAL03-92-G-0115.

[†] e-mail: mannila@cs.helsinki.fi.

[‡] e-mail: danr@das.harvard.edu. Research supported by NSF grant CCR-92-00884 and by DARPA AFOSR-F4962-92-J-0466.

# 1 Introduction

One of the major tasks in database systems as well as artificial intelligence systems is to express some knowledge about the domain in question and then use this knowledge to determine the validity of certain queries on the domain. This normally involves settling on a semantic implication relation with respect to the knowledge representation.

Traditionally, a language $\mathcal{L}$ for expressing statements about objects in a class $\mathcal{O}$ is devised, and $\Sigma \subseteq \mathcal{L}$, a set of sentences, denotes the knowledge about the domain. The statement $\phi \in \mathcal{L}$, a single sentence, represents a query in question. We are interested in knowing whether $\Sigma$ logically implies $\phi$, that is, whether we have that for each object $m \in \mathcal{O}$ such that all sentences of $\Sigma$ are true in $m$, we also have that $\phi$ is true in $m$.

It is normally argued that checking this semantic implication relation by using the above definition explicitly is impossible due to the large number of possible objects, and hence one needs to find efficiently implementable sound and complete axiomatizations for the problem.

Some theories on human reasoning [JL83], however, claim that humans typically argue by just looking at some examples: one selects some objects $m_i \in \mathcal{O}$ such that $\Sigma$ is true, and checks whether $\phi$ is also true for these objects. If not, then one can make the correct conclusion that $\Sigma$ does not imply $\phi$; if $\phi$ is true for all the examples $m_i$, one concludes that $\Sigma$ implies $\phi$. Of course, this conclusion can be wrong.

There are, nevertheless, some domains where inference of this type can yield correct answers. A dramatic example is give by Hong [Hon86], who proves that for certain types of geometric statements one example (consisting of real numbers) is sufficient to prove or disprove any geometric theorem about points, lines and circles in the plane. Moreover, he shows that one does not have to consider more than a polynomial number of digits in the example (with respect to the number of objects mentioned in the theorem).

In this paper we consider two areas where such *reasoning with examples* has been used, and show that the methods are actually quite closely related.

The first area is in database integrity constraints, where so called *Armstrong relations* serve as a single example capturing all implications of a set of sentences [Fag82b, Fag82a, BDFS84]. More formally, if $\mathcal{L}$ is a set of database dependencies and $\Sigma \subseteq \mathcal{L}$, then an Armstrong relation for $\Sigma$ and $\mathcal{L}$ is a database relation $r_\Sigma$ such that for all $\phi \in \mathcal{L}$ we have that $\Sigma$ implies $\phi$ if and only if $r_\Sigma$ satisfies $\phi$. That is, the semantic implication relation for $\Sigma$ reduces to the truth in a single example relation $r_\Sigma$.

The second area is in automated reasoning, where *reasoning with models* has been recently studied [KKS93, KR94b]. Assume the language consists of propositional formulae, and the objects are models (i.e., truth assignments). A set of models $M \subseteq \mathcal{O}$ is a sufficient set of examples for $\Sigma$ and $\mathcal{L}$, if for all $\phi \in \mathcal{L}$ we have: if for all $m \in M$ $\phi$ is true in $m$, then $\Sigma$ implies $\phi$. That is, instead of having to look at all objects to determine semantic implication, it is sufficient to look only at a sub-collection of the objects. In particular, the set of characteristic models has this property.

In these applications it is of course important that the example relation or the set of examples is small; otherwise there is no sense in using example-based deduction. The size issue has been treated for database dependencies in [BDFS84, MR86] and for propositional formulae in [KKS93, KR94b].

In this paper we show that these two application areas of the general idea of example-based reasoning are actually very closely related. Namely, we show that the characteristic models of [KKS93] are essentially the intersection generators for sets of functional dependencies in [BDFS84]. Further, the correspondence holds for generalizations of characteristic models introduced in [KR94b].

We then apply this correspondence to get some new results. First, we prove some bounds for the size of Armstrong relations for various types of database dependencies. We present a new family of Bounded Disjunctive Dependencies, which generalizes the class of Functional Dependencies. We show that this family enjoys Armstrong relations, and derive size bounds for its Armstrong relations.

Secondly, we show that the theory of reasoning with models can be of help for interactive design of relational databases. It has been suggested [MR86] that in such scenarios it would be useful if we could translate a set of sentences into the corresponding Armstrong relation and vice versa, translate a given relation into a set of dependencies which it describes. An immediate corollary from the equivalence shown here, and recent results on characteristic models [Kha95], is that the complexity of the two translation problems is equivalent under polynomial reductions. While the complexity of these problem is an open problem [MR86, EG94, FK94, Kha95], we show that results from the theory of reasoning with models [KR94b] and computational learning theory [Bsh93] can be used to derive some positive results. In particular, we show that a closed form (although not in the form of functional dependencies) for a given relation can be found. Furthermore, we show that, given a set of dependencies, an approximate Armstrong relation can be found, where approximate is properly quantified.

Lastly, we show another correspondence between the two domains. In particular, abductive explanations for propositional formulae correspond to keys for relational schemas with Boolean dependencies. We show how several results developed independently in the two domains are in fact equivalent.

The paper is organized as follows: In Section 2 we describe the theory of reasoning with models. In Section 3 we introduce some of the basic notions in relational databases, and discuss the correspondence between them and notions in the Boolean domain. In Sections 4, 5, 6 we show the equivalence between Armstrong relations and characteristic models, and apply this relation to get new results in to the domain of relational databases. In Section 7 we discuss the close relation between the study of abductive explanations in the Boolean domain and keys in relational databases.

## 2 Reasoning with Models

In this section we describe the theory of reasoning with models, and then give applications in the Boolean domain. We start with some notation. We consider a Boolean functions $f : \{0,1\}^n \rightarrow \{0,1\}$. The elements in the set $\{x_1, \ldots, x_n\}$ are called variables. Assignments in $\{0,1\}^n$ are denoted by $x, y, z$, and $weight(x)$ denotes the number of 1 bits in the assignment $x$. A literal is either a variable $x_i$ (called a positive literal) or its negation $\overline{x_i}$ (a negative literal). A clause is a disjunction of literals and a CNF formula is a conjunction of clauses. For example $(x_1 \vee \overline{x_2}) \wedge (x_3 \vee \overline{x_1} \vee x_4)$ is a CNF formula with two clauses. A term is a conjunction of literals and a DNF formula is a disjunction of terms. For example $(x_1 \wedge \overline{x_2}) \vee (x_3 \wedge \overline{x_1} \wedge x_4)$ is a DNF formula with two terms. A CNF formula is Horn if every clause in it has at most one positive literal. We note that every Boolean function has many possible representations and in particular, both a CNF representation and a DNF representation. The size of the CNF and DNF representation are, respectively, the number of clauses and the number of terms in the representation.

An assignment $x \in \{0,1\}^n$ satisfies $f$ if $f(x) = 1$. Such an assignment $x$ is also called a model of $f$. By "$f$ implies $g$", denoted $f \models g$, we mean that every model of $f$ is also a model of $g$. Throughout the paper, when no confusion can arise, we identify a Boolean function $f$ with the set of its models, namely $f^{-1}(1)$. Observe that the connective "implies" ($\models$) used between Boolean functions is equivalent to the connective "subset or equal" ($\subseteq$) used for subsets of $\{0,1\}^n$. That is, $f \models g$ if and only if $f \subseteq g$.

## 2.1 Theory

We start by describing some results of the Monotone Theory of Boolean functions, introduced by Bshouty [Bsh93], and then use those to present the theory of reasoning with models, developed in [KR94b]. All the proofs in this section are omitted; they can be found in [KR94b].

