Horn Clauses and Database Dependencies

RONALD FAGIN

IBM Research Laboratory, San Jose, California

Abstract. Certain first-order sentences, called "dependencies," about relations in a database are defined and studied. These dependencies seem to include all previously defined dependencies as special cases A new concept is introduced, called "faithfulness (with respect to direct product)," which enables powerful results to be proved about the existence of "Armstrong relations" in the presence of these new dependencies. (An Armstrong relation is a relation that obeys precisely those dependencies that are the logical consequences of a given set of dependencies.) Results are also obtained about characterizing the class of projections of those relations that obey a given set of dependencies.

Categories and Subject Descriptors: F.4.1 [**Mathematical Logic and Formal Language**]: Mathematical Logic; H 2.1 [**Database Management**]: Logical Design—*schema and subschema*

General Terms: Design, Languages, Theory

Additional Key Words and Phrases: Armstrong relation, Armstrong database, database dependencies, Horn clause, relational database, faithfulness

1. *Introduction*

Certain sentences about relations are of special practical and/or theoretical interest for relational databases. For historical reasons, such sentences are usually called *dependencies*. The first dependency introduced and studied was the *functional dependency*, or *FD*, due to Codd [14]. As an example, consider the relation in Figure 1, with three columns: EMP (which represents employees), DEPT (which represents departments), and MGR (which represents managers). The relation in Figure 1 obeys the FD DEPT → MGR, which is read "DEPT determines MGR." This means that whenever two tuples (that is, rows) agree in the DEPT column, then they necessarily agree also in the MGR column. The relation in Figure 2 does not obey this FD, since, for example, the first and fourth tuples agree in the DEPT column but not in the MGR column. FDs (and some of the other dependencies we discuss) are of interest in database *normalization*. For example, assume that the database obeys the FD DEPT → MGR as a constraint (i.e., that it is decreed to be always the case that two employees in the same department necessarily have the same manager). Then it might be better to store the data not in one relation, as in Figure 1, but rather in two relations, as in Figure 3: one relation that relates employees to departments, and one relation that relates departments to managers. For more information, see [14] or [24].

An extended abstract of this paper appeared in the Proceedings of the 1980 ACM SIGACT Symposium on the Theory of Computing, Los Angeles, Calif. [23].

Author's address: IBM Research Laboratory K51/BM1, 5600 Cottle Road, San Jose, CA 95193.

EMP	DEPT	MGR
Hilbert	Math	Gauss
Pythagoras	Math	Gauss
Turing	Computer Science	von Neumann

FIGURE 1

FIGURE 2

EMP	DEPT	MGR
Hilbert	Math	Gauss
Pythagoras	Math	Gauss
Turing	Computer Science	von Neumann
Cauchy	Math	Euler

EMP	DEPT
Hilbert	Math
Pythagoras	Math
Turing	Computer Science

FIGURE 3

DEPT	MGR
Math	Gauss
Computer Science	von Neumann

More generally, Codd defined FDs $A_1 \cdots A_n \rightarrow B$, where each of $A_1, \ldots, A_n, B$ are names of columns of a relation. (We assume that no two distinct columns of the same relation have the same name.) This FD holds for a relation R if every pair of tuples of R that agree in each of the columns $A_1, \ldots, A_n$ also agree in the B column. It is easy to see [39] that FDs can be represented as sentences in first-order logic. Assume, for example, that we are dealing with a 4-ary relation, where the first, second, third, and fourth columns are called, respectively, A, B, C, and D. Then the FD $AB \rightarrow C$ is represented by the sentence

$$(\forall abc_1c_2d_1d_2)((Pabc_1d_1 \wedge Pabc_2d_2) \Rightarrow (c_1 = c_2)). \tag{1.1}$$

Here $(\forall abc_1c_2d_1d_2)$ is shorthand for $\forall a \forall b \forall c_1 \forall c_2 \forall d_1 \forall d_2$; that is, each variable is universally quantified. Unlike Nicolas [39], we have used individual variables rather than tuple variables. Incidentally, we think of P in (1.1) as a *relation symbol*,

which should not be confused with an *instance* (that is, a *relation*) R, for which (1.1) can hold.

The next dependency to be introduced was the *multivalued dependency* [21], or *MVD*. For the purposes of this paper it is convenient simply to discuss a single example rather than to give the general definition. Assume that we are dealing with ternary relations, where we refer to the three columns as A, B, and C. The MVD $A \twoheadrightarrow B$ is said to hold for such a relation if the following sentence is true (where P plays the role of the ternary relation):

$$(\forall ab_1b_2c_1c_2)((Pab_1c_1 \wedge Pab_2c_2) \Rightarrow Pab_1c_2. \tag{1.2}$$

In relational terminology the above sentence says that the ternary relation is the join of its projections onto AB and AC. The *projection* of a ternary relation R onto AB is $\{(a, b) : \exists cRabc\}$. The *join* of R_1 and R_2, where R_1 is a relation whose column names are A and B, and where R_2 is a relation whose column names are A and C, is $\{(a, b, c) : R_1(a, b) \text{ and } R_2(a, c)\}$.

Embedded dependencies were introduced [21] as dependencies that hold in a projection of a relation (although, as we shall see, they are now defined a little more generally). For example, assume that we are dealing with 4-ary relations, where we call the four columns $ABCD$. We say that such a 4-ary relation R obeys the *embedded MVD* (or *EMVD*) $A \twoheadrightarrow B \mid C$ if the projection of R onto ABC obeys the MVD $A \twoheadrightarrow B$. Thus the EMVD $A \twoheadrightarrow B \mid C$ can be written

$$(\forall ab_1b_2c_1c_2d_1d_2)((Pab_1c_1d_1 \wedge Pab_2c_2d_2) \Rightarrow \exists d_3 Pab_1c_2d_3). \tag{1.3}$$

In the last few years a number of generalizations of these dependencies have appeared: Nicolas' *mutual dependencies* [39], which say that a relation is the join of three of its projections; Mendelzon and Maier's *generalized mutual dependencies* [38]; Rissanen's [44] and Aho et al.'s [1] *join dependencies*, which generalize further to an arbitrary number of projections; Paradaens' *transitive dependencies* [41], which generalize both FDs and MVDs; Ginsburg and Zaiddan's *implied dependencies* [28], which generalize FDs; Sagiv and Walecka's *subset dependencies* [47], which generalize embedded MVDs; Sadri and Ullman's *template dependencies* [46], which generalize embedded join dependencies; and Parker and Parsaye-Ghomi's *generalized transitive dependencies* [43], which generalize transitive dependencies. We remark that the last three kinds of dependencies mentioned were introduced to deal with the issue of a complete axiomatization: subset dependencies were introduced to show the difficulty of completely axiomatizing embedded multivalued dependencies, while template dependencies and generalized transitive dependencies were introduced to provide a class of dependencies that include join dependencies and can be completely axiomatized.

The purpose of this paper is to help bring order to the chaos by presenting certain mathematical properties shared by all of these dependencies. The "right" definition of "dependency" might be those sentences that have certain properties (including, possibly, "faithfulness" and "domain independence," which are among the concepts discussed in this paper, and possibly also including the property that these sentences are true about empty relations). Each dependency of one of the types listed above is equivalent to a finite set of our implicational (or embedded implicational) dependencies, which we define soon. We note that Yannakakis and Papadimitriou [59] have, independently of the author, defined "algebraic dependencies," which, on the surface, look very different from our embedded implicational dependencies. Somewhat surprisingly, the class of algebraic dependencies and the class of embedded implicational dependencies turn out to be identical [59]. This is evidence for the naturalness

of the class. Yannakakis and Papadimitriou present a complete axiomatization. Beeri and Vardi [10] have defined *tuple-generating dependencies* and *equality-generating dependencies*, which, when they are restricted to be typed, together comprise our embedded implicational dependencies. (Beeri and Vardi have defined both typed and untyped versions; the typed version they call *many-sorted*.) Paradaens and Janssens [42] have defined *general dependencies*, which are implicational (but not embedded implicational) dependencies. Also, Grant and Jacobs [29] have defined *generalized dependency constraints*, which are untyped and interrelational versions of our implicational dependencies.

We begin with a few preliminary concepts. Let P be a relation symbol that represents the relation of interest. (When we deal with interrelational constraints, which we shall do later, then we shall, of course, need several relation symbols. For now we assume that we are dealing with only a single relation at a time.) We assume that we are given a set of *individual variables* (which represent entries in a relation). Assume that P represents a d-ary relation. Then the *atomic* formulas are those that are either of the form $Pz_1 \cdots z_d$ (where the z_i's are individual variables) or else of the form $x = y$ (where x and y are individual variables). We call atomic formulas $Pz_1 \cdots z_d$ *relational formulas*, and atomic formulas $x = y$ *equalities*. A *negation-atomic formula* is the negation of an atomic formula.

Formulas (which can involve Boolean connectives and quantifiers) and *sentences* (formulas with no free variables) are defined as usual (see any standard textbook in logic, e.g., [20] or [49].) We sometimes abbreviate $\forall x_1 \cdots \forall x_n \phi$, where each x_i is universally quantified, by $(\forall x_1 \cdots x_n)\phi$. Similarly, we sometimes abbreviate $\exists y_1 \cdots \exists y_r \phi$, where each y_i is existentially quantified, by $(\exists y_1 \cdots y_r)\phi$.

A formula is said to be *typed* if there are d disjoint classes, or types, of variables (where d is the arity, or degree, of relation symbol P and we say that a variable in the ith class is of *type* i), such that (a) if the relational formula $Pz_1 \cdots z_d$ appears in the formula, then z_i is of type i ($1 \leq i \leq d$), and (b) if the equality $x = y$ appears in the formula, then x and y have the same type.

In a typed formula no individual variable can represent an entry in two distinct columns. Thus, if Pxy appears in a typed formula (where x and y are individual variables), then Pzx cannot also appear, since if it did, then x would represent an entry in both the first and second columns.

An *implicational dependency* (or *ID*), is a typed sentence of the form

$$(\forall x_1 \cdots x_m)((A_1 \wedge \cdots \wedge A_n) \Rightarrow B), \tag{1.4}$$

where each A_i is a relational formula, B is atomic (either a relational formula or an equality), and each of the individual variables $x_1, \ldots, x_m$ that appear in at least one of $A_1, \ldots, A_n$ or B is universally quantified. We assume also that each variable (each of the x_j's) appears in at least one of the A_i's. In particular, $n \geq 1$, that is, there is at least one A_i. We also make similar assumptions when we define embedded implicational dependencies, so that, in particular, each implicational and embedded implicational dependency automatically holds for an "empty" relation with no tuples. Furthermore, our assumptions guarantee that we can tell if an implicational (or embedded implicational) dependency holds for a relation by simply considering the collection of tuples of the relation and ignoring the underlying "domains" (defined later) of attributes. We call this latter property *domain independence*. It is possible to define domain independence not only for *sentences*, but also for *formulas*. This class of domain-independent formulas is equivalent to Kuhns' [36] class of *definite* formulas. Unfortunately, the class of domain-independent formulas (and of domain-independent sentences) is not recursive [19, 55]. For this reason, various authors have

defined syntactically defined subclasses of domain-independent formulas. These include Codd's *range-separable* formulas [15], Nicolas' *range-restricted* formulas [40], Cooper's *permissible* formulas [16], Ullman's *safe* formulas [54], and Demolombe's *evaluable* formulas [17]. Since each of these classes is syntactically defined, each is recursive, unlike the full class of domain-independent formulas. Each of these classes is a *proper* subset of the class of domain-independent formulas, since each of these classes is recursive, while the class of all domain-independent formulas is not. Demolombe and Nicolas [18] show that each of Ullman's safe formulas is a domain-independent formula, and, conversely, that for each domain-independent formula there is an equivalent safe formula.

In line with dependency tradition, we may refer to IDs as *full* dependencies and EIDs that are not IDs as *strictly embedded* (or *strictly partial*) dependencies. There are two kinds of IDs, depending on whether the right-hand side of the implication is a relational formula or an equality. IDs in which the right-hand side is a relational formula are precisely the *full template dependencies* of Sadri and Ullman [46], and the *full tuple-generating dependencies* of Beeri and Vardi [10]. We may refer to these full tuple-generating dependencies as *full TGDs*, or *FTGDs*. IDs in which the right-hand side is an equality are Beeri and Vardi's *equality-generating dependencies*, or *EGDs* [10]. (In a preliminary version [23] of this paper we called EGDs *extended functional dependencies*, or *XFDs*).

Note that the FD (1.1) and the MVD (1.2) are each an ID (where the FD is an EGD and the MVD is an FTGD). We note that there is a possible confusion in treating an FD both in the usual manner (as a sentence written in the form $AB \rightarrow C$) and also as a sentence written in the form (1.1). For example, the FD $AB \rightarrow C$, when referring to a 4-ary relation, is written as in (1.1), whereas the FD $AB \rightarrow C$, when referring to a 3-ary relation, is written as

$$(\forall abc_1c_2)((Pabc_1 \wedge Pabc_2) \Rightarrow (c_1 = c_2)). \tag{1.5}$$

If we speak about an FD as being written in the usual $AB \rightarrow C$ notation, then we can speak (as we shall in Section 6) about the *same* FD $AB \rightarrow C$ holding in a relation and its projection, whereas this would not make sense if we think in the latter terms (since (1.1) and (1.5) are certainly not the same syntactically). In the usual $AB \rightarrow C$ notation, the role of attributes is emphasized and the arity of the relations is not, whereas in the notation of (1.1), the reverse is true. We hope that this double manner of thinking about FDs does not cause confusion.

Following Slagle and Koniver [51], let us call an unquantified formula of the form

$$((A_1 \wedge \cdots \wedge A_n) \Rightarrow B), \tag{1.6}$$

where $n \geq 1$ (i.e., where there is at least one A_i), an *implication*. A *Horn clause* [32] is the disjunction of atomic and negation-atomic formulas, where at most one is atomic. The implication (1.6) is equivalent to the Horn clause

$$(\neg A_1 \vee \cdots \vee \neg A_n \vee B),$$

which has *exactly* one atomic formula and at least one negation-atomic formula.

An *embedded implicational dependency* (or *EID*) is a typed sentence of the form

$$(\forall x_1 \cdots x_m)((A_1 \wedge \cdots \wedge A_n) \Rightarrow (\exists y_1 \cdots y_r)(B_1 \wedge \cdots \wedge B_s)), \tag{1.7}$$

where each A_i is a relational formula and each B_i is atomic (either a relational formula or an equality). We assume also that each of the x_j's appears in at least one of the A_i's and that $n \geq 1$, that is, that there is at least one A_i. We assume that $r \geq 0$ (if $r = 0$, then there are no existential quantifiers) and that $s \geq 1$ (i.e., there must be

at least one B_i). Because of all these assumptions, each EID is obeyed by an empty relation with no tuples. Note that each ID is an EID (in which there are no existential quantifiers).

Remark. We could modify our definition of IDs so that, like EIDs, they can have more than one atomic formula on the right-hand side of the implication. That is, by analogy with (1.7) we could allow IDs to be of form

$$(\forall x_1 \cdots x_m)((A_1 \wedge \cdots \wedge A_n) \Rightarrow (B_1 \wedge \cdots \wedge B_s)). \tag{1.8}$$

However, we do not do so, since (1.8) is equivalent to the set of s IDs

$$(\forall x_1 \cdots x_m)((A_1 \wedge \cdots \wedge A_n) \Rightarrow B_i)$$

for $i = 1, \ldots, s$, and we are interested in what can be said with *sets* of dependencies, not just with *single* dependencies. The analogous equivalence does *not* hold for EIDs (1.7).

There are many open questions about embedded dependencies. For example, it is not even known whether the decision problem for EMVDs is decidable, that is, whether, given a set Σ of EMVDs and a single EMVD σ, it is the case that Σ logically implies σ [46]. (However, we note that Vardi [57] and Gurevich and Lewis [31] have proven the undecidability of the decision problem for the more general class of template dependencies.) The existence of an Armstrong relation in the presence of EMVDs (which we shall prove) is itself a new result, for which the old proof techniques seem to be inadequate.

Sadri and Ullman's *template dependency* (or *TD*) is a special case of an EID in which there is only one atomic formula on the right-hand side of the implication and in which this atomic formula is a relational formula (i.e., $s = 1$ in (1.7), and also B_1 in (1.7) is a relational formula). We note that Fagin et al. [26] develop a number of techniques, counterexamples, and results about TDs.

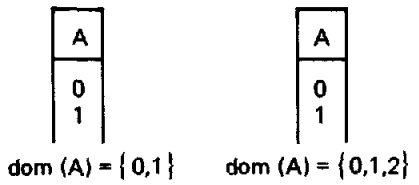
In Section 2 we introduce the concept of "faithfulness" (with respect to direct product) and show that IDs and EIDs are faithful, whereas slight variations are not necessarily faithful. In Section 3 we discuss "Armstrong relations," which were known to exist in certain special cases (such as when the only sentences of interest were functional, multivalued, and join dependencies). We show that Armstrong relations exist even in the presence of EIDs. This is perhaps the most interesting result technically in this paper. In Section 4 we discuss finite Armstrong relations. An existence theorem and a counterexample to an extension of the theorem are presented. In Section 5 we present some more counterexamples about the existence of Armstrong relations. In Section 6 we discuss projections of classes of relations. Although Ginsburg and Zaiddan [28] showed that projections of FD classes are not necessarily FD classes, it turns out that projections of FD classes (and, even more, of ID classes) are ID classes. In Section 7 we discuss certain extensions of our results (that, in particular, allow some interrelational and nontyped dependencies).

2. *Faithfulness with Respect to Direct Product*

In this section we define the direct product operator, and we introduce a concept of *faithfulness* (with respect to direct product). A sentence is faithful when it holds for each member of a nonempty family of nonempty relations if and only if it holds for their direct product. We show that our class of EIDs (embedded implicational dependencies) is faithful. Furthermore, we show that under slight modifications of our definition of EID, we can obtain a sentence that is not faithful.

Let U be a finite set of distinct symbols, called *attributes* (or *column names*). A

FIGURE 4



domain mapping is a mapping that associates to each attribute A in U a set dom(A), called the *domain* of A. In the spirit of Armstrong [2] and of Aho et al. [1], we define a *tuple* to be a function that maps each attribute A into a member of dom(A). We call the value associated with the attribute A the *A entry* of the tuple. If the attributes are, say, A, B, and C, then for notational convenience we sometimes write (a, b, c) to represent the tuple, where the A entry is a, etc. A *d-ary relation* is a domain mapping (over d attributes), along with a set of tuples (involving the same attributes). We say that the *arity* of the relation is d. This definition of a relation, which is slightly different from the usual definition in that it explicitly considers the role of domains, is usually necessary in the presence of quantifiers. Thus, to decide whether a sentence holds for a given relation, the domains tell us over what set of x's a "$\forall x$" ranges. For example, the first relation in Figure 4 obeys the sentence $\forall x Px$, and the second does not, even though both relations have the same set of tuples. EIDs have been defined in such a way that it is possible to determine whether they hold for a given relation by considering only the tuples, and not the underlying domains (this property we have called *domain independence*).

Our definition of "relation" is in the spirit of Tarski's definition [53] of "model," in that domains are explictly considered. Our definition is analogous to considering a graph as a set of nodes, along with a set of edges, whereas the usual definition (of a relation as simply a set of tuples) is analogous to considering a graph as simply a set of edges.

We say that a relation is *empty* if its set of tuples is the empty set. This is *not* the same as saying that one or more of the domains is empty. In fact, it is traditional to require that none of the domains be empty in any relation, including an empty relation.

Let $\langle R_i : i \in I \rangle$ be a (finite or infinite) family of relations, each with the same set U of attributes. (Note: Throughout this paper we assume for convenience that whenever we speak of a *family* $\langle R_i : i \in I \rangle$, we always mean a *nonempty family*, i.e., we assume that the index set I is nonempty.) We now define the *direct product* $\otimes \langle R_i : i \in I \rangle$. The direct product has the same set U of attributes as does each of the R_i's. In particular, the direct product maps a family of d-ary relations into a d-ary relation (with the same arity d as each of the R_i's). For notational convenience let us assume that U contains precisely three attributes ABC. (It is obvious how to generalize the definition from this special case.) Let us denote the domain dom(A) in R_i by D_i, for each i. Note that we make no restrictions on these domains, such as that the A domains of distinct relations be the same or distinct, or that the domains be finite or infinite. The domain dom(A) in the direct product is defined to be the Cartesian product $\times \langle D_i : i \in I \rangle$. A similar statement holds for dom(B) and dom(C). The tuple $(\langle a_i : i \in I \rangle, \langle b_i : i \in I \rangle, \langle c_i : i \in I \rangle)$ is a tuple of the direct product if and only if (a_i, b_i, c_i) is a tuple of R_i, for every i. For example, the direct product of the first two relations in Figure 5 is the third relation in Figure 5. It is sometimes convenient to refer to R_i as the *ith component of* $\otimes \langle R_i : i \in I \rangle$ and to a_i as the *ith component of* $\langle a_i : i \in I \rangle$.

R_1

A	B	C
a_1	b_1	c_1
a_1	b_1	c'_1

R_2

A	B	C
a_2	b_2	c_2
a_2	b'_2	c_2

dom (A) = $\{a_1, a'_1\}$
dom (B) = $\{b_1\}$
dom (C) = $\{c_1, c'_1\}$

dom (A) = $\{a_2, a'_2\}$
dom (B) = $\{b_2, b'_2\}$
dom (C) = $\{c_2\}$

$R_1 \otimes R_2$

A	B	C
$\langle a_1, a_2\rangle$	$\langle b_1, b_2\rangle$	$\langle c_1, c_2\rangle$
$\langle a_1, a_2\rangle$	$\langle b_1, b'_2\rangle$	$\langle c_1, c_2\rangle$
$\langle a_1, a_2\rangle$	$\langle b_1, b_2\rangle$	$\langle c'_1, c_2\rangle$
$\langle a_1, a_2\rangle$	$\langle b_1, b'_2\rangle$	$\langle c'_1, c_2\rangle$

dom (A) = $\{\langle a_1, a_2\rangle, \langle a_1, a'_2\rangle, \langle a'_1, a_2\rangle, \langle a'_1, a'_2\rangle\}$
dom (B) = $\{\langle b_1, b_2\rangle, \langle b_1, b'_2\rangle\}$
dom (C) = $\{\langle c_1, c_2\rangle, \langle c'_1, c_2\rangle\}$

FIGURE 5

We have made no restrictions in our definitions as to whether a relation must be finite (i.e., have a finite number of tuples) or may be infinite. In particular, it is easy to see that the direct product of an infinite family of relations, each of which contains at least two tuples, is not only infinite but even uncountable. At various points in this paper we explicitly focus our attention on *finite* relations.

We sometimes write $\otimes\langle R_1, R_2\rangle$ as $R_1 \otimes R_2$; similarly, we may write $\otimes\langle R_1, \ldots, R_t\rangle$ as $R_1 \otimes \cdots \otimes R_t$.

We define a *database* (which is, intuitively, a labeled collection of relations) in Section 7. We also define the direct product of databases, which is simply the direct product taken relationwise. Until Section 7 we mainly discuss single relations, rather than databases.

Let σ be a sentence of first-order logic. For now we assume, in order to simplify our definitions, that σ is unirelational (not interrelational), that is, that it is a sentence about a relation and not about a multirelation database. We say that σ is *faithful* (with respect to direct products) if whenever $\langle R_i : i \in I\rangle$ is a family of nonempty relations, then σ holds for $\otimes\langle R_i : i \in I\rangle$ if and only if σ holds for every R_i.

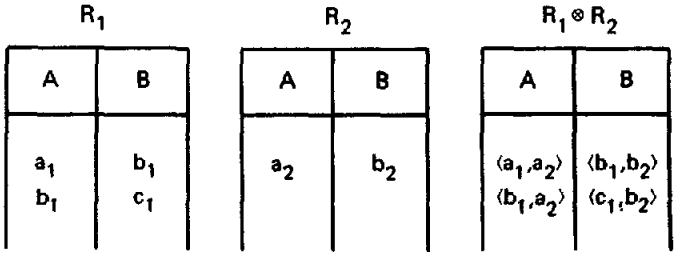
The main theorem of this section is as follows.