### 2.1.1 Monotone Theory

**Definition 2.1 (Order)** *We denote by $\leq$ the* usual partial order *on the lattice $\{0,1\}^n$, the one induced by the order $0 < 1$. That is, for $x, y \in \{0,1\}^n$, $x \leq y$ if and only if $\forall i, x_i \leq y_i$. For an assignment $b \in \{0,1\}^n$ we define $x \leq_b y$ if and only if $x \oplus b \leq y \oplus b$ (Here $\oplus$ is the bitwise addition modulo 2). We say that $x > y$ if and only if $x \geq y$ and $x \neq y$.*

Intuitively, if $b_i = 0$ then the order relation on the $i$th bit is the normal order; if $b_i = 1$, the order relation is reversed and we have that $1 <_{b_i} 0$. We now define:
The *monotone extension of* $z \in \{0,1\}^n$ with respect to $b$:

$$\mathcal{M}_b(z) = \{x \mid x \geq_b z\}.$$

The *monotone extension of* $f$ with respect to $b$:

$$\mathcal{M}_b(f) = \{x \mid x \geq_b z, \ \textit{for some}\ z \in f\}.$$

The set of *minimal assignments of* $f$ with respect to $b$:

$$\min_b(f) = \{z \mid z \in f, \quad \textit{such that}\ \forall y \in f, z \not>_b y\}.$$

The following claims lists some properties of $\mathcal{M}_b$. All are immediate from the definitions:

**Claim 2.1** *Let $f, g : \{0,1\}^n \rightarrow \{0,1\}$ be Boolean functions. The operator $\mathcal{M}_b$ satisfies the following properties:*
*(1) If $f \subseteq g$ then $\mathcal{M}_b(f) \subseteq \mathcal{M}_b(g)$.*
*(2) $\mathcal{M}_b(f \wedge g) \subseteq \mathcal{M}_b(f) \wedge \mathcal{M}_b(g)$.*
*(3) $\mathcal{M}_b(f \vee g) = \mathcal{M}_b(f) \vee \mathcal{M}_b(g)$.*
*(4) $f \subseteq \mathcal{M}_b(f)$.*

**Claim 2.2** *Let $z \in f$. Then, for every $b \in \{0,1\}^n$, there exists $u \in min_b(f)$ such that $\mathcal{M}_b(z) \subseteq \mathcal{M}_b(u)$.*

Using Claims 2.2 and 2.1 we get a characterization of the monotone extension of $f$:

**Claim 2.3** *The monotone extension of $f$ with respect to $b$ is:*

$$\mathcal{M}_b(f) = \bigvee_{z \in f} \mathcal{M}_b(z) = \bigvee_{z \in min_b(f)} \mathcal{M}_b(z).$$

Clearly, for every assignment $b \in \{0,1\}^n$, $f \subseteq \mathcal{M}_b(f)$. Moreover, if $b \notin f$, then $b \notin \mathcal{M}_b(f)$ (since $b$ is the smallest assignment with respect to the order $\leq_b$). Therefore:

$$f = \bigwedge_{b \in \{0,1\}^n} \mathcal{M}_b(f) = \bigwedge_{b \notin f} \mathcal{M}_b(f).$$

The question is if we can find a small set of negative examples $b$, and use it to represent $f$ as above.

**Definition 2.2 (Basis)** *A set $B$ is a* basis for $f$ *if $f = \bigwedge_{b\in B} \mathcal{M}_b(f)$. $B$ is a basis for a class of functions $\mathcal{F}$ if it is a basis for all the functions in $\mathcal{F}$.*

Using this definition, the representation

$$f = \bigwedge_{b\in B} \mathcal{M}_b(f) = \bigwedge_{b\in B} \bigvee_{z\in \min_b(f)} \mathcal{M}_b(z) \tag{1}$$

yields the following necessary and sufficient condition describing when $x \in \{0,1\}^n$ is positive for $f$:

**Corollary 2.4** *Let $B$ be a basis for $f$, $x \in \{0,1\}^n$. Then, $x \in f$ (i.e., $f(x) = 1$) if and only if for every basis element $b \in B$ there exists $z \in min_b(f)$ such that $x \geq_b z$.*

It is known that the size of the basis for a function $f$ is bounded by the size of its CNF representation, and that for every $b$ the size of $\min_b(f)$ is bounded by the size of its DNF representation. There also exist functions $f$ such that every basis for $f$ is exponential in the size of of the DNF representation of $f$.

### 2.1.2 Deduction

Recall that $f \models \alpha$ if and only if every model of $f$ is also a model of $\alpha$. We are interested in answering deduction queries of the form $f \models \alpha$, given some knowledge about $f$. In particular we study the model based approach for answering such queries. Let $\Gamma \subseteq f \subseteq \{0,1\}^n$ be a set of models. The model-based algorithm, when presented with a query $f \models \alpha$, performs the following: for all the models $z \in \Gamma$ check whether $\alpha(z) = 1$. If for some $z$, $\alpha(z) = 0$ say "no"; otherwise say "yes".

By definition, if $\Gamma = f$ this approach yields correct deduction. Thus for every function $f$ there exists an example-based approach to the deduction of $f$.

However, representing $f$ by explicitly checking *all* the possible models of $f$ is not plausible. A model-based approach becomes feasible if $\Gamma$ supports correct deduction and is small. In the following we characterize a model-based knowledge base that provides for correct reasoning.

**Definition 2.3** *Let $\mathcal{F}$ be a class of functions. For a knowledge base $f \in \mathcal{F}$ we define the set $\Gamma = \Gamma_f^B$ of* characteristic models *to be the set of all minimal assignments of $f$ with respect to the basis $B$. Formally,*

$$\Gamma_f^B = \cup_{b\in B}\{z \in min_b(f)\}.$$

Next we discuss the notion of a least upper bound of a Boolean function [SK91], its relation to the monotone theory and its usage in model-based reasoning.

**Definition 2.4 (Least Upper-bound)** *Let $\mathcal{F}, \mathcal{G}$ be classes of Boolean functions. Given $f \in \mathcal{F}$ we say that $g \in \mathcal{G}$ is a $\mathcal{G}$-least upper bound of $f$ if and only if $f \subseteq g$ and there is no $f' \in \mathcal{G}$ such that $f \subset f' \subset g$.*

**Theorem 2.5** *Let $f$ be any Boolean function and $\mathcal{G}$ a class of all Boolean functions with basis $B$. Then*

$$f_{lub}^B = \bigwedge_{b\in B} \mathcal{M}_b(f)$$

*is a $\mathcal{G}$-upper bound of $f$.*

The above theorem shows that the logical function represented by the set $\Gamma_f^B$, where $B$ is a basis for $\mathcal{G}$, is the LUB of $f$ in $\mathcal{G}$. Nevertheless, this representation is sufficient to support exact deduction with respect to queries in $\mathcal{G}$:

**Theorem 2.6** *Let $B$ be a basis for $\mathcal{G}$, and let $f \in \mathcal{F}$ and $\alpha \in \mathcal{G}$. Then $f \models \alpha$ if and only if for every $u \in \Gamma_f^B$, $\alpha(u) = 1$.*

Thus we need to look only at the set $\Gamma_f^B$ to decide whether $f$ implies a set in $\mathcal{G}$.

## 2.2 Applications

We now apply the general theory developed above to specific classes of Boolean functions.

We say that queries are *common* if they are taken from some common function class[1] as defined below.

**Definition 2.5** *A class of functions $\mathcal{F}$ is* common *if there is a small (polynomial size) fixed basis for all $f \in \mathcal{F}$.*

In [KR94b] it is shown that some important function classes are common. Those include: (1) Horn-CNF formulas, (2) reversed Horn-CNF formulas (CNF with clauses containing at most one *negative* literal), (3) $k$-quasi-Horn formulas (a generalization of Horn theories in which there are at most $k$ positive literals in each clause), (4) $k$-quasi-reversed-Horn formulas and (5) $\log n$CNF formulas (CNF in which the clauses contain at most $O(\log n)$ literals).

The basis for the class of Horn-CNF formulas is the set of assignments $B_H = \{u \in \{0,1\}^n \mid \text{weight}(u) \geq n-1\}$. The basis for the class of $k$-quasi-Horn formulas is the set of assignments $B_{H_k} = \{u \in \{0,1\}^n \mid \text{weight}(u) \geq n-k\}$. By flipping all the bits in each of the basis elements one gets a basis for the reversed classes. The basis for the class of $\log n$CNF formulas is derived using a combinatorial construction called an $(n,k)$ set [ABN$^+$92]. For more details see [KR94b]. Here we need the following observations: (1) $|B_H| = n+1$, (2) $|B_{H_k}| = O(n^k)$, and (3) $|B_{\log n-CNF}| = O(n^3)$.