THEOREM 2.1. *Every EID (and thus every ID) is faithful.*

Before we prove Theorem 2.1 it is helpful to introduce some more concepts, to give a few examples, and to state some other results.

We say that a sentence σ is *upward faithful* (with respect to direct products) if whenever $\langle R_i : i \in I\rangle$ is a family of nonempty relations such that σ holds for every R_i,

FIGURE 6



then σ holds for $\otimes\langle R_i : i \in I\rangle$. We say that a sentence σ is *downward faithful* (with respect to direct products) if whenever $\langle R_i : i \in I\rangle$ is a family of nonempty relations such that σ holds for $\otimes\langle R_i : i \in I\rangle$, then σ holds for every R_i. Clearly, σ is faithful if and only if it is both upward and downward faithful. We remark that it is not necessary to assume that the components R_i are nonempty in the definition of upward faithful, but the assumption is important to us in the downward faithful case. We return to this point at the end of this section.

Example 2.2. The "degenerate MVD" [3, 48],

$$(\forall x y_1 y_2 z_1 z_2)((Pxy_1z_1 \wedge Pxy_2z_2) \Rightarrow ((y_1 = y_2) \vee (z_1 = z_2))) \tag{2.1}$$

is not upward faithful (although it is downward faithful, by Theorem 2.5 below). Thus, relations R_1 and R_2 in Figure 5 (where $b_1 \neq b_1'$, etc.) both obey this sentence, but the direct product $R_1 \otimes R_2$ does not (as we see by looking at the first and fourth tuples in the direct product). This sentence differs from an ID in that the right-hand side of the implication is not an atomic formula, but the disjunction of atomic formulas. □

This example brings up a few comments about the role of domains. In Figure 5 we have noted the domains of each of the attributes, although for this sentence (and, in addition, for all EIDs) it is possible to determine the truth of the sentence for a particular relation by considering only the tuples in the relation and ignoring the underlying domains. The domains are explicitly noted because we make use of this example later in a context where the role of the domains *will* be important (it is important in our later example that only one of the two possible A values appears in relation R_1).

We also note that for convenience we have allowed some of the domains (such as dom(B) in R_1) to contain only one element. One-element domains are sometimes considered undesirable (see [24]); however, in none of our examples with one-element domains is this feature in any way essential; it is simply convenient.

Example 2.3. The sentence

$$(\forall xyz)((Pxy \wedge Pyz) \Rightarrow Pxz)$$

is not downward faithful (although it is upward faithful, by Theorem 2.4 below). Thus $R_1 \otimes R_2$ in Figure 6 obeys this sentence, although R_1 does not. This sentence says that the relation is transitive. It differs from an ID in that it is not typed. □

In this example we have not bothered to note explicitly the underlying domains, since in this case the domains are not needed to determine truth or falsity of the given sentence for the given relations.

Horn's motivation [32] for introducing Horn clauses is the following theorem.

THEOREM 2.4 [32; 49, pp. 94–95]. *Let σ be a sentence of the form*

$$Q_1x_1 \cdots Q_mx_m(M_1 \wedge \cdots \wedge M_s),$$

where each Q_i is a quantifier ($\forall$ or $\exists$) and each M_i is a Horn clause. Then σ is upward faithful.

Theorem 2.4 does *not* require that σ be typed or that it not be interrelational. We make use of this fact in Section 7.

As we shall see, every EID is equivalent to a sentence of the kind mentioned in Theorem 2.4; thus it follows from Theorem 2.4 that if σ is an EID, then σ is upward faithful, which proves part of Theorem 2.1. As we shall see, it is *not* true that if σ is as in Theorem 2.4, then σ is necessarily downward faithful.

After giving a few definitions, we state two theorems that give sufficient conditions for a sentence to be downward faithful. We present the proofs of these theorems later in this section.

The class of *quantifier-free formulas* is defined as usual (see [20] or [49]). The class of *positive quantifier-free formulas* is the smallest class such that (a) it contains all atomic formulas, and (b) if it contains ϕ_1 and ϕ_2, then it also contains $\phi_1 \wedge \phi_2$ and $\phi_1 \vee \phi_2$.

THEOREM 2.5. *Let σ be a universal sentence, that is, a sentence of the form $(\forall x_1 \cdots x_m)\phi$, where ϕ is a quantifier-free formula. Assume further that σ is typed and unirelational. Then σ is downward faithful.*

THEOREM 2.6. *Let σ be a sentence of the form*

$$(\forall x_1 \cdots x_m)(\phi \Rightarrow (\exists y_1 \cdots y_r)\gamma),$$

where ϕ is a quantifier-free formula and γ is a positive quantifier-free formula. Assume further that σ is typed and unirelational. Then σ is downward faithful.

Remark. In the theorem, instead of assuming that σ is typed and unirelational, it is possible to make the weaker assumption that the left-hand side ϕ is typed and unirelational. We come back to this point in Section 7.

We now show that Theorem 2.1 follows from Theorems 2.4–2.6.

PROOF OF THEOREM 2.1. We must show that each EID is faithful. We first show that they are upward faithful.

Let σ be

$$(\forall x_1 \cdots x_m)((A_1 \wedge \cdots \wedge A_n) \Rightarrow (\exists y_1 \cdots y_r)(B_1 \wedge \cdots \wedge B_s)), \tag{2.2}$$

where each A_i and B_i is atomic. Then σ is equivalent to the sentence

$$(\forall x_1 \cdots x_m)(\exists y_1 \cdots y_r)((A_1 \wedge \cdots \wedge A_n) \Rightarrow (B_1 \wedge \cdots \wedge B_s)). \tag{2.3}$$

Sentence (2.3) is equivalent to the sentence

$$(\forall x_1 \cdots x_m)(\exists y_1 \cdots y_r)(M_1 \wedge \cdots \wedge M_s), \tag{2.4}$$

where M_i is

$$A_1 \wedge \cdots \wedge A_n \Rightarrow B_i$$

for each i. Thus, by Theorem 2.4, we know that σ is upward faithful.

So, EIDs are upward faithful. Also, each EID is a sentence of the kind described in Theorem 2.6, so each EID is downward faithful. Thus EIDs are faithful, which

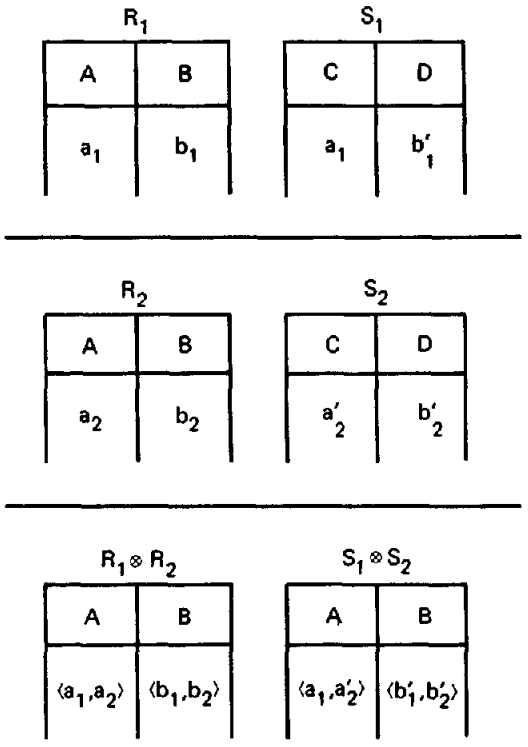


FIGURE 7

was to be shown. We note that the fact that IDs are downward faithful follows from either Theorem 2.5 or Theorem 2.6. □

Yannakakis and Papadimitriou [59] give an elegant proof of Theorem 2.1, by first showing that embedded implicational dependencies are equivalent to their algebraic dependencies (which are built out of projections and joins), and then showing that the direct product commutes with projections and joins. For details, see [59].

In Examples 2.2 and 2.3 we demonstrated some sentences that are only "slightly different" from implicational dependencies but that are not faithful. We now give some more examples.

Example 2.7. The sentence

$$(\forall x y_1 y_2)((Pxy_1 \wedge Qxy_2) \Rightarrow (y_1 = y_2))$$

is not downward faithful (although it is upward faithful, by Theorem 2.4). Thus the first database of Figure 7 (containing R_1 and S_1) violates this sentence, although the direct product database (at the bottom of Figure 7) obeys it. As in Example 2.3, we have not bothered to note explicitly the underlying domains, since in this case the domains are not needed to determine truth or falsity of the given sentence for the given databases. This sentence differs from an ID in that it is "interrelational." (Technically, we have not yet defined the direct product when each member of the direct product is a "database" consisting of several relations. We simply take direct products relationwise.) □

Example 2.8. The sentence

$$(\forall y_1 y_2)\exists x((Pxy_1 \wedge Pxy_2) \Rightarrow (y_1 = y_2))$$

is not downward faithful (although it is upward faithful, by Theorem 2.4). Thus,

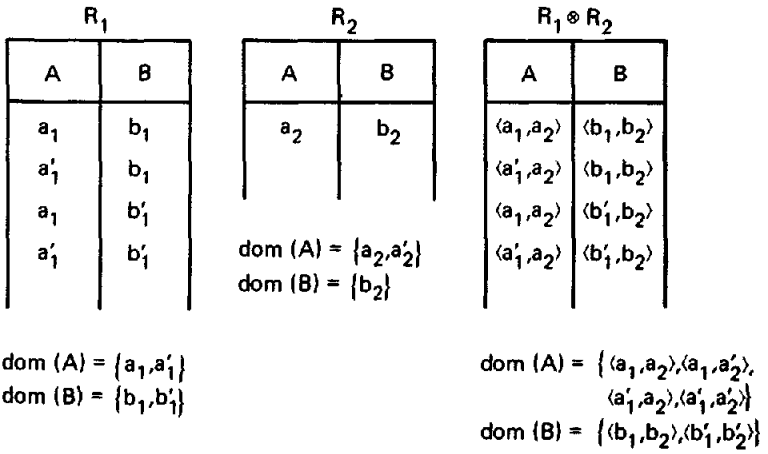


FIGURE 8

consider the relations in Figure 8. In this case it is important for us to consider explicitly the domains of the attributes. In Figure 8, relation R_1 does not obey the sentence, whereas the direct product $R_1 \otimes R_2$ does. (The reason that the direct product does is, intuitively, that when y_1 is $\langle b_1, b_2\rangle$ and y_2 is $\langle b_1', b_2\rangle$, then we can take x to be $\langle a_1, a_2'\rangle$, which is in dom(A) in the direct product, although it does not appear in the A column of $R_1 \otimes R_2$.) This sentence differs from an ID in that it is not universally quantified but instead is a "$\forall\exists$" sentence. Note, incidentally, that EIDs (1.7) are special $\forall\exists$ sentences that *are* faithful. □

Example 2.9. The sentence

$$\exists x(\forall y_1 y_2)((Pxy_1 \wedge Pxy_2) \Rightarrow (y_1 = y_2))$$

is not downward faithful (although it is upward faithful, by Theorem 2.4). Thus relation R_1 in Figure 8 does not obey this sentence, although the direct product $R_1 \otimes R_2$ does (where, intuitively, x is taken to be $\langle a_1, a_2'\rangle$). This sentence differs from an ID in that it is not universally quantified but instead is a "$\exists\forall$" sentence. □

Example 2.10. What about existential sentences? Let us first consider sentences

$$(\exists x_1 \cdots x_m)((A_1 \wedge \cdots \wedge A_n) \Rightarrow B),$$

which are just like IDs except that the variables are existentially rather than universally quantified. It is not hard to see that these sentences are all tautologies and so, of course, are faithful. However, there are existential sentences that are not faithful. For example, the sentence

$$(\exists x_1 x_2 x_3)\neg Px_1x_2x_3 \tag{2.5}$$

is not downward faithful (although it is upward faithful, by Theorem 2.4). Thus relation R_1 in Figure 9 violates this sentence, although the direct product $R_1 \otimes R_2$ obeys it. □

Example 2.11. As our final counterexample we exhibit a sentence that is neither upward nor downward faithful. Our sentence is taken to be the conjunction of the sentence (2.1) in Example 2.2 and the sentence (2.5) in Example 2.10. This new sentence is not upward faithful, since the relations R_1 and R_2 in Figure 5 both obey it, whereas their direct product does not. The sentence is not downward faithful, since the relation R_1 in Figure 9 does not obey it, whereas the direct product $R_1 \otimes R_2$ in Figure 9 does. □

R_1

A	B	C
a_1	b_1	c_1
a'_1	b_1	c_1

dom (A) = $\{a_1, a'_1\}$
dom (B) = $\{b_1\}$
dom (C) = $\{c_1\}$

R_2

A	B	C
a_2	b_2	c_2

dom (A) = $\{a_2, a'_2\}$
dom (B) = $\{b_2\}$
dom (C) = $\{c_2\}$

$R_1 \otimes R_2$

A	B	C
$\langle a_1, a_2 \rangle$	$\langle b_1, b_2 \rangle$	$\langle c_1, c_2 \rangle$
$\langle a'_1, a_2 \rangle$	$\langle b_1, b_2 \rangle$	$\langle c_1, c_2 \rangle$

dom (A) = $\{\langle a_1, a_2 \rangle, \langle a_1, a'_2 \rangle, \langle a'_1, a_2 \rangle, \langle a'_1, a'_2 \rangle\}$
dom (B) = $\{\langle b_1, b_2 \rangle\}$
dom (C) = $\{\langle c_1, c_2 \rangle\}$

FIGURE 9

Before we prove Theorems 2.5 and 2.6, we establish a few conventions. If p is a member of the jth domain of a relation R and x is a variable of type j, then we say that p is *substitutable* for x (with respect to R). If ψ is a quantifier-free formula and $z_1, \ldots, z_k$ is a fixed ordering of a collection of variables, where the set of variables appearing in ψ is a subset (possibly proper) of $\{z_1, \ldots, z_k\}$, then it is sometimes convenient for us to write ψ as $\psi(z_1, \ldots, z_k)$. Further, if $r_1, \ldots, r_k$ are elements, where r_i is substitutable for z_i for each i, then we may write $\psi(r_1, \ldots, r_k)$ to mean the result of substituting r_i for z_i, for each i.

PROOF OF THEOREM 2.5. Let σ be the universal sentence $(\forall x_1 \cdots x_m)\phi$, where ϕ is a quantifier-free formula. Assume further that σ is typed and unirelational. We now show that σ is downward faithful. Assume that σ holds for $\otimes\langle R_i : i \in I \rangle$, where each R_i is nonempty. We must show that σ holds for each R_i. Assume not; by relabeling, if necessary, assume that R_1 is an R_i for which σ fails. Thus there are $q_1, \ldots, q_m$ such that $\neg\phi(q_1, \ldots, q_m)$ holds for R_1. Let us denote $\otimes\langle R_i : i \in I \rangle$ by R.

By assumption, each R_i is nonempty. Let us select a tuple t_i from R_i for each i except $i = 1$. Let $t_{i,j}$ be the jth member of the tuple t_i; thus t_i equals $(t_{i,1}, \ldots, t_{i,d})$, where d is the arity of each of the R_i's. Let $q_1, \ldots, q_m$ be as above. We now define new points $Q_1, \ldots, Q_m$, each of which is in some domain of R. Assume that x_i, the variable in σ that corresponds to q_i, is of type p. Define Q_i by letting its kth component be q_i, if $k = 1$, and $t_{k,p}$ otherwise. So, Q_i is a member of the pth domain of the direct product. We now show, by induction on the structure of quantifier-free formulas, that for every quantifier-free formula $\psi(x_1, \ldots, x_m)$,

$$\psi(Q_1, \ldots, Q_m) \text{ holds for } R \quad \text{iff} \quad \psi(q_1, \ldots, q_m) \text{ holds for } R_1. \tag{2.6}$$

We first show (2.6) in the case where ψ is atomic.

Case 1. ψ is an equality $x = y$. Since the only variables that can appear in ψ are $x_1, \ldots, x_m$, let us assume that ψ is the formula $x_i = x_j$. To show (2.6), we must show that

$$Q_i = Q_j \text{ holds for } R \quad \text{iff} \quad q_i = q_j \text{ holds for } R_1. \tag{2.7}$$

Now x_i and x_j are of the same type; let us say that they are of type p. If $k \neq 1$, then the kth components of Q_i and of Q_j are the same, namely $t_{k,p}$. Hence Q_i and Q_j are the same if and only if their first components (the kth component for $k = 1$) are the same. But the first component of Q_i is q_i, and the first component of Q_j is q_j. Thus (2.7) holds.

Case 2. ψ is a relational formula $Pz_1 \cdots z_d$. Since the only variables that can appear in ψ are $x_1, \ldots, x_m$, let us assume that ψ is the formula $Px_{i_1} \cdots x_{i_d}$. To show (2.6), we must show that

$$(Q_{i_1}, \ldots, Q_{i_d}) \quad \text{is a tuple of} \quad R \qquad \text{iff} \qquad (q_{i_1}, \ldots, q_{i_d}) \quad \text{is a tuple of} \quad R_1. \tag{2.8}$$

Now x_{i_p} is of type p, and so the kth component of Q_{i_p} is q_{i_p}, if $k = 1$, and $t_{k,p}$ otherwise. By definition of the direct product, $(Q_{i_1}, \ldots, Q_{i_d})$ is a tuple of R if and only if

$$\begin{array}{ll} (q_{i_1}, \ldots, q_{i_d}) & \text{is a tuple of} \quad R_1, \\ (t_{2,1}, \ldots, t_{2,d}) & \text{is a tuple of} \quad R_2, \\ (t_{3,1}, \ldots, t_{3,d}) & \text{is a tuple of} \quad R_3. \\ \qquad\vdots & \end{array} \tag{2.9}$$

Now, the second, third, ... statements in (2.9) hold by definition of the tuples $t_i = (t_{i,1}, \ldots, t_{i,d})$. Thus (2.9) holds if and only if the first statement in (2.9) holds. Thus (2.8) holds, which was to be shown.

We have shown that (2.6) holds if ψ is atomic. It is straightforward to verify that if (2.6) holds when ψ is ψ_1 and when ψ is ψ_2, then it holds when ψ is $\psi_1 \wedge \psi_2$, when ψ is $\psi_1 \vee \psi_2$, and when ψ is $\neg\psi_1$. For example, let us demonstrate the "$\psi_1 \vee \psi_2$" case. Then $\psi_1 \vee \psi_2$ holds for the direct product if and only if either ψ_1 or ψ_2 holds for the direct product, which, by the induction assumption, happens if and only if either ψ_1 or ψ_2 holds for R_1, which happens if and only if $\psi_1 \vee \psi_2$ holds for R_1.

We have now proved (2.6) whenever ψ is quantifier-free. But by assumption, $\neg\phi(q_1, \ldots, q_m)$ holds for R_1. So, by (2.6) we know that $\neg\phi(Q_1, \ldots, Q_m)$ holds for R. But this contradicts our assumption that $(\forall x_1 \cdots x_m)\phi$ holds for R. □

Proof of Theorem 2.6. Let σ be the sentence

$$(\forall x_1 \cdots x_m)(\phi \Rightarrow (\exists y_1 \cdots y_r)\gamma), \tag{2.10}$$

where ϕ is a quantifier-free formula and γ is a positive quantifier-free formula. Assume further that σ is typed and unirelational. We now show that σ is downward faithful. As in the previous proof, let us denote $\otimes\langle R_i : i \in I\rangle$ by R.

Assume that σ holds for R, where each R_i is nonempty. We must show that σ holds for each R_i. Assume not; as before, by relabeling if necessary, assume that σ fails for R_1. Thus there are $q_1, \ldots, q_m$ such that $\phi(q_1, \ldots, q_m)$ holds for R_1 and there are no points $s_1, \ldots, s_r$, for which

$$\gamma(q_1, \ldots, q_m, s_1, \ldots, s_r) \tag{2.11}$$

holds for R_1.

By assumption, each R_i is nonempty. Let us select a tuple t_i from R_i for each i except $i = 1$. Let $t_{i,j}$ be the jth member of the tuple t_i; thus t_i equals $(t_{i,1}, \ldots, t_{i,d})$, where d is the arity of each of the R_i's. Let $q_1, \ldots, q_m$ be as above. As in the proof of Theorem 2.3, we now define new points $Q_1, \ldots, Q_m$, each of which is in some domain of R. Assume that x_i, the variable in σ that corresponds to q_i, is of type p. Define Q_i by letting its kth component be q_i, if $k = 1$, and $t_{k,p}$ otherwise. So, Q_i is a member of the pth domain of the direct product.

By the same proof as that given in the proof of Theorem 2.5 (except using ϕ instead of $\neg\phi$), it follows that $\phi(Q_1, \ldots, Q_m)$ holds for R. By assumption, sentence (2.10) is

true about R. If we write γ as $\gamma(x_1, \ldots, x_m, y_1, \ldots, y_r)$, then it follows that there are $S_1, \ldots, S_r$ such that

$$\gamma(Q_1, \ldots, Q_r, S_1, \ldots, S_r) \tag{2.12}$$

holds for R. Denote the first component (i.e., the component corresponding to R_1) of S_i by s_i $(1 \leq i \leq r)$. We already know that the first component of Q_i is q_i $(1 \leq i \leq r)$. We now show that for each positive, quantifier-free formula ψ,

$$\begin{aligned} &\psi(q_1, \ldots, q_r, s_1, \ldots, s_r) \quad \text{holds for} \quad R_1 \\ &\quad \text{if} \quad \psi(Q_1, \ldots, Q_r, S_1, \ldots, S_r) \quad \text{holds for} \quad R. \end{aligned} \tag{2.13}$$

Notice that (2.13) is an "if" statement, not an "if and only if."

The proof of (2.13) is by induction on positive, quantifier-free formulas. First we show (2.13) in case ψ is an equality. If $Q_i = S_j$, then the first component of Q_i, namely q_i, must equal the first component of S_j, namely s_j. We have shown that if $Q_i = S_j$, then $q_i = s_j$. Similarly, if $Q_i = Q_j$, then $q_i = q_j$, and if $S_i = S_j$, then $s_i = s_j$. We just proved (2.13) in case ψ is an equality. Also, (2.13) holds if ψ is a relational formula (of the form $Pz_1 \cdots z_d$), by the definition of the direct product. We have shown that (2.13) holds if ψ is atomic. Finally, it is easy to verify that if (2.13) holds when ψ is ψ_1 and when ψ is ψ_2, then it holds when ψ is $\psi_1 \wedge \psi_2$ and when ψ is $\psi_1 \vee \psi_2$. Thus (2.13) holds for each positive, quantifier-free formula ψ. In particular, (2.13) holds when ψ is γ. So, since (2.12) holds for R, it follows from (2.13) that (2.11) holds for R_1. This is a contradiction. □

We close this section with some comments relating upward and downward faithfulness to concepts defined earlier in the literature. The following two definitions are standard (see, e.g., [13]). A sentence σ is *preserved under direct products* if whenever $\langle R_i : i \in I \rangle$ is a family of relations such that σ holds for every R_i, then σ holds for $\otimes\langle R_i : i \in I \rangle$. A sentence σ is *preserved under direct factors* if whenever $\langle R_i : i \in I \rangle$ is a family of relations such that σ holds for $\otimes\langle R_i : i \in I \rangle$, then σ holds for every R_i. We can easily verify that a sentence is upward faithful with respect to direct products if and only if it preserves direct products. This is because the only difference in the definitions of upward faithfulness and of being preserved under direct products involves whether or not the component relations are allowed to be empty, and because the direct product of relations, one of which is empty, is also empty. Horn stated his theorem (Theorem 2.4 above) in terms of preservation under direct product.

However, there is an important difference between downward faithfulness and being preserved under direct factors. Thus the restriction in the definition of downward faithfulness that we consider only nonempty relations is actually important. To see this, we first show that if a sentence σ is preserved under direct factors and is true about the empty relation, then σ is a tautology. For, if not, assume that σ is preserved under direct factors, is true about the empty relation, and is not a tautology. Let R_1 be the empty relation, and let R_2 be a relation for which σ fails. We know that σ holds for $R_1 \otimes R_2$, since $R_1 \otimes R_2$ is the empty relation. Since σ is preserved under direct factors, it follows that σ holds for R_2. This is a contradiction. Now let σ be a nontautologous EID. Since σ holds for the empty relation, it follows from what we just showed that σ is not preserved under direct factors. However, σ is downward faithful. So being preserved under direct factors and being downward faithful are not equivalent. Keisler [35] gave a complicated characterization of sentences that are preserved under direct factors. His class and our class of EIDs have in common only tautologies.