We note that if we have two common classes, and correspondingly two bases then the union of these bases is a union for the combined class. Hence, denoting by $\mathcal{L}_C$ the set of formulas that can be represented as a CNF with clauses from any of the above classes, we have that $\mathcal{L}_C$ is a common class. We denote the basis for $\mathcal{L}_C$ by $\mathcal{B}_C$.

**Theorem 2.7** *Let $f$ be a Boolean function on $n$ variables. Then there exists a fixed set of models $\Gamma = \Gamma_f^{\mathcal{B}_C}$, such that for any common query $\alpha \in \mathcal{L}_C$, model-based deduction using $\Gamma$, is correct.*

## 2.3 The Size of $\Gamma$

The size of the model-based representation used is an important factor in the complexity of reasoning with it. The following lemma gives a bound on this size.

**Lemma 2.8** *Let $B$ be a basis for the Boolean function $f$, and denote by $|DNF(f)|$ the size of its DNF representation. Then, the size of the model-based representation of $f$ is*

$$|\Gamma_f^B| \leq \sum_{b \in B} |min_b(f)| \leq |B| \cdot |DNF(f)|.$$

[1] Note that a fixed basis uniquely characterizes a family of Boolean functions which can be represented using it. There are of course other ways to characterize classes of functions which do not correspond to any basis (e.g. some subset of DNF).

We note that this bound is tight in the sense that for some functions the size of the DNF is indeed needed. It does however allow for an exponential gap in other cases. Namely, there are functions with an exponential size DNF and a linear size model-based representation [KR94b]. It is also interesting to compare the size of this representation to the size of other representations for functions. Examples in [KKS93] show that there are cases where the (Horn CNF) formula representation is small and the model-based representation is exponentially large, and vice versa. For a discussion of these issues see [KR94b].

# 3 Relational Databases

In this section we introduce some of the basic notions in relational databases, and discuss the correspondence between them and notions in the Boolean domain.

## 3.1 Relations and Dependencies

We assume a finite set $U$ of *attributes*. A *tuple* (over $U$) is a mapping with domain $U$, and a *relation* (over $U$) is a set of tuples (over $U$). If $X \subseteq U$, and if $t$ is a tuple over $U$, then we denote the restriction of $t$ to $X$ by $t[X]$. If $R$ is a relation over $U$, then $R[X] = \{t[X] \mid t \in R\}$. If $A$ is an attribute of $U$, and if $t$ is a tuple over $U$, then we may refer to $t[A]$ as an *entry*, in the $A$ *column*.

A *functional dependency* (over $U$), an FD, is a statement, $X \rightarrow Y$ where $X, Y \subseteq U$. A relation $R$ over $U$ *obeys* the FD $X \rightarrow Y$ if whenever $t_1, t_2$ are tuples of $R$ with $t_1[X] = t_2[X]$, then $t_1[Y] = t_2[Y]$. We also say then that FD *holds* for $R$. If the FD does not hold for $R$, then we say that $R$ *violates* the FD.

A *Boolean Dependency*, BD, is an arbitrary Boolean combination of attributes. The semantics are defined on the same lines as in functional dependencies. For example the dependency $A \rightarrow B \vee \neg C$ means that if for two tuples $t_1$ and $t_2$ we have $t_1[A] = t_2[A]$ then either $t_1[B] = t_2[B]$ or $t_1[C] \neq t_2[C]$.

While Boolean dependencies as such are not very often used in database design, the following extension of them is quite useful. Associate with each attribute $A \in U$ an equivalence relation $E_A$ on the domain (set of possible values) of $A$. Define that $A \rightarrow B \vee \neg C$ holds if and only if for any two tuples $t_1$ and $t_2$ we have $(t_1[A], t_2[A]) \in E_A$ then either $(t_1[B], t_2[B]) \in E_B$ or $(t_1[C], t_2[C]) \notin E_C$. Using this generalization, we can, e.g., express the following statement about insurance policy holders. Assume that we have attributes Age, Premium, and Sex, and that we define equivalence relations for age groups and premium groups. Then we can state the constraint "if two policy holders belong to the same age group, then their premiums are in the same class or they are of different sexes".

In the sequel we consider only Boolean dependencies; the extension to the above class is straightforward.

If $\Sigma$ is a set of dependencies and $\sigma$ a single dependency, we say that $\Sigma$ *logically implies* $\sigma$, and denote $\Sigma \models \sigma$ if whenever every dependency in $\Sigma$ holds for a relation $R$, then also $\sigma$ holds for $R$. If $\Sigma \not\models \sigma$, then there is a relation $R_\sigma$ (a witness) such that $R_\sigma$ obeys $\Sigma$ but not $\sigma$.

Mapping a Boolean dependency into a Boolean function is straightforward; simply map every attribute to a Boolean variable with the same name. Formally, we assume that the set of attributes $U$ is of size $n$, and correspondingly discuss the Boolean cube $\{0,1\}^n$. We map the attributes in $U$ to the set of $n$ Boolean variables $\{x_1, \ldots x_n\}$ which, for convenience, we denote also by $U$. For example, the FD $\sigma$, $X \rightarrow Y$, corresponds to the Boolean formula $f_\sigma$, $\bigwedge_{x_i \in X} x_i \rightarrow \bigwedge_{x_j \in Y} x_j$.

The following notation is useful when discussing relations. Let $t_1, t_2$ be a set of tuples, and let $X \subseteq U$ be a set of attributes. We say that $t_1$ and $t_2$ *agree exactly* on $X$ if $t_1[X] = t_2[X]$, and if $t_1[A] \neq t_2[A]$ for each attribute $A \notin X$. For a relation $R$ we define

$$agr(R) = \{X \subseteq U \mid \text{there is a pair of distinct tuples in } R \text{ that agree exactly on } X \ \}.$$

Given a relation $R$ we associate with it a set of assignments in $\{0,1\}^n$ as follows: with every set of attributes $X = \{x_{i_1}, \ldots, x_{i_j}\} \in agr(R)$ we associate an element $z^X \in \{0,1\}^n$ as follows: $z_i^X$, the $i$-th bit of $z^X$ is 1 if and only if the $i$-th attribute $x_i$, is in $X$. We denote the set of assignments in $\{0,1\}^n$ that corresponds to the agree set of the relation $R$ by $R_{agr}$.

**Claim 3.1** *The relation $R$ obeys the Boolean dependency $\sigma$ if and only if for all $z \in R_{agr}$, $f_\sigma(z) = 1$.*

**Proof:** Let $R$ be a relation that obeys $\sigma$ and let $t_1, t_2$ be two tuples in $R$. Then, by definition, $z^{agr\{t_1,t_2\}} \in \{0,1\}^n$ satisfies $f_\sigma$.

For the other direction, assume that $R$ does not obey $\sigma$. Then, there are two tuples $t_1, t_2$ in $R$ such that the set of attributes $Z = agr(t_1, t_2)$ provides a counterexample for $\sigma$. Therefore, by definition, $z^{agr\{t_1,t_2\}} \in \{0,1\}^n$ does not satisfy $f_\sigma$. ■

The following theorem shows that, with a small caveat, the semantics of dependencies is equivalent to that of the Boolean formulas. The theorem was first reported in [SDPF81] and a small caveat reported in [BB88] implies that it holds only under some restrictions. We discuss this issue briefly.

The definition of when a relation $R$ obeys the FD $X \rightarrow Y$ does not specify whether the tuples $t_1$ and $t_2$ have to be distinct or not. It turns out that neither choice yields an interpretation which is semantically equivalent to Boolean formulas implication. The source of the problem is that if the tuples are not distinct, then the assignment $1^n$ is always in $R_{agr}$, and otherwise it is never in $R_{agr}$. There are several possible solutions for this problem. In order to be consistent with the standard definition for the semantics given in first order logic we choose to restrict our discussion to Boolean dependencies $\Sigma$, such that $f_\Sigma(1^n) = 1$. This avoids the problem altogether. (Our results however do not depend on this choice and hold for the other solutions too.) Therefore, in the following whenever we refer to Boolean dependencies we mean Boolean dependencies which satisfy the assignment $1^n$.

**Theorem 3.2 ([SDPF81, BB88])** *Let $\Sigma$ be a set of Boolean dependencies, and $\sigma$ a Boolean dependency. Let $f_\Sigma$ and $f_\sigma$ be the corresponding Boolean functions. Then $\Sigma \models \sigma$ if and only if $f_\Sigma \models f_\sigma$.*

## 3.2 Armstrong Relations

Next we introduce the notion of *Armstrong relations* that will turn out to be analogous to the notion of characteristic model in the Boolean case. An Armstrong relation appeals to our notion of reasoning with examples. To test whether a dependency follows from our knowledge we would test whether it holds in the Armstrong relation and decide accordingly. Unlike the Boolean case where the existence of a set of models with this property is trivial, for database dependencies there in no *a priori* guarantee that there exists a finite set of relations such that checking only those yields correct results.

Recall that if $\Sigma \not\models \sigma$, then there is a relation $R_\sigma$ (a witness) such that $R_\sigma$ obeys $\Sigma$ but not $\sigma$. Let $\mathcal{F}$ be some class of dependencies, and $\Sigma$ a set of dependencies in $\mathcal{F}$.

**Definition 3.1** *An* Armstrong relation *for $\Sigma$ (with respect to $\mathcal{F}$) is a relation $R$ which obeys $\Sigma$ and such that for every $\sigma \in \mathcal{F}$ for which $\Sigma \not\models \sigma$, $R$ does not obey $\sigma$.*

That is, an Armstrong relation for $\Sigma$ (with respect to $\mathcal{F}$) is a global witness, a relation that simultaneously serves the role of witness $R_\sigma$ for every $\sigma \in \mathcal{F}$ which is not a consequence of $\Sigma$. If every set of dependencies $\Sigma$ in $\mathcal{F}$ has an Armstrong relation then we say that $\mathcal{F}$ enjoys Armstrong relations. So, Armstrong relations is a property of a class of dependencies rather than a single dependency.

In the following we discuss the existence of Armstrong relations, generalizing a result on Armstrong relations for FDs proved in [BDFS84].

**Definition 3.2** *A Boolean function $f$ is* clipped *if it can be written as $f = g_1 \wedge g_2$, where $g_1$ is a (possibly empty) conjunction of positive literals and $g_2$ does not depend on any of the variables that appears in $g_1$, and is satisfied by the assignment $0^n$.*

**Definition 3.3** *Let $M \subseteq \{0,1\}^n$ be a set of assignments. We say that $M$ is clipped if either (1) $0^n \in M$ or (2) there is a set of attributes $C = \{x_{i_1}, x_{i_2}, \ldots, x_{i_m}\}$ such that (2.1) for all $x \in M$ and for all $x_i \in C$, $x_i$ is assigned 1 in $x$, and (2.2) the assignment in which all the literals in $C$ are set to 1 and all other literals are set to 0 is in $M$.*

It is easy to observe that a function is clipped if and only if its set of models is clipped, and that any set of FDs is clipped. The following claim gives a different characterization of this class.

**Claim 3.3** *A Boolean function $f$ is clipped if and only if it has a unique minimal model (i.e. its set of models has a minimum).*

**Proof:** Suppose that $f$ is clipped. If $0^n \in f$ then it is the unique minimal model. Otherwise, let $C$ be the set of literals from the definition. Then, the assignment in which all the literals in $C$ are set to 1 and all other literals are set to 0 is the unique minimal model.

For the other direction, suppose that $f$ has a unique minimal model $x$. Let $C$ be the set of variables which are set to 1 in $x$. Then, any model of $f$ must have all the variables in $C$ mapped to 1, or otherwise $x$ will not be a minimum. Further, the assignment in which all the literals in $C$ are set to 1 and all other literals are set to 0, is exactly $x$. So the set of models of $f$ is clipped and $f$ is clipped. ■

We now show how, given a clipped set of assignments $M$, we can build a relation $R(M)$ such that $R(M)_{agr} = M$. Namely, the agree set of $R(M)$ corresponds exactly to the set $M$. This is essentially the "disjoint union" construction used in [Fag82b, BDFS84].

**Claim 3.4** *For any clipped set of assignments $M \subseteq \{0,1\}^n$ there is a relation $R(M)$ such that the number of tuples in $R(M)$ is $2|M|$ and $R(M)_{agr} = M$.*

**Proof:** First consider the case in which $0^n \in M$. Order the assignments in $M$ arbitrarily, and for each assignment in $M$ construct a pair of "sibling" tuples in $R(M)$ as follows: for the $i$'th assignment construct one tuple with all attributes mapped to the value $2i$, and a second tuple in which the bits assigned 1 in the assignment are mapped to $2i$ and the bits assigned 0 are mapped to $2i + 1$. For example if $i = 4$ and the *i'th* assignment is (010) then the tuples added to $R(M)$ are (8 8 8) and (9 8 9). Since tuples generated form different assignment do not agree on any attribute, and the agree set of two sibling tuples corresponds exactly to the 1 bits in the assignment that generated them we have that $R(M)_{agr} = M$.

If $0^n \notin M$ we are guaranteed that there is a set of attributes $C$ such that all the variables in $C$ are assigned 1 in all the assignments in $M$. We construct a relation with all of the attributes in $C$ set to the same value, say 0, and for the other attributes we use the same construction as before. Clearly, the agree set of two sibling tuples corresponds to the assignment that generated them. Further, the agree set of two non-sibling tuples is exactly $C$, but this corresponds to the assignment with all attributes in $C$ mapped to 1 and all other attributes mapped to 0, which is in $M$. ■

**Claim 3.5** *The class of clipped Boolean Dependencies enjoys Armstrong relations.*

**Proof:** Let $\Sigma$ be a set of BDs, and $f_\Sigma$ its Boolean counterpart. We first observe that reasoning with models with the set of all models of $f_\Sigma$ is correct for all Boolean queries. Let $M$ denote this set of models, and let $R(M)$ be the relation guaranteed by Claim 3.4. Namely, $R(M)_{agr} = M$. Claim 3.1 implies that $R(M)$ is an Armstrong relation of $\Sigma$. ■

This is a generalization of the result on Armstrong relations for FDs [BDFS84]. As observed in the above proof, in the Boolean domain, every function has "Armstrong sets": the set of all models of $f$ is an "Armstrong set". In the use of example-based reasoning for database dependencies, however, we want to use a single relation as the example. It is not always possible to capture all the assignments in this set and no other assignment, using the agree set of a single relation[2]. Therefore we need the restriction to clipped functions. We give an example for this phenomena in the appendix.

# 4 Armstrong Relations As Characteristic Models

For the Boolean domain, the possibility of using example-based reasoning is clear from the outset: to reason about a function $f$ by using examples one can alway use the set of all models of $f$. The problem there is whether there exists a small set of models of $f$ that support correct deduction.

As discussed above, the situation is different for database dependencies. Dependencies are statements about arbitrary relations (even possibly infinite ones), and hence there in no *a priori* guarantee that there exists a finite set of relations such that checking only those yields correct results. We have discussed the existence issue earlier. We now show how results from the theory of reasoning with models, introduced in Section 2, can be used in the study of Armstrong relations.