EMP	DEPT	MGR
Hilbert	Math	Gauss
Pythagoras	Math	Gauss
Turing	Computer Science	von Neumann
Einstein	Physics	Gauss

FIGURE 10

3. *Armstrong Relations*

In this section we show that a theorem due to Armstrong about FDs generalizes to EIDs. Further, we demonstrate a general equivalence that is useful in our context and also, we believe, in other contexts.

Let Σ be a set of sentences, and let σ be a single sentence. When we say that Σ *logically implies* σ or that σ *is a logical consequence* of Σ, we mean that whenever every sentence in Σ holds for a relation R, then σ also holds for R. That is, there is no "counterexample relation" or "witness" R such that every sentence in Σ holds for R but σ fails in R. We write $\Sigma \models \sigma$ to mean that Σ logically implies σ, and we write $\Sigma \not\models \sigma$ to mean that Σ does not logically imply σ. If Γ is a set of sentences, then we may write $\Sigma \models \Gamma$ to mean that $\Sigma \models \gamma$ for every γ in Γ. For example, $\{A \rightarrow B, B \rightarrow C\} \models A \rightarrow C$.

Let Σ be a set of FDs, and let Σ^* be the set of all FDs that are logical consequences of Σ. For each FD σ not in Σ^*, we know (by definition of $\models$) that there is a relation R_σ (a witness) such that R_σ obeys Σ but not σ. It follows from Armstrong's results [2] that there is a relation (a global witness) that can simultaneously serve the role of all of the R_σ's. That is, Armstrong showed that there is a relation that obeys Σ^* and *no other FDs.* We call such a relation an *Armstrong relation* for Σ. Actually, Armstrong did not *explicitly* state or prove the existence of what we call an Armstrong relation. Instead, he proved a result that implies both the completeness of a certain set of axioms about FDs (see [22]) and the existence of an Armstrong relation.

Let us consider an example. Let Σ be the set {EMP $\rightarrow$ DEPT, DEPT $\rightarrow$ MGR}, containing two FDs. Then Σ^* contains the FDs in Σ, along with, for example, the FD EMP $\rightarrow$ MGR. It is easy to verify (by considering all possible FDs involving only EMP, DEPT, and MGR) that the relation (call it R) in Figure 10 is an Armstrong relation for Σ, that is, that it obeys every FD in Σ^* and no others. At this point the reader is encouraged to examine relation R in Figure 10 before reading further.

The striking feature that the reader probably noticed almost immediately is that (in relation R) Gauss is the manager of two distinct departments (Math and Physics). Thus R does *not* obey the FD MGR $\rightarrow$ DEPT. This is as it should be, since R is an Armstrong relation for {EMP $\rightarrow$ DEPT, DEPT $\rightarrow$ MGR}, while the FD MGR $\rightarrow$ DEPT is not a logical consequence of these dependencies. We explain in the next two paragraphs why we asked the reader to discover for himself that Gauss is the manager of two departments.

We note an interesting "practical" application for Armstrong relations. Silva and Melkanoff [50] have developed a database design aid in which the database designer inputs a set of FDs and MVDs. The design aid then presents him with an Armstrong relation, that is, a "sample relation" that obeys just those dependencies that are

logical consequences of those that he has inputted. (As we discuss soon, Armstrong relations exist in the presence of FDs and MVDs, and this is the case in which Silva and Melkanoff were interested.) Let us say, for example, that the designer gives as input the set {EMP → DEPT, DEPT → MGR} of FDs. The database design aid would then present the designer with an Armstrong relation, such as relation R in Figure 10, for this set of dependencies. The designer would then inspect the sample relation and might observe, for example, "Here's a manager, namely Gauss, who manages two distinct departments. *Therefore*, the dependencies that I inputted must not have implied that no manager can manage two distinct departments. Since I want this to be a constraint for my database, I'd better input the FD MGR → DEPT."

In this example the designer did not have to think explicitly about the dependency MGR → DEPT and whether or not it was a consequence of the dependencies that he input; rather, by seeing the Armstrong relation and thinking about what it said, he simply *noticed* that the FD MGR → DEPT failed. Thus Silva and Melkanoff's approach is a partial solution, in the spirit of Query-by-Example [60], to the problem of helping a designer think of what dependencies should be included.

Unfortunately, it turns out [6] that the time complexity of finding an Armstrong relation, given a set of functional dependencies, is precisely exponential in the number of attributes. That is, there exists an exponential-time algorithm, and furthermore there is an example in which the time simply to write down the Armstrong relation is exponential.

In ordinary first-order logic (where arbitrary first-order sentences, and not just our dependencies, are allowed) there can be no Armstrong relations. For example, let Σ be the empty set $\emptyset$. Assume that R is a relation that obeys just Σ^* (i.e., just the tautologies) and no other first-order sentences. Let σ be an arbitrary first-order sentence such that neither σ nor $\neg\sigma$ is a tautology. Clearly, R must obey one of σ or $\neg\sigma$; thus R obeys a nontautology. This is a contradiction. Thus there is a witness for σ (a relation that shows that σ is not a tautology) and a witness for $\neg\sigma$ (a relation that shows that $\neg\sigma$ is not a tautology), but there is no global witness (a relation that simultaneously shows that σ is not a tautology and $\neg\sigma$ is not a tautology).

It is common to speak of a relation obeying an "accidental" dependency, that is, a dependency that is not a logical consequence of the collection of "specified" dependencies. Thus each specified dependency is supposed to hold "for all time," that is, for every "snapshot" (instance) of the database, whereas an accidental dependency is one that happens to hold in some snapshot of the database but may fail in other snapshots. An Armstrong relation is precisely one that obeys every specified dependency and no accidental dependency.

Beeri et al. [7] generalized Armstrong's result to allow not just FDs but also MVDs. That is, they showed that if Σ is a set of FDs and MVDs and Σ^* is the set of all FDs and MVDs that are logical consequences of Σ, then there is a relation (an "Armstrong relation for Σ") that obeys the FDs and MVDs in Σ^* and no other FDs or MVDs. The proof was subtly incorrect in that it neglected the case of MVDs (and FDs) for which the left-hand side is the empty set. Beeri [4] generalized the result to allow FDs, MVDs, and JDs (join dependencies). His proof was rather long, and his technique does not generalize to allow embedded MVDs. We generalize to allow EIDs (which includes all of the above, including embedded MVDs) and even more.

We state our next theorem rather generally, since it has applications in various fields and not just in database theory. In our general setting we assume that there is a class of *models* (which, in our case of immediate interest, is the class of nonempty

relations), a class of *sentences*, and a relationship HOLDS between these two classes, which tells when a given sentence holds for a given model. Thus, if σ is a sentence and R is a model, then HOLDS(σ, R) means that σ "holds for" R or that R "obeys" σ. We then define $\models$ and Σ^* in the natural way. Thus, if Σ is a set of sentences and σ is a single sentence, then $\Sigma \models \sigma$ means that every model that obeys Σ also obeys σ; we say then that σ is a *logical consequence* of Σ. We say that a set Σ of sentences is *consistent* if Σ has a model, that is, if there is a model that simultaneously obeys every sentence in Σ.

THEOREM 3.1. *Let $\mathscr{S}$ be a set of sentences. The following properties of $\mathscr{S}$ are equivalent.*

(*a*) **Existence of a faithful operator.** *There is an operator $\oplus$ that maps nonempty families of models into models, such that if σ is a sentence in $\mathscr{S}$ and $\langle R_i : i \in I \rangle$ is a nonempty family of models, then σ holds for $\oplus\langle R_i : i \in I \rangle$ if and only if σ holds for each R_i.*

(*b*) **Existence of Armstrong models.** *Whenever Σ is a consistent subset of $\mathscr{S}$ and Σ^* is the set of sentences in $\mathscr{S}$ that are logical consequences of Σ, then there is a model (an "Armstrong model") that obeys Σ^* and no other sentences in $\mathscr{S}$.*

(*c*) **Splitting of disjunctions.** *Whenever Σ is a subset of $\mathscr{S}$ and $\{\sigma_i : i \in I\}$ is a nonempty subset of $\mathscr{S}$, then $\Sigma \models \bigvee\{\sigma_i : i \in I\}$ if and only if there is some i in I such that $\Sigma \models \sigma_i$.*

Note. Earlier we made the assumption that whenever we speak of a *family* $\langle R_i : i \in I \rangle$, we always mean a *nonempty family* (i.e., that the index set I is nonempty). In Theorem 3.1(a) we have made this assumption explicit, since this assumption needs to be dealt with explicitly in the proof of Theorem 3.1.

In Theorem 3.1(c) above, when we say $\Sigma \models \bigvee\{\sigma_i : i \in I\}$, we mean that every model that obeys Σ necessarily obeys some σ_i; thus we can think of $\bigvee\{\sigma_i : i \in I\}$ as a big disjunction. If the index set I is infinite, then this disjunction is infinite.

We prove Theorem 3.1 at the end of this section. We first make some comments.

Parts (a)–(c) of Theorem 3.1 certainly need not hold in general. For example, let a "model" be a binary relation, and let $\mathscr{S}$ be the set of all first-order sentences about binary relations. We showed earlier that Theorem 3.1(b) fails, that is, that there can be no binary relation that obeys precisely the tautologies (about binary relations). Similarly, we now show directly that (c) fails. Let Σ be $\emptyset$, the empty set, and let σ be a sentence such that neither σ nor its negation is a tautology. Then $\Sigma \models (\sigma \vee \neg\sigma)$, but $\Sigma \not\models \sigma$ and $\Sigma \not\models \neg\sigma$. So, (c) fails. Of course, (a) fails also, since (a)–(c) are equivalent. When $\mathscr{S}$ is a set of sentences for which (a)–(c) of Theorem 3.1 hold, then we say that $\mathscr{S}$ *enjoys Armstrong models.* (If the models are relations, then of course we may say that $\mathscr{S}$ *enjoys Armstrong relations.*) In Section 5 we present other examples of sets $\mathscr{S}$ that do not enjoy Armstrong models.

Remark. In the remainder of this paper, whenever we make statements about collections of EIDs, we assume that all EIDs mentioned contain the same relation symbol, with the same arity.

Parts (b) and (c) of Theorem 3.1 deal with consistent subsets Σ of $\mathscr{S}$. We note that in our case of primary interest, in which $\mathscr{S}$ is the set of EIDs, every subset Σ of $\mathscr{S}$ is consistent, since a one-tuple relation obeys every EID. (We cannot take the empty relation to show consistency, since our definition of "model" in this case is the class of nonempty relations.)

Before we can apply Theorem 3.1 to our case of primary interest (where $\mathscr{S}$ is the set of EIDs and $\oplus$ is $\otimes$), we must do a little bit of fussing, because of the minor bother that empty relations have been neglected. If Σ is a set of EIDs and σ is a single EID, then by $\Sigma \models_{\text{nonempty}} \sigma$, we mean that every *nonempty* relation that obeys Σ necessarily obeys σ.

LEMMA 3.2. *Let Σ be a set of EIDs and σ a single EID. Then $\Sigma \models_{\text{nonempty}} \sigma$ if and only if $\Sigma \models \sigma$.*

PROOF. Clearly, if $\Sigma \models \sigma$, then $\Sigma \models_{\text{nonempty}} \sigma$. Conversely, assume that $\Sigma \models_{\text{nonempty}} \sigma$, but $\Sigma \not\models \sigma$. Since $\Sigma \not\models \sigma$, there is a relation R that obeys Σ but not σ. The relation R must be empty, since by assumption $\Sigma \models_{\text{nonempty}} \sigma$. Thus the empty relation R violates the EID σ. But EIDs have been defined in such a way that they are true about empty relations. This is a contradiction. □

COROLLARY 3.3. *Let Σ be a set of EIDs, and let Σ^* be the set of EIDs that are logical consequences of Σ. Then there is a relation that obeys Σ^* and no other EIDs, that is, there is an Armstrong relation for Σ.*

PROOF. In Theorem 3.1, let $\mathscr{S}$ be the set of all EIDs (about d-ary relations), let a "model" be a nonempty d-ary relation, and let $\oplus$ be the direct product $\otimes$. Theorem 2.1 says that Theorem 3.1(a) then holds. So, by Theorem 3.1, we know that (b) holds. That is, we know that there is a relation that obeys precisely those EIDs σ such that $\Sigma \models_{\text{nonempty}} \sigma$. So, by Lemma 3.2, there is a relation that obeys precisely those EIDs σ such that $\Sigma \models \sigma$. This was to be shown. □

Note that the Armstrong relation of Corollary 3.3 is not unique. For, it is easy to verify that the direct product of Armstrong relations for Σ is also an Armstrong relation for Σ. If R has k tuples, then $R \otimes R$ has k^2 tuples; hence, if R has more than one tuple, then $R \otimes R$ is not isomorphic to R (since it has more tuples). So, R and $R \otimes R$ are nonisomorphic Armstrong relations for Σ. Beeri et al. [6] have various results about the size of *minimal* Armstrong relations in the presence of FDs.