The concept of generators, defined below, has been introduced by Beeri et. al. [BDFS84] for the purpose of studying Armstrong relations for functional dependencies. Let $\Sigma$ be a set of FDs, over the set $U$ of attributes. A subset $V \subseteq U$ is *closed* if for every dependency $X \rightarrow Y$, in $\Sigma$, for which $X \subset V$, also $Y \subset V$. It is easy to see that the intersection of closed sets is closed, and that the minimal closed set containing $X$ is $X^*$, the set of all attributes $A$ such that $\Sigma \models X \rightarrow A$. We denote by $CL(\Sigma)$ the family of closed sets defined by $\Sigma$, and by $GEN(\Sigma)$ the *intersection generators* of $CL(\Sigma)$. If $M$ is a family of subsets of a finite set, closed under intersection, the smallest set $M'$ such that $M = \{S_1 \cap \ldots \cap S_k \mid S_i \in M'\}$ is the set of *intersection generators* of $M$. It is easy to show that $M'$ is uniquely defined. Similar definitions in the Boolean domain have been given

[2] We could however talk on a set of relations which serve as an Armstrong set instead of a single relation. It is easy to observe that such sets always exist, using the two tuple relations separately in the set, instead of collating them all into one relation. We would not pursue this further here.

by [KKS93]. Let $gen(\Sigma)$ denote the Boolean counterpart of $GEN(\Sigma)$. That is, for every subset $Z$ in $GEN(\Sigma)$ construct the assignment $z^X$ as in the construction of the set $R_{agr}$. The following theorem shows that this notion coincides with the notion of characteristic models (Definition 2.3).

**Theorem 4.1 ([KR94b])** *Let $\Sigma$ be a set of FDs and $f_\Sigma$ it Boolean counterpart. Then, $\Gamma^{B_H}_{f_\Sigma} = gen(\Sigma)$.*

The following theorem gives another alternative definition for the set $GEN(\Sigma)$. Denote by $MAX(\Sigma)$ the collection of all attribute sets $X$ such that there exists an attribute $A \in R$ such that $\Sigma \not\models X \rightarrow A$, but for any superset $Y$ of $X$, $\Sigma \models Y \rightarrow A$.

**Theorem 4.2 ([MR86])** *Let $\Sigma$ be a set of FDs. Then $MAX(\Sigma) = GEN(\Sigma)$.*

It is well known [Fag82b, BDFS84] that functional dependencies enjoy Armstrong relations. Beeri et al. [BDFS84] have also shown the correspondence between generators and Armstrong relations for functional dependencies.

**Theorem 4.3 ([BDFS84])** *Let $\Sigma$ be a set of FDs, then a relation $R$ is an Armstrong relation (with respect to FDs) for $\Sigma$ if and only if $GEN(\Sigma) \subseteq agr(R) \subseteq CL(\Sigma)$.*

In the following theorems we use the theory for reasoning with models to show that this property holds in more general cases:

**Theorem 4.4** *Let $\mathcal{F}$ be a set of Boolean functions, $B$ a basis for $\mathcal{F}$, $\Sigma$ be a set of dependencies in the class corresponding to $\mathcal{F}$ and $M(\Sigma)$ the set of all models of $f_\Sigma$. Then*
*(1) If $\Gamma^B_{f_\Sigma} \subseteq R_{agr} \subseteq M(\Sigma)$ then $R$ is an Armstrong relation for $\Sigma$ with respect to $\mathcal{F}$.*
*(2) If $R$ is an Armstrong relation for $\Sigma$ with respect to $\mathcal{F}$ then $R_{agr} \subseteq M(\Sigma)$.*

**Proof:** Part (1) follows from Theorem 2.6, noting that adding models of $f_\Sigma$ to the set $\Gamma$ cannot harm the correctness of the reasoning. Part (2) follows from the observation that if $R_{agr}$ has an assignment not in $M(\Sigma)$ then this assignment (by definition) falsifies $f_\Sigma$, which is a dependency in the class $\mathcal{F}$. ■

**Theorem 4.5** *Let $B$ be a set of assignments, and let $\mathcal{F}$ be the set of all Boolean functions that can be represented using $B$. Let $\Sigma$ be a set of dependencies in the class corresponding to $\mathcal{F}$, and $M(\Sigma)$ the set of all models of $f_\Sigma$. If $R$ is an Armstrong relation for $\Sigma$ with respect to $\mathcal{F}$ then $\Gamma^B_{f_\Sigma} \subseteq R_{agr}$.*

**Proof:** Suppose that there exists an Armstrong relation $R$ such that $\Gamma^B_{f_\Sigma} \not\subseteq R_{agr}$, and consider $x \in \Gamma^B_{f_\Sigma} \setminus R_{agr}$. We show that there is a function $h \in \mathcal{F}$, not implied by $f_\Sigma$, which holds in $R$, therefore yielding a contradiction.

Define the function $g = R_{agr}$ (that is, the elements of $R_{agr}$ are all the satisfying assignments of $g$), and consider $h = g^B_{lub}$. Then, by definition[3], $h \in \mathcal{F}$, and $g \subseteq h$, which means that $R$ obeys $h$. However, since $x \in \Gamma^B_{f_\Sigma}$, $x$ is a minimal model with respect to some $b \in B$. Therefore, with respect to this element $b$, we get that for all $z \in \Gamma^B_{f_\Sigma}$, $x \not\geq_b z$. This implies that $h(x) = 0$ and therefore $h$ is not implied by $f_\Sigma$. ■

[3] Note that we have to show that $h$ is a legal Boolean dependency. That is, by our previous choice, that it satisfies $1^n$. This is guaranteed by the fact that $1^n \in R_{agr}$, and that $R_{agr} \subseteq h$.

Note the difference in the premises of the previous two theorems. Theorem 4.5 shows that every element of the $\Gamma$ constructed is necessary in order to get correct deduction. What the proof shows is that there exists a function $h$ in the class represented by $B$ which necessitates the use of each element $x$. Note that, in general, if $B$ is a basis for $\mathcal{F}$ it does not mean that all functions in the class represented by $B$ are in the class $\mathcal{F}$, and therefore the premises of Theorem 4.4 are not enough to yield this result. We note however that the bases $B_H$ and $B_{H_k}$ presented above, for the classes of Horn functions and $k$-quasi-Horn functions respectively, represent those classes exactly. That is, a function is $k$-quasi-Horn if and only if it can be represented using $B_{H_k}$.

## 5 Bounded Disjunctive Dependencies

We now derive some corollaries of the equivalence between Armstrong relations and characteristic models. Claim 3.5 shows that Armstrong relations exist for the class of clipped functions. We derive bounds on the size of such relations for special cases of this class.

Let $U = \{x_1, \ldots, x_n\}$ be a set of attributes. A Disjunctive Dependency $\sigma$ is a statement of the form

$$x_{i_1} \wedge x_{i_2} \ldots \wedge x_{i_m} \rightarrow x_{j_1} \vee x_{j_2} \ldots \vee x_{j_l}.$$

The semantics of disjunctive dependencies is the same as in Boolean Dependencies. It is easy to see that any Boolean Dependency can be transformed into a set of Disjunctive Dependencies (this is simply the conjunctive normal form CNF for Boolean functions).

**Definition 5.1** *A $(k, q)$-Bounded Disjunctive Dependency ($(k, q)$-BDD) is a statement of the form*

$$x_{i_1} \wedge x_{i_2} \ldots \wedge x_{i_m} \rightarrow x_{j_1} \vee x_{j_2} \ldots \vee x_{j_l},$$

*where $m \geq 1$ and one of the following must hold: (1) $l \leq k$, or (2) $m \leq k$, or (3) $m + l \leq q$.*

Case (1) in the definition above corresponds to $k$-quasi-Horn functions, Case (2) in the definition above corresponds to Reversed $k$-quasi-Horn functions, and case (3) to $q$-CNF. We would refer to these sub cases as type (i) BDDs for $i \in \{1, 2, 3\}$ respectively. Note that BDDs are defined so that they are clipped and therefore this class enjoys Armstrong relations. In the appendix we give an example which shows that if we lift the restriction $m \geq 1$ then 3-quasi-Horn functions do not enjoy Armstrong relations, and therefore implies that the restriction is necessary. We can now derive bounds on the size of minimal Armstrong relations. The next two theorems follow from the combination of Theorem 4.4, Claim 3.4, Lemma 2.8 and the bound on the sizes of the basis for the corresponding classes.

**Theorem 5.1** *Let $\Sigma$ be a set of $(k, q)$-BDDs. If $q = O(\log n)$ then the size of the* minimal *Armstrong relation of $\Sigma$, with respect to $(k, q)$-BDDs, is $O(p_1(n) \cdot |DNF(f_\Sigma)|)$, where $p_1(n) = O(n^3 + n^k)$.*

Sagiv et. al. [SDPF81] studied the class of Multi-Valued Dependencies (MVDs). They show that although MVDs cannot be described as BDs there is a set of dependencies they call degenerate MVDs which is semantically equivalent to MVDs and which can be described as BDs. The degenerate MVDs is a statement of the form $X \rightarrow Y \vee Z$ where $X, Y, Z \subseteq U$. It can be easily seen that degenerate MVDs are a subset of 2-quasi-Horn functions. This implies the following theorem:

**Theorem 5.2** *Let $\Sigma$ be a set of FDs and degenerate MVDs. Then the size of the* minimal *Armstrong relation of $\Sigma$, with respect to FDs and degenerate MVDs, is $O(p_1(n) \cdot |DNF(f_\Sigma)|)$, where $p_1(n) = O(n^2)$.*

While, FDs are not BDDs if we allow for empty antecedent (the set $X$ in the dependency $X \rightarrow Y$), we can still get the following bound:

**Theorem 5.3** *Let $\Sigma$ be a set of FDs. Then the size of the* minimal *Armstrong relation of $\Sigma$, with respect to FDs, is $O(p_1(n) \cdot |DNF(f_\Sigma)|)$, where $p_1(n) = O(n)$.*

**Proof:** The claim follows from the combination of Theorem 4.3, Theorem 4.1, Claim 3.4, Lemma 2.8 and the bound on the size of $B_H$. ∎

Note that, if we add FDs (with empty antecedent), to BDDs, then a set of dependencies is not necessarily clipped. For example $\Sigma = \{x_1, (x_1 \rightarrow (x_2 \vee x_3))\}$ is not clipped. We can, however, get a bound on the size of Armstrong relations for the union of these classes. This follows from the same arguments as in Theorem 5.1, noting that the basis for $k$-quasi-Horn functions is also a basis for Horn functions.

**Theorem 5.4** *Let $\Sigma$ be either a set of $(k,q)$-BDDs or a set of FDs, where $q = O(\log n)$. Then the size of the* minimal *Armstrong relation of $\Sigma$, with respect to queries which are either FDs or $(k,q)$-BDDs, is $O(p_1(n) \cdot |DNF(f_\Sigma)|)$, where $p_1(n) = O(n^3 + n^k)$.*

For the case where Theorem 4.5 holds we can also get a lower bound. As mentioned before the basis $B_{H_k}$ corresponds exactly to the class of $k$-quasi-Horn functions. However, the restriction $m \geq 1$ in the definition of type (1) BDDs violates this exact correspondence. Let UBDD denote the class of type (1) and type (2) BDDs with the restriction $m \geq 1$ removed. As mentioned above, this class does not enjoy Armstrong relation. It may still happen, though, that some set of dependencies in the class has an Armstrong relation with respect to this class (in particular all type (1) BDDs do). In such cases the following lower bound holds.

**Theorem 5.5** *Let $\Sigma$ be a set of UBDDs. Then the size of the* minimal *Armstrong relation of $\Sigma$, with respect to UBDDs, is $\Omega(\sqrt{|\Gamma^B_{f_\Sigma}|})$ where $B$ is the basis for $k$-quasi-Horn functions and Reversed $k$-quasi-Horn functions.*

**Proof:** The proof follows from the observation [BDFS84] that the agree set of a relation with $m$ tuples has at most $\binom{m}{2}$ elements, together with Theorem 4.5 and Claim 3.1. ∎

# 6 Translating between Relations and Dependencies

In this section we use the equivalence between Armstrong relations and characteristic models to discuss the issue of translating between different representations of relational database, namely representation via dependencies and via relations. This issue is important, in particular, in the context of designing relational databases.

It has been suggested [MR86] that the design of databases can be benefited from translations between relations and dependencies. The Designer, in this scheme, tries to specify some knowledge which is not available explicitly, with the help of a computerized design tool.

The process start by the designer suggesting a set of dependencies. In return, the design tool computes an Armstrong relation for these dependencies, and presents it to the designer. The designer inspects the relation, and if it is found unsatisfactory, presents an alternative relation which captures the intuition better. In return, the design tool computes a set of dependencies for this relation, and presents it to the designer. This process goes on for several stages, where the designer modifies the representations whenever it is found unsatisfactory.

The task of the design tool is therefore to translate from relations to dependencies and vice versa. Unfortunately, no polynomial time algorithm for these tasks have been found, even for the restricted case of functional dependencies. In fact, the complexity of the problem has been studied [MR86, EG94, FK94, Kha95] but is still an open problem. In general, these problems are at least as hard as the hypergraph transversal problem. In [EG94, Kha95] it is shown that certain special cases are equivalent to the latter, and therefore using the algorithm in [FK94] these can be solved in sub-exponential time $n^{O(\log n)}$. Furthermore, an immediate corollary from the equivalence shown here, and recent results on characteristic models [Kha95], is that the complexity of the two translation problems is equivalent under polynomial reductions.

While we do not solve the problems here we do show how some alternative translations can be performed. First, for translation from a relation to a set of dependencies, we show that an alternative closed form can be given for the set of dependencies. While not in traditional dependencies form, it does convey some structure and can still be presented to a designer for inspection.

Secondly, for translating a set of dependencies to a relation we show how an "approximate" relation can be computed, which disagrees with the dependencies on at most a small fraction of possible tuples.

## 6.1 A Closed Form from Armstrong relations.

Given a set of characteristic models $\Gamma$ and a basis $B$ we can give a closed form for the function that these describe. In particular we have

$$f = \wedge_{b_B} \vee_{z \in \Gamma} \mathcal{M}_b(z).$$

We first observe that the function $\mathcal{M}_b(z)$ for an assignment $z$ is a conjunction of literals. Simply take $t_{z,b} = \wedge_{z_i \neq b_i} x_i^{z_i}$, where $x_i^0 = \overline{x_i}$ and $x_i^1 = x_i$. This implies that the closed form we get is a depth 3 circuit: a conjunction of disjunction of conjunctions,

$$f = \wedge_{b_B} \vee_{z \in \Gamma} t_{z,b}.$$

In Boolean terms, this allows for the evaluation of the function $f$ given its set of characteristic models. As discussed above, in the context of designing a relational database, this is a conjunction of disjunctive restrictions which may be useful for a human inspector.

## 6.2 Armstrong relations from Dependencies

We now consider the relation inference problem. Namely, given a set of dependencies $\Sigma$ as input the problem is to compute an Armstrong relation $R$ for this set of dependencies. While the complexity of this problem is an open question, we present an algorithm which "approximately" solves this problem. The output of the algorithm is a relation $R$ which, relative to the uniform probability measure $D$ on $\{0,1\}^n$, disagrees with $\Sigma$ on a small fraction of the distribution. The complexity of the algorithm depends on the DNF size of the input.

Approximate inference solutions for the problem of dependency inference have been studied before [KM94], where the output was a set of dependencies which approximated the input relation in a similar sense. Using the monotone theory we apply the same technique to the problem of relation inference. Approximating a relation in such a way is useful when answering queries. In [KR94a] it has been shown that it is sufficient to answer a large set of queries correctly.

The results presented here draw on previous results in computational learning theory. In this framework a function $f : \{0,1\}^n \rightarrow \{0,1\}$ is hidden from a learner that has to reproduce it by accessing certain "oracles". A membership query allows the learner to find the value of the function on a certain point.

**Definition 6.1** *A membership query oracle for a function $f : \{0,1\}^n \rightarrow \{0,1\}$, denoted $MQ(f)$, is an oracle that when presented with $x \in \{0,1\}$ returns $f(x)$.*

An equivalence query allows the learner to find out whether the current hypothesis is equivalent to $f$ or not. In case it is not equivalent the learner is supplied with a counterexample.