COROLLARY 3.4. *Let Σ be a set of EIDs, and let $\sigma_1, \sigma_2, \ldots$ each be EIDs. Then $\Sigma \models (\sigma_1 \vee \sigma_2 \vee \cdots)$ if and only if there is some i such that $\Sigma \models \sigma_i$.*

PROOF. It is obvious that if there is some i such that $\Sigma \models \sigma_i$, then $\Sigma \models (\sigma_1 \vee \sigma_2 \vee \cdots)$. Conversely, assume that $\Sigma \models (\sigma_1 \vee \sigma_2 \vee \cdots)$. All the more so, we know that $\Sigma \models_{\text{nonempty}} (\sigma_1 \vee \sigma_2 \vee \cdots)$. In Theorem 3.1 let $\mathscr{S}$ be the set of all EIDs (about d-ary relations), let a "model" be a nonempty d-ary relation, and let $\oplus$ be the direct product $\otimes$. Theorem 2.1 says that Theorem 3.1(a) then holds. So, by Theorem 3.1, we know that (c) holds. Hence, since $\Sigma \models_{\text{nonempty}} (\sigma_1 \vee \sigma_2 \vee \cdots)$, we know that there is some i such that $\Sigma \models_{\text{nonempty}} \sigma_i$. So, by Lemma 3.2, we know that $\Sigma \models \sigma_i$. This was to be shown. □

The reason for our interest in "faithfulness with respect to direct product" is not because of anything inherent about the direct product as such, but rather that the direct product is an operator $\oplus$ that fulfills Theorem 3.1(a) for a natural class $\mathscr{S}$ of sentences. Furthermore, it is nice that the direct product is fairly simple conceptually and that it is often fairly easy to verify in practice whether or not a given sentence is faithful with respect to direct product.

Theorem 3.1 might well be useful in a number of contexts. Brooks [11] has noted an application of Theorem 3.1 in which a "model" is a set of test data about a computer program and a "sentence" is a characterization of the computations done

by a program. Brooks is interested in obtaining what he calls a "generic model," which is a collection of test data with no unneeded relationships. Since his environment obeys Theorem 3.1(c), it obeys (b) also, which guarantees generic models. A famous example in logic where Theorem 3.1(b) is well known occurs when the set $\mathscr{S}$ of sentences is the set of all equations over a given set of function symbols. Then the free algebra with countably many generators [30] is an Armstrong model. Interestingly enough, in this case the operator $\oplus$ in Theorem 3.1(a) again turns out to be the direct product. Another interesting operator that can sometimes be used to play the role of $\oplus$ in Theorem 3.1(a) is the *disjoint union.* The disjoint union of a collection of relations (all with the same attributes) is obtained by first replacing each relation by an isomorphic copy in such a way that no entry in one relation equals any entry in any of the other relations; then a new relation is formed by taking the union of all of the tuples in all of the relations. If a "sentence" is an FD in which the left-hand side is nonempty or an MVD in which the left-hand side is nonempty, a "model" is a relation (with the appropriate attributes), and $\oplus$ is disjoint union, then Theorem 3.1(a) holds. This, in fact, was the proof technique used by Beeri et al. [7] to show the existence of Armstrong relations in the presence of FDs and MVDs (although they neglected to "patch" the proof to deal with FDs and MVDs in which the left-hand side is empty).

We are now ready to prove Theorem 3.1.

PROOF OF THEOREM 3.1

(a) $\Rightarrow$ (b). Assume Theorem 3.1(a); we shall prove (b). Let Σ be consistent, and let **BAD** be the set of all sentences in $\mathscr{S}$ that are *not* logical consequences of Σ. There are two cases, depending on whether or not **BAD** is empty.

Case 1. **BAD** is empty. So, $\Sigma^* = \mathscr{S}$. By assumption, Σ is consistent, that is, Σ has a model R. Clearly R itself is the desired "Armstrong model," which obeys Σ^* and no other sentences in $\mathscr{S}$ (since there are no other sentences in $\mathscr{S}$).

Case 2. **BAD** is nonempty. For each σ in **BAD** we know (by definition of logical consequence) that there is a model R_σ that obeys Σ but not σ. Define R to be $\oplus\langle R_\sigma : \sigma \in \mathbf{BAD}\rangle$. We now show that R obeys Σ^* and no other sentence in $\mathscr{S}$.

For each sentence τ in Σ, we know by construction that every R_σ obeys τ. By property (a) it follows that R also obeys τ. So, R obeys Σ, and hence Σ^*. Now we must show that if σ is a sentence *not* in Σ^*, then R does *not* obey σ. Now σ is a sentence in **BAD**. By the definition of R_σ we know that R_σ does not obey σ. By (a) it follows that R does not obey σ. This was to be shown.

(b) $\Rightarrow$ (c). Assume Theorem 3.1(b); we shall prove (c). It is obvious that if there is some i such that $\Sigma \models \sigma_i$, then $\Sigma \models \bigvee\{\sigma_i : i \in I\}$. Conversely, assume that $\Sigma \models \bigvee\{\sigma_i : i \in I\}$, where I is a nonempty index set. If Σ is inconsistent, then let i be an arbitrary member of I. Since Σ is inconsistent, it logically implies everything, and in particular, $\Sigma \models \sigma_i$. So we may assume that Σ is consistent. Let R be the model guaranteed by (b) that obeys Σ^* and obeys no other sentence in $\mathscr{S}$. Since R obeys Σ, and since $\Sigma \models \bigvee\{\sigma_i : i \in I\}$, it follows that there is some i such that R obeys σ_i. So, by definition of R, we know that σ_i is in Σ^*. This was to be shown.

(c) $\Rightarrow$ (b). Assume Theorem 3.1(c); we shall prove (b). Assume that (b) is false. Then there is a consistent subset Σ of $\mathscr{S}$ such that no model that obeys Σ is an Armstrong model for Σ^*, that is, such that each model M that obeys Σ also obeys a sentence σ_M not in Σ^*. Let I be the set of all models of Σ; by the consistency of Σ, we

know that I is nonempty. By the definition of the sentences σ_M, we know that $\Sigma \models \bigvee\{\sigma_M : M \in I\}$. By (c) we know that for some M, necessarily $\Sigma \models \sigma_M$. Thus σ_M is in Σ^*. This is a contradiction.

(b) $\Rightarrow$ (a). Assume Theorem 3.1(b); we shall prove (a). We now define $\oplus\langle R_i : i \in I\rangle$, where I is a nonempty index set and each R_i is a model. For each R_i define T_i to be the set of all sentences of $\mathscr{S}$ that hold for R_i. Let Σ equal $\bigcap\{T_i : i \in I\}$.

We first show that Σ is consistent. Since I is nonempty, take i in I. Clearly, $\Sigma \subseteq T_i$. So, since R_i obeys T_i, R_i also obeys Σ. Thus Σ is consistent.

We now show that $\Sigma = \Sigma^*$. The inclusion $\Sigma \subseteq \Sigma^*$ always holds, so we must show that $\Sigma^* \subseteq \Sigma$. That is, we must show that if τ is a sentence of $\mathscr{S}$ such that $\Sigma \models \tau$, then τ is in Σ. Assume that $\Sigma \models \tau$. Fix i in I. Now $\Sigma \subseteq T_i$, and $\Sigma \models \tau$. It follows that $T_i \models \tau$. Since T_i is the set of all sentences of $\mathscr{S}$ obeyed by R_i, and since $T_i \models \tau$, it follows that R_i obeys τ, and so τ is in T_i. We have shown that if $\Sigma \models \tau$, then τ is in T_i for every i in I. Hence τ is in Σ, which was to be shown.

We have shown that Σ is consistent and $\Sigma = \Sigma^*$. So, by Theorem 3.1(b) we know that there is a model R that obeys Σ and no other sentences in $\mathscr{S}$. Let us define $\oplus\langle R_i : i \in I\rangle$ to be R. To prove (a), we must show that if σ is an arbitrary sentence of $\mathscr{S}$, then σ holds for R if and only if σ holds for each R_i. Assume first that σ holds for R; we must show that σ holds for each R_i. Since σ holds for R, we know (by definition of R) that σ is in $\bigcap\{T_i : i \in I\}$. Thus, for every i we know that σ is in T_i. Hence σ holds for R_i, which was to be shown. Conversely, assume that σ holds for every R_i. Then σ is in T_i for every i, and so σ is in Σ (by definition of Σ). So, by definition of R, we know that σ holds for R. □

We close this section by noting that the existence of Armstrong relations is a property of *collections* of sentences and not of single sentences. Thus there can be no theorem that says something like, "Armstrong relations can only exist in the context of EIDs." As a dramatic example, let τ be a totally arbitrary sentence, and let $\mathscr{S}$ be the singleton set $\{\tau\}$. It is easy to verify that Theorem 3.1(c) then holds. Thus $\mathscr{S}$ enjoys Armstrong models (i.e., conditions (a)–(c) of Theorem 3.1 hold).

It is easy to see that if $\mathscr{S}_1$ enjoys Armstrong models and $\mathscr{S}_2 \subseteq \mathscr{S}_1$, then $\mathscr{S}_2$ also enjoys Armstrong models. It is an interesting problem to consider classes $\mathscr{S}$ that enjoy Armstrong models. Note that the collection of such classes $\mathscr{S}$ is *not* closed under finite union. For, we just noted that every singleton set $\mathscr{S}$ enjoys Armstrong models. So, if the collection of classes $\mathscr{S}$ that enjoy Armstrong models were closed under finite union, then every finite set $\mathscr{S}$ of sentences would enjoy Armstrong models. However, in Section 5 we exhibit several finite sets $\mathscr{S}$ of sentences that do not enjoy Armstrong models.

4. *Finite Armstrong Relations*

In the previous section we were not explicitly concerned with whether or not the relations we were dealing with are *finite* (i.e., have a finite number of tuples). In particular, as we noted, our construction of taking a direct product of a possibly infinite collection of relations can lead to an infinite relation, even though every component of the direct product is a finite relation. In this section we deal specifically with the existence of *finite* Armstrong relations.

If Σ is a set of sentences about a d-ary relation and σ is a single such sentence, then we say that $\Sigma \models_{\text{fin}} \sigma$ if every finite d-ary relation that obeys Σ also obeys σ. It has been shown [26] that there is a set of four EIDs Σ and a single EID σ such that

$\Sigma \models_{\text{fin}} \sigma$, but for which $\Sigma \not\models \sigma$. In fact, the EIDs in that example are all TDs (template dependencies). We also note that Vardi [57] and Gurevich and Lewis [31] have shown that both the decision problem and the finite decision problem for TDs are undecidable. That is, they have shown that the problems of deciding whether $\Sigma \models \sigma$ and whether $\Sigma \models_{\text{fin}} \sigma$ are each undecidable.

We have the following result, which is the analog of Corollary 3.3, but where we are only interested in *finite* relations.

THEOREM 4.1. *Let $\mathcal{S}$ be a* finite *set of EIDs. Let Σ be a subset of $\mathcal{S}$, and let Σ^*_{fin} be those members σ of $\mathcal{S}$ for which $\Sigma \models_{fin} \sigma$. Then there is a finite relation that obeys Σ^*_{fin} and no other member of $\mathcal{S}$.*

PROOF. The proof is almost the same as that of Corollary 3.3, except that we restrict our attention to finite relations. The key point is that the direct product of a finite number of finite relations is a finite relation. □

Let $\mathcal{S}$ be the set of all FDs, MVDs, EMVDs, JDs, and EJDs (embedded join dependencies) over a given set of attributes. Then $\mathcal{S}$ is a finite set of dependencies. This is an important special case of Theorem 4.1.