**Definition 6.2** *An equivalence query oracle for a function $f : \{0,1\}^n \rightarrow \{0,1\}$, denoted $EQ(f)$, is an oracle that when presented with a hypothesis $h : \{0,1\}^n \rightarrow \{0,1\}$, returns Yes if $f \equiv h$. Otherwise it returns No and a counterexample $x$ such that $f(x) \neq h(x)$.*

We use a result that has been obtained in this framework. The result is due to Bshouty [Bsh93] and its relation to characteristic models has been pointed out in [KR94a].

**Theorem 6.1** *Let $f$ be a Boolean function and let $B$ be a monotone basis for $f$. There is an algorithm $A$ that, on input $B$, when given access to $MQ(f)$ and $EQ(f)$, runs in time polynomial in the number of variables and in the DNF size of $f$, and outputs the set $\Gamma = \Gamma_f^B$.*
*The hypothesis $h$ the algorithm uses when accessing $EQ(f)$ is always in the form $h = \wedge_{b \in B} \vee_{z \in G} \mathcal{M}_b(z)$, where $G$ is a set of assignments such that $G \subseteq f$.*

**Theorem 6.2** *Let $\Sigma$ be a set of Functional Dependencies and $B = B_H$ a monotone basis for $\Sigma$. There is a randomized algorithm APPROX that on input $0 < \epsilon, \delta < 1, \Sigma, B$ computes a relation $R$ such that the function $f_R = \wedge_{b \in B} \vee_{z \in R_{agr}} \mathcal{M}_b(z)$ satisfies*
*(1) $f_R \models f_\Sigma$ and*
*(2) with probability $> 1 - \delta$, $Prob_D[f_\Sigma \setminus f_R] < \epsilon$, where $D$ is the uniform distribution over $\{0,1\}^n$, and*
*the algorithm is polynomial in $n, 1/\epsilon, 1/\delta$ and the DNF size of $\Sigma$.*

**Proof:** The algorithm APPROX will run the algorithm $A$ from Theorem 6.1 and answer the $MQ$ and $EQ$ queries that $A$ presents.

Given $x \in \{0,1\}^n$ for $MQ$ the algorithm tests whether $x$ satisfies $f_\Sigma$ and answers yes or no accordingly.

Given a a set of assignments $G$ for $EQ$ we have to test whether $f_\Sigma$ is equivalent to $h = \wedge_{b \in B} \vee_{z \in G} \mathcal{M}_b(z)$. The algorithm APPROX will draw $m = (1/\epsilon)\log(1/\delta)$ assignments in $\{0,1\}^n$ according to $D$ and will evaluate $f_\Sigma$ and $h$ on these assignments. If an assignment $x$ such that $h \neq f_\Sigma$ is found then $x$ is returned as a counterexample. Otherwise APPROX says that the functions are equivalent, and stops the simulation of $A$.

APPROX then finds the unique minimal assignment of $f_\Sigma$, and adds it to $G$. This is possible since $f_\Sigma$ is in Horn form. Then it outputs the relation $R(G)$ such that $R(G)_{agr} = G$, guaranteed by Claim 3.4.

To prove (1) note that since Theorem 6.1 guarantees that the assignments in $G$ satisfy $f_\Sigma$, Corollary 2.4 implies that $f_R = h \models f_\Sigma$.

To prove (2) note that when the algorithm stops only it could not find a counterexample in a random sample of size $m$. Suppose that $Prob[h \neq f] > \epsilon$ then the probability that $m$ independent samples did not find a counterexample is at most $(1-\epsilon)^m < \delta$.

Finally, note that algorithm $A$ is guaranteed to run in time polynomial in $n$ and the DNF size of $f_\Sigma$, and the samples we take are polynomial in $1/\epsilon, 1/\delta$. ■

Note that the proof depends on the restriction to Functional Dependencies only for the task of finding the minimum assignment, which makes $G$ clipped. It can therefore be generalized to any class in which this is possible, and in particular to BDDs where all dependencies satisfy $0^n$.

# 7 Abductive Explanations as Keys

In this section we show another connection between notions in database theory and the propositional domain. In particular we show a close relation between abductive explanations in the propositional domain and keys in relational databases.

Abduction is the task of finding a minimal explanation to some observation. Formally (see, e.g., [RDK87]), the reasoner is given a Boolean function KB (the *background theory*), a set of propositional letters $A$ (the *assumption set*), and a query letter $q$. An *assumption based explanation* of $q$, with respect to the background theory KB, is a minimal subset $E \subseteq A$ such that

1. KB$\wedge(\wedge_{x \in E} x) \models q$ and
2. KB$\wedge(\wedge_{x \in E} x) \neq \emptyset$.

Thus, abduction involves tests for entailment and consistency, but also a search for an explanation that passes both tests.

When the set $A$ of assumptions includes *all* the propositional letters, the set $E$ is simply called an *explanation*. In this case the task of computing an explanation is, on input KB,$q$, to compute an explanation $E$ for $q$ with respect to $KB$.

Given a set $\Sigma$ of functional dependencies over a set $U$ of attributes, a key for $U$, with respect to a set $\Sigma$ of Boolean dependencies, is a set $X \subseteq U$ such that $\Sigma \models X \rightarrow U$, but no proper subset of $X$ has this property. That is, a key is a minimal set of attributes that functionally determines all attributes. The task of computing a key is, on input $\Sigma$, to compute a key $X$ for $U$ with respect to $\Sigma$.

Let $\mathcal{S}$ be a class of Boolean dependencies, which includes all the dependencies of the form $x_i \rightarrow x_j$. For example $\mathcal{S}$ can be the class of functional dependencies, or the class of $(k, q)$-BDDs. Let $F_{\mathcal{S}}$ be the class of Boolean functions which corresponds to $\mathcal{S}$.

**Theorem 7.1** *The problem of computing an abductive explanation with respect to functions in $F_{\mathcal{S}}$ is computationally equivalent (under polynomial reductions) to the problem of computing a key with respect to dependencies in $\mathcal{S}$.*

**Proof:** Assume first that $\Sigma \in \mathcal{S}$ is a set of Boolean dependencies over a set of attributes $U$. Given an algorithm that computes an abductive explanation efficiently, we can use it to compute a key for $U$. Let $f_\Sigma$ be the corresponding Boolean function over the variables $\{x_1, \ldots x_n\}$, and $q$ be an additional propositional letter. Denote by KB the background theory consisting of the function

$$KB = f_\Sigma \wedge ((\bigwedge_{x_i \in U} x_i) \rightarrow q).$$

Then $X$ is a key for $U$ with respect to $\Sigma$ if and only if $(\bigwedge_{x_i \in X} x_i)$ is an explanation for $q$ with respect to KB.

For the other direction, we are given a Boolean function $f \in F_{\mathcal{S}}$ over $\{x_1, \ldots x_n\}$, and assume we are looking for an explanation for $x_j$. Consider the function $g = f \bigwedge \wedge_{i \neq j}(x_j \rightarrow x_i)$, and its corresponding set of Boolean dependencies $\Sigma_g$. Then $(\bigwedge_{x_i \in X} x_i)$ is an explanation for $x_j$ with respect to $f$ if and only if $X$ is a key for $U$ with respect to $\Sigma_g$. ■

As one can expect from the above equivalence, similar results have been derived in the two domains. For the case where $\Sigma$ is a Horn theory, Theorem 1 of [SL90] gives an $O(k||\Sigma||)$ algorithm for producing an explanation, where $k$ is the number of propositional variables and $||\Sigma||$ is the number of occurrences of symbols in $\Sigma$. This corresponds to a result of [Kun85], showing how to compute keys with respect to set of dependencies consisting only of definite Horn clauses. Using his result and the equivalence theorem above, we can find explanations in time $O(K||\Sigma||)$, where $K$ is the number of variables in the resulting explanation.

Furthermore, Theorem 2 of [SL90] shows that it is NP-complete to determine whether a propositional letter occurs in an explanation. This corresponds to the late 1970's result of [LO78]: it is NP-hard to determine whether an attribute is prime, i.e., occurs in a key.