Example 4.2. We now show that Theorem 4.1 would be false if we were to drop the restriction that $\mathcal{S}$ be finite. Let R be a binary relation with attributes A and B, let a be a member of dom(A), and let b be a member of dom(B). If t is a tuple (x, y) of R, we may write $t[A] = x$ and $t[B] = y$. We say that R has a *k-tuple path* from a to b if there are tuples $t_1, \ldots, t_k$ of R such that

$$\begin{array}{rl} a = t_1[A], & \\ t_i[B] = t_{i+1}[B] & \text{if } i \text{ is odd and } 1 \le i < k, \\ t_i[A] = t_{i+1}[A] & \text{if } i \text{ is even and } 2 \le i < k, \\ b = t_k[B]. & \end{array}$$

For example, R has a 5-tuple path from a to b if there are elements $x_1, \ldots, x_4$ such that the following five tuples appear in R:

$$\begin{array}{l} (a, x_1), \\ (x_2, x_1), \\ (x_2, x_3), \\ (x_4, x_3), \\ (x_4, b). \end{array}$$

Let τ_i ($i = 1, 3, 5, \ldots$) be the EID that says, "Whenever there is an $(i + 2)$-tuple path from a to b, then there is an i-tuple path from a to b." For example, τ_3 is

$$\begin{array}{l} (\forall a b x_1 x_2 x_3 x_4)((Pax_1 \wedge Px_2x_1 \wedge Px_2x_3 \wedge Px_4x_3 \wedge Px_4b) \\ \quad \Rightarrow (\exists y_1 y_2)(Pay_1 \wedge Py_2y_1 \wedge Py_2b)). \end{array}$$

Yannakakis [58] introduced the EIDs τ_i to show that there are an infinite number of nonequivalent EIDs. We now make use of these EIDs to show that Theorem 4.1 is false if the assumption that $\mathcal{S}$ is finite is dropped. Let Σ be the empty set $\emptyset$. Assume that there is a finite Armstrong relation for $\emptyset$, that is, there is a finite relation R that obeys only trivial EIDs. (An EID σ is said to be *trivial*, or *tautologous*, if $\emptyset \models \sigma$.) Since R is finite, it has a finite number k of tuples. But then R clearly obeys the nontrivial EID τ_k. This is a contradiction. □

We note that Fagin et al. [26] have strengthened the result of Example 4.2 by using only TDs. In particular, they show that there is a set Σ of two TDs such that there is

no finite relation that obeys precisely those TDs σ for which $\Sigma \models_{\text{fin}} \sigma$. Also, Vardi [56] has shown that there is an EID τ such that the set of EIDs σ where $\tau \models_{\text{fin}} \sigma$ is not recursive. This result implies that there is no finite Armstrong relation for τ, since we could test whether or not $\tau \models_{\text{fin}} \sigma$ by simply checking whether or not the finite Armstrong relation obeys σ.

We close this section with a few miscellaneous comments about Armstrong relations. The construction of the previous section can generate an Armstrong relation that is not only infinite, but even uncountable, since, as we noted earlier, the direct product of a countably infinite number of relations, each of which has at least two tuples, contains *uncountably* many tuples. However, the Lowenheim–Skolem Theorem [20, p. 141] implies that there is then an Armstrong relation with a *countable* number of tuples. Of course, in this section we have been interested in Armstrong relations with a *finite* number of tuples.

The reader might be concerned that the Armstrong relations we have created have strange entries, such as ⟨Smith, Jones, Thomas⟩ in the EMP column of the direct product of three relations. However, one can systematically replace each occurrence of ⟨Smith, Jones, Thomas⟩ everywhere it occurs by a single unique name, such as Anderson, with a new unique name for each triple. The new relation is then isomorphic to the earlier relation, since we have simply renamed the entries. Thus the new relation is still an Armstrong relation. This renaming would be desirable, for example, in the application of Silva and Melkanoff [50], described earlier. The alert reader may have noticed that in this renaming process, we have tacitly assumed that there are as many distinct names as we want. Consideration of bounded domains immediately leads to combinatorial problems (see [24] and [34] for examples of such problems).

We note that Fagin et al. [26] present a necessary and sufficient condition for the existence of finite Armstrong relations in the presence of TDs. Their necessary and sufficient condition is given in terms of the implication structure of TDs.

Finally, we remark that our direct product construction of an Armstrong relation is especially valuable when we desire to produce a *finite* Armstrong relation in the presence of *embedded* dependencies. For, in the presence of embedded dependencies, chase-type procedures for constructing relations tend to produce an *infinite* relation. (Grant and Jacobs [29] describe such a chase-type procedure, which they describe in terms of deductions. For a discussion, see [25].)

5. *More Armstrong Relation Counterexamples*

In this section we present three amusing counterexamples about Armstrong relations. In each case we exhibit a set Σ of sentences and single sentences σ_1 and σ_2, such that $\Sigma \models (\sigma_1 \vee \sigma_2)$, but such that $\Sigma \not\models \sigma_1$ and $\Sigma \not\models \sigma_2$. Thus, if $\mathscr{S}$ is a set of sentences that includes Σ and each of σ_1 and σ_2 (and if a "model" is a relation of the appropriate arity), then Theorem 3.1(c) fails. In particular, by the proof of Theorem 3.1, it follows that there is no Armstrong relation for Σ (with respect to sentences $\mathscr{S}$). Thus $\mathscr{S}$ does not enjoy Armstrong relations.

Example 5.1. Let σ_0 be the sentence (which is *not* an EID) that says that the relation has at most two tuples. Thus (assuming that we are dealing with binary relations), the sentence σ_0 is

$$(\forall x_1 y_1 x_2 y_2 x_3 y_3)((Px_1y_1 \wedge Px_2y_2 \wedge Px_3y_3) \Rightarrow (((x_1 = x_2) \wedge (y_1 = y_2)) \vee ((x_1 = x_3) \wedge (y_1 = y_3)) \vee ((x_2 = x_3) \wedge (y_2 = y_3)))).$$

We note that σ_0 is equivalent to (the conjunction of) a set of eight sentences, each of the form

$$(\forall x_1 y_1 x_2 y_2 x_3 y_3)((Px_1y_1 \wedge Px_2y_2 \wedge Px_3y_3) \Rightarrow (B_1 \vee B_2 \vee B_3)),$$

where each B_i is an equality. Note that these sentences "start out" looking like IDs (1.4) but have a disjunction of atomic formulas, rather than a single atomic formula, on the right-hand side of the implication.

Let Σ be $\{\sigma_0\}$, σ_1 be the FD $A \rightarrow B$, and σ_2 be the FD $B \rightarrow A$. We now show that $\Sigma \models (\sigma_1 \vee \sigma_2)$. Let R be a relation obeying Σ. Thus R has at most two tuples. We must show that R obeys either σ_1 or σ_2. If R has zero or one tuple, then R obeys both σ_1 and σ_2. So assume that R has exactly two tuples t_1 and t_2. There are two cases, depending on whether $t_1[A] = t_2[A]$ or $t_1[A] \neq t_2[A]$. If $t_1[A] = t_2[A]$, then σ_2 holds. If $t_1[A] \neq t_2[A]$, then σ_1 holds. So, either σ_1 or σ_2 holds, which was to be shown.

We have shown that $\Sigma \models (\sigma_1 \vee \sigma_2)$. However, it is easy to verify that $\Sigma \not\models \sigma_1$ and $\Sigma \not\models \sigma_2$. □

Example 5.2. Let σ_1 be the sentence (which is *not* an EID) that says that the relation is nonempty. Thus (assuming that we are dealing with binary relations) the sentence σ_1 is $(\exists xy)Pxy$. Note that σ_1 is of the form (1.7) of an EID, *except* that $n = 0$ (i.e., the left-hand side is empty).

Let Σ be $\emptyset$, the empty set, and let σ_2 be the FD $A \rightarrow B$.

Then $\Sigma \models (\sigma_1 \vee \sigma_2)$; for, if S is a nonempty relation, then it obeys σ_1, and if S is an empty relation, then it obeys σ_2. However, $\Sigma \not\models \sigma_1$ and $\Sigma \not\models \sigma_2$. □

Example 5.3. This example is due to Statman [52]. We show that if we deal with sentences that look like IDs, except that they are not typed, then there is not necessarily an Armstrong relation.

Let σ_0, σ_1, and σ_2 be the following three sentences respectively (where the first two are *not* typed):

$$(\forall x_0 y_0 x_1 y_1 z_1 y_2 z_3)((Px_0x_0y_0 \wedge Px_1y_1z_1 \wedge Px_1y_2z_2) \Rightarrow (y_1 = y_2)),$$
$$(\forall x_0 y_0)(Px_0x_0y_0 \Rightarrow Px_0x_0x_0),$$
$$(\forall x_1 y_1 z_1 y_2 z_2)((Px_1y_1z_1 \wedge Px_1y_2z_2) \Rightarrow (y_1 = y_2)).$$

Let Σ be $\{\sigma_0\}$. It is easy to verify that $\Sigma \not\models \sigma_1$ and $\Sigma \not\models \sigma_2$. However, we now show that $\Sigma \models (\sigma_1 \vee \sigma_2)$. Assume not. Let R be a relation that obeys σ_0 but violates σ_1 and σ_2. Since σ_1 fails, there are a_0 and b_0 such that $Ra_0a_0b_0$ holds. Since σ_2 fails, there are a_1, b_1, c_1, b_2, and c_2 such that $Ra_1b_1c_1$ holds, $Ra_1b_2c_2$ holds, and $b_1 \neq b_2$. But then σ_0 fails, a contradiction. □

6. *Projections of Classes of Relations*

We assume throughout this section that we are dealing only with finite relations. Thus, in this section, whenever we say "relation," we mean "finite relation." We call the collection of all relations with attributes U that obey a given set of FDs an *FD class*. (Ginsburg and Zaiddan [28] define a closely related notion, called a *functional dependency family*, which is like an FD class except that they also fix the domains of the attributes. We do not fix domains, by analogy with the usual definition of such classes, e.g., *elementary classes* [20], in logic.) Ginsburg and Zaiddan show that a projection of an FD class is not necessarily an FD class. Thus, later in this section we exhibit a set Σ of FDs that all deal with 5-ary relations with attributes $ABCDE$, where the following happens. Let $\mathscr{R}$ be the class of all 5-ary relations that obey Σ,

and let $\mathscr{T}$ be the class of all relations that are projections of members of $\mathscr{R}$ onto $ABCD$. Then there is no set Σ' of FDs such that $\mathscr{T}$ is precisely the class of all relations with attributes $ABCD$ that obey Σ'.

However, we can show that there is a set Σ' of *EGDs* (about 4-ary relations) such that $\mathscr{T}$ is precisely the class of all relations that obey Σ'. (Recall that an EGD, or equality-generating dependency, is an ID for which the right-hand side is an equality.) Thus, although $\mathscr{T}$ is not an FD class, it is an EGD class, that is, $\mathscr{T}$ is the collection of all relations that obey a given set Σ' of EGDs. In fact, the following theorem holds (we present the proof later in this section).

THEOREM 6.1. *Every projection of an EGD class is an EGD class.*

Let Σ, $\mathscr{R}$, and $\mathscr{T}$ be as above. Consider the following natural scenario. In some application, Σ is the set of constraints that each instance R (with attributes $ABCDE$) must obey, and so $\mathscr{R}$ is the collection of possible instances that the relation can assume. Thus the possible instances are precisely those relations that obey the set Σ of FDs. Assume that Jones is a user who has a "view" of the database in which he sees only the first four columns $ABCD$ of the relation R (Jones might, e.g., be shielded from seeing the E column of the relation, for privacy or security reasons). Thus the possible instances for his view are precisely the relations in $\mathscr{T}$. Then the set of constraints for Jones' view is given not by a set of FDs, but by a set of EGDs.

It follows immediately from Theorem 6.1 that every projection of an FD class is an EGD class. Ginsburg and Zaiddan [28] define a class of dependencies called *implied dependencies*, which are special cases of EGDs and show that every projection of an FD class is an implied dependency class.