The task of computing an assumption based explanation is NP-Hard when the input is given as a propositional expression [SL90]. However, if the input is given as a set of characteristic models then this task has a polynomial time algorithm [KKS93, KR94b]. Using the above equivalence, this algorithm can be applied to find keys, restricted to certain subset of the attributes, when the input is an Armstrong relation.

## 8 Conclusions

We have revealed a useful connection between theories for relational databases and theories for automated reasoning. The notion of Armstrong relation for relational databases has been shown to be equivalent to the notion of characteristic models in the theory for automated reasoning. Some corollaries of this correspondence have been developed.

Using the results from the automated reasoning domain we derived bounds for the size of Armstrong relations for functional dependencies. We then presented a new family of Bounded Disjunctive Dependencies, which generalizes the class of FDs. This family enjoys Armstrong relations, and similar size bounds have been derived for it.

We then discussed the issue of the translation between the relations and dependencies representations of relational databases. We have shown that a closed form, which conveys the exact information in the relation, can be easily computed for any relation. Further, given a set of dependencies, we can compute a relation which approximately captures the information in the dependencies.

Lastly, we have shown that there is a close relation between keys and abductive explanations in these domains, and that similar results have been found independently in the two fields.

We believe that studying the correspondence between the fields can be fruitful to both.

## A Disjunctive Dependencies do not enjoy Armstrong Relations

We give an example which shows that if we allow for empty antecedents in BDDs then the class does not enjoy Armstrong relations, even for $k = 3$.

Let $\Sigma = (x_1 \vee x_2) \wedge (x_1 \vee x_3) = x_1 \vee x_2 x_3$. In the following when we discuss assignments in the agree set we mean the assignment $z^X$ which corresponds to the set $X$ in the agree set. We have the following observations on the agree set of any Armstrong relation for $\Sigma$.

- $\Sigma \models (x_1 \vee x_2 \vee x_3)$, and therefore the agree set of $R$ must not include the assignment $(0\ 0\ 0)$.
- $\Sigma \models (x_1 \vee x_3)$, and therefore the agree set of $R$ must not include the assignment $(0\ 1\ 0)$.
- $\Sigma \not\models (x_2 x_3 \rightarrow x_1)$, and therefore the agree set of $R$ must include the assignment $(0\ 1\ 1)$.

- $\Sigma \not\models (x_2 \vee x_3)$, and therefore the agree set of $R$ must include the assignment (1 0 0).

We now show that there does not exist a relation for which the agree set includes both (0 1 1) and (1 0 0) but neither of (0 0 0) and (0 1 0).

Let $t_1 = (a\ b\ c)$ and $t_2 = (a'\ b\ c)$ be two tuples which contribute the assignment (0 1 1) to the agree set. Note that if $agr(t_1, t_3)$ corresponds to the assignment (1 0 0) for some third tuple $t_3$, then $agr(t_2, t_3)$ corresponds to (0 0 0) which is not possible, and similarly for $t_2$. So we must use two new tuples $t_3 = (d\ e\ f)$ and $t_4 = (d\ e'\ f')$ in order to create (1 0 0). Note that since the assignment (0 0 0) is not allowed in the agree set, the tuples $t_3$ and $t_4$ must agree with $t_1$ and $t_2$ on something.

Consider first the case where $d = a$. If $e \neq b$ and $f \neq c$ then $agr(t_3, t_2)$ corresponds to (0 0 0) so this is not possible. If $e = b$ and $f = c$ then $agr(t_4, t_1)$ corresponds to (0 0 0) and this is not possible too. If $e = b$ and $f \neq c$ then $agr(t_3, t_2)$ corresponds to (0 1 0) and this is not possible too. This implies that $d \neq a$ and by symmetry we also get $d \neq a'$.

Consider now the case in which $d \neq a$ and $d \neq a'$. If $e \neq b$ and $f \neq c$ then $agr(t_3, t_1)$ corresponds to (0 0 0) so this is not possible. If $e = b$ and $f = c$ then $agr(t_4, t_1)$ corresponds to (0 0 0) and this is not possible too. If $e = b$ and $f \neq c$ then $agr(t_3, t_1)$ corresponds to (0 1 0) and this is not possible too. So there is no way in which we can construct an Armstrong relation for $\Sigma$.

Note that although the analysis is done for $n = 3$ is holds for any $n$ by simply extending the assignments discussed for the required length, since the truth value of the statements involved does not depend on the values assigned to the other attributes.

# References

[ABN+92] N. Alon, J. Bruck, J. Naor, M. Naor, and R. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on information theory*, 38(2):509–516, 1992.

[BB88] J. Berman and W.J. Blok. Positive boolean dependencies. *Information Processing Letters*, 27:147–150, 1988.

[BDFS84] C. Beeri, M. Dowd, R. Fagin, and R. Statman. On the structure of Armstorng relations for functional dependencies. *Journal of the ACM*, 31(1):30–46, 1984.

[Bsh93] N. H. Bshouty. Exact learning via the monotone theory. In *IEEE Symp. of Foundation of Computer Science*, pages 302–311, Palo Alto, CA., 1993.

[EG94] T. Eiter and G. Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM Journal of Computing*, 1994. To appear.

[Fag82a] R. Fagin. Horn clauses and database dependencies. *Journal of the ACM*, 29(4):952–985, 1982.

[Fag82b] Ronald Fagin. Armstrong databases. Research Report RJ3440, IBM, San Jose, CA, May 1982.

[FK94] M. Fredman and L. Khachiyan. On the complexity of dualization of monotone disjunctive normal forms. Technical Report LCS-TR-225, Department of Computer Science, Rutgers University, May 1994.

[Hon86] J. Hong. Proving by example and gap theorems. In *Proc. 27th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 107–116, 1986.

[JL83] P. N. Johnson-Laird. *Mental Models.* Harvard University Press, 1983.

[Kha95] R. Khardon. Translating between Horn expressions and their characteristic models. Technical Report TR-03-95, Aiken Computation Lab., Harvard University, February 1995.

[KKS93] H. Kautz, M. Kearns, and B. Selman. Reasoning with characteristic models. In *Proceedings of the National Conference on Artificial Intelligence*, pages 34–39, 1993.

[KM94] J. Kivinen and H. Mannila. Approximate inference of functional dependencies from relations. *Theoretical Computer Science*, 1994. To Appear. A preliminary version of the paper appeared in ICDT 1992.

[KR94a] R. Khardon and D. Roth. Learning to reason. In *Proceedings of the National Conference on Artificial Intelligence*, pages 682–687, 1994. Full version: Technical Report TR-02-94, Aiken Computation Lab., Harvard University, January 1994.

[KR94b] R. Khardon and D. Roth. Reasoning with models. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1148–1153, 1994. Full version: Technical Report TR-1-94, Aiken Computation Lab., Harvard University, January 1994.

[Kun85] Sukhamay Kundu. An improved algorithm for finding a key of a relation. In *Proceedings of the Fourth ACM SIGACT-SIGMOD Symposium on Principles of Database Systems (PODS'85)*, pages 189–192, New York, NY, 1985. ACM.

[LO78] C. L. Lucchesi and Sylvia L. Osborn. Candidate keys for relations. *Journal of Computer and System Sciences*, 17(2):270–279, 1978.

[MR86] H. Mannila and K. Räihä. Design by example: an application of Armstrong relations. *Journal of Computer and System Sciences*, 33(2):126–141, 1986.

[RDK87] R. Reiter and J. De Kleer. Foundations of assumption-based truth maintenance systems. In *Proceedings of the National Conference on Artificial Intelligence*, pages 183–188, 1987.

[SDPF81] Y. Sagiv, C. Delobel, D. S. Parker, and R. Fagin. An equivalence between relational database dependencies and a fragment of propositional logic. *Journal of the ACM*, 28(3):435–453, 1981.

[SK91] B. Selman and H. Kautz. Knowledge compilation using Horn approximations. In *Proceedings of the National Conference on Artificial Intelligence*, pages 904–909, 1991.

[SL90] B. Selman and H. Levesque. Abductive and default reasoning: A computational core. In *Proceedings of the National Conference on Artificial Intelligence*, pages 343–348, 1990.