Recall that an *FTGD* (full tuple-generating dependency) is an ID in which the right-hand side is a relational formula. We have the following results (which we prove soon).

THEOREM 6.2. *Every projection of an FTGD class is an FTGD class.*

THEOREM 6.3. *Every projection of an ID class is an ID class.*

Thus EGD classes, FTGD classes, and ID classes obey a natural closure property that FD classes do not. As we saw, by considering FD classes and their projections one is "forced" into considering more general dependencies, such as IDs. Apparently, Sadri [45] obtained Theorems 6.1–6.3 independently. We note that Hull [33] showed that the join of ID classes is an ID class. Thus the collection of ID classes is closed under projection and join.

Hull [33] has given an example of an FD class with a projection $\mathscr{T}$ such that $\mathscr{T}$ is an ID class given by an infinite set of IDs, but such that $\mathscr{T}$ is not equivalent to an ID class given by any finite set of IDs. It is an interesting open problem to characterize those cases where the projection $\mathscr{T}$ would be given by a *finite* set of IDs. Another open problem is whether or not the projection of an EID class is necessarily an EID class.

Let R be a fixed relation. In the spirit of Ginsburg and Zaiddan [28], we define *the FD class generated by* R to be the smallest FD class that contains R. It is easy to see that this class is simply those relations (with attributes the same as those of R) that obey Σ, where Σ is the set of all FDs obeyed by R. A natural question is whether every FD class has a generator. The answer [28] is yes: if the FD class $\mathscr{R}$ consists of all relations with attributes U that obey Σ, then let R be an Armstrong relation (with attributes U) for Σ^*; it is easy to see that R is a generator for the class $\mathscr{R}$. Similarly, we can define generators for EID classes and obtain the result that every EID class

A	B	C	D
0	0	0	0
0	1	1	0
1	1	0	1

FIGURE 11

has a generator (by once again taking an Armstrong relation). Thus a natural interpretation for Armstrong relations is as class generators.

Before we present our promised example of an FD class whose projection is not an FD class, we prove a simple lemma, which was first shown by Ginsburg and Zaiddan [28].

LEMMA 6.4. *Let $\mathscr{R}$ be the class of all relations (with attributes U) that obey the set Σ of FDs (over U). Let V be a subset of U, and let $\mathscr{T}$ be the class of all projections onto V of members of $\mathscr{R}$. If $\mathscr{T}$ is an FD class, then $\mathscr{T}$ is the class of all relations over V that obey Σ', where Σ' is the set of FDs over V that are logical consequences of Σ.*

Note. See the comments after (1.5) in the introduction.

PROOF. Assume that $\mathscr{T}$ is an FD class, say, the class of all relations over V that obey Γ. We must show that Γ is equivalent to Σ', that is, that each logically implies the other.

$\Gamma \models \Sigma'$: Assume not. Then there is a relation T that obeys Γ but not some σ in Σ'. By assumption, T is in $\mathscr{T}$, and so there is a relation R in $\mathscr{R}$ such that T is a projection of R. Since T violates the FD σ, and since T is a projection of R, necessarily R violates σ. But R obeys Σ (since R is in $\mathscr{R}$), and so R obeys Σ' (because $\Sigma \models \Sigma'$). Thus R obeys σ. This is a contradiction.

$\Sigma' \models \Gamma$: It is certainly sufficient to show that $\Gamma \subseteq \Sigma'$. Take γ in Γ. To show that γ is in Σ', we must show (by definition of Σ') that $\Sigma \models \gamma$. Assume that $\Sigma \not\models \gamma$. Then there is a relation R in $\mathscr{R}$ that violates γ. Let T be the projection of R onto V. Then T violates γ. This is a contradiction, since T is in $\mathscr{T}$. □

Example 6.5. We now present our example of an FD class with a projection that is not an FD class. Let Σ be the set $\{B \to E, D \to E, CE \to A\}$ of FDs over $ABCDE$, and let $\mathscr{R}$ be the class of all relations over $ABCDE$ that obey Σ. Let $\mathscr{T}$ be the class of all projections of $\mathscr{R}$ onto $ABCD$. We now show that $\mathscr{T}$ is not an FD class. Assume that it were. Then, by Lemma 6.4, $\mathscr{T}$ is the class of all relations over $ABCD$ that obey Σ', where Σ' is the set of all FDs over $ABCD$ that are logical consequences of Σ. Note that Σ' contains some nontrivial FDs; an example is the FD $BC \to A$ (the FD $BC \to A$ is in Σ' because of the FDs $B \to E$ and $CE \to A$ in Σ). Let T be the relation in Figure 11. We now show that (i) T obeys Σ', and (ii) T is not in $\mathscr{T}$. This is a contradiction. □

T obeys Σ': We shall show that the relation consisting of each pair of tuples from T obeys Σ'. We begin with the first two tuples of Figure 11. It is easy to verify (by using, say, the FD membership algorithm of Beeri and Bernstein [5]) that if $Y \subseteq \{A, B, C, D\}$ and $AD \to Y$ is an FD in Σ', then $AD \to Y$ is a trivial FD, that is, that $Y \subseteq \{A, D\}$. Since the first two tuples in Figure 11 agree precisely on AD, it follows that the relation consisting of the first two tuples in Figure 11 obeys Σ'. Similarly, the relation consisting of the first and third tuples and the relation consisting of the

second and third tuples obey Σ'. It follows that the whole relation (relation T) of Figure 11 obeys Σ'.

T is not the projection of any relation in $\mathscr{R}$: Assume that T were the projection of a relation R in $\mathscr{R}$. Then R would consist of at least three tuples t_1, t_2, and t_3, such that the projection $t_i[ABCD]$ equal the ith tuple in Figure 11 ($i = 1, 2, 3$). Because of the FD $D \rightarrow E$ in Σ, it follows that $t_1[E] = t_2[E]$. Because of the FD $B \rightarrow E$ in Σ, it follows that $t_2[E] = t_3[E]$. Thus $t_1[E] = t_3[E]$. But then, because of the FD $CE \rightarrow A$, it follows that $t_1[A] = t_3[A]$. But this is false.

We have shown that $\mathscr{T}$, the set of projections onto $ABCD$ of members of $\mathscr{R}$, is not an FD class. What went wrong here is that every member of $\mathscr{T}$ obeys the *EGD* τ that says, "If there are three tuples such that the first and second tuples agree in the AD columns, the first and third agree in the C column, and the second and third agree in the B column, then the first and third agree in the A column." Formally, this EGD is

$$(\forall a_0 b_0 c_0 d_0 a_1 b_1 c_1 d_1)((Pa_0b_0c_0d_0 \wedge Pa_0b_1c_1d_0 \wedge Pa_1b_1c_0d_1) \Rightarrow (a_0 = a_1)). \quad (6.1)$$

Then, for every 5-ary relation that obeys Σ, its projection onto the first 4 columns obeys τ. Relation T in Figure 11 does not obey τ and so is not the projection of a member of $\mathscr{R}$. This concludes this example. □

PROOF OF THEOREMS 6.1–6.3. We first prove Theorem 6.3, and then we indicate how to modify the proof to prove Theorems 6.1 and 6.2. Let $\mathscr{R}$ be an ID class, say, the class of all relations (with attributes U) that obey the set Σ of IDs. Let V be a subset of U, $\mathscr{T}$ the class of all projections onto V of members of $\mathscr{R}$, and Σ' the set of all IDs (over relations with attributes V) that hold for every member of $\mathscr{T}$. The proof is complete if we show that $\mathscr{T}$ is the class of all relations (with attributes V) that obey Σ', since this would show that $\mathscr{T}$ is an ID class. Certainly, every relation in $\mathscr{T}$ obeys Σ', by the definition of Σ'. So, we need only show that each relation that obeys Σ' is in $\mathscr{T}$. Let T be a relation that obeys Σ'. We must show that $T \in \mathscr{T}$, that is, that there is a relation R (with attributes U) that obeys Σ and such that T is the projection of R onto V. Create a tableau with columns U and with as many rows as there are tuples in T. Order the tuples of T, let the ith row of the tableau look exactly like the ith tuple of T (when we restrict our attention to V), and let new, distinct variables appear in each of the other entries. Thus, if A is an attribute in V, then the A entry for the ith row of the tableau equals the A entry for the ith tuple of T, and if A is an attribute not in V, then the A entry for the ith row of the tableau is a new, distinct variable. Now apply the chase procedure (using Σ) to the tableau [8, 37, 46] (actually, we are doing a slight generalization of the chase, since Σ may be infinite). The important point is that the chase procedure terminates with a finite tableau, since the dependencies in Σ are full, and so no new symbols are added during the chase. Let us treat the final tableau as a relation, which we call R. The chase procedure guarantees that R obeys Σ. Let us denote the projection of R onto V by T'; we must show that $T = T'$.

$T' \subseteq T$: Let t be a tuple of T'. We must show that t is in T. Let us denote the tuples of T by $t_1, \ldots, t_k$. Since every dependency in Σ is full, it follows easily that every entry of t is an entry of some t_i. Let us denote by α the FTGD (over relations with attributes V) that tells us that if $t_1, \ldots, t_k$ are tuples, then so is t (α is constructed in a similar manner to how the EGD (6.1) above was constructed). Then α holds for every relation in $\mathscr{T}$. Thus α is in Σ'. So, T obeys α, and so t is in T. This was to be shown.

$T \subseteq T'$: The only way that this could fail would be if the chase procedure were to force two entries of T that were not originally equal to be equal. It is not hard to see that this means that Σ' contains an EGD that tells us that if $t_1, \ldots, t_k$ are tuples, then, say, $t_i[A] = t_j[A]$. (As above, $t_1, \ldots, t_k$ are of tuples of T.) But then T itself would have obeyed this EGD (since T obeys Σ'). So, the chase procedure cannot force two entries of T that were not originally equal to be equal.

This concludes the proof of Theorem 6.3. We now indicate how to modify the proof to prove Theorems 6.1 and 6.2. Let us consider first Theorem 6.1. The proof is identical to the proof of Theorem 6.3, except in the portion of the proof in which we show that $T' \subseteq T$. If $T' \not\subseteq T$, then this would be caused by an EGD in Σ that forced two entries of the tableau to be equal (in the case of Theorem 6.1, Σ contains only EGDs and no FTGDs). But then there would be a corresponding EGD in Σ', that forces the same entries to be equal. Finally, the proof of Theorem 6.2 is the same as the proof of Theorem 6.3, except that the inclusion $T \subseteq T'$ is automatic (because Σ contains only FTGDs, and no EGDs). □

7. *Interrelational and Nontyped Dependencies*

In this section we discuss a generalization of EIDs, in which the assumption that the sentences are typed and unirelational is weakened. However, faithfulness is maintained. Our enlarged class of sentences includes the important *inclusion dependencies* [12, 24], which can say, for example, that every manager is an employee. The models of interest are no longer simply relations, but instead databases, consisting of a number of relations. It turns out that Armstrong databases need not exist in our new context, but that something almost as strong takes place.

Since in this section we deal with databases, rather than with single relations, we need some more conventions.

We assume that we are given a fixed finite set of *relation symbols* (usually called *relation names* in practice), and a positive integer, called the *arity*, associated with each relation symbol. A *database* is a mapping that associates a relation (of the proper arity) with each relation symbol. When it can cause no confusion, we may speak of the collection of relations themselves as the database. We can write first-order sentences about databases, just as we earlier wrote first-order sentences about single relations. For example, assume that PROF and STAFF are among the relation names. Assume that we wish to write a sentence σ that says that the first column of the instance of PROF is a subset (not necessarily proper) of the second column of the instance of STAFF. This sentence might represent the fact that, say, every professional employee is an employee on the staff. Assume that, say, PROF is binary and STAFF is ternary. Such a sentence σ is

$$(\forall ax)(\text{PROF}ax \Rightarrow (\exists yz)\text{STAFF}yaz). \tag{7.1}$$

An *extended embedded implicational dependency* (or *XEID*) is a sentence of the form

$$(\forall x_1 \cdots x_m)((A_1 \wedge \cdots \wedge A_n) \Rightarrow (\exists y_1 \cdots y_r)(B_1 \wedge \cdots \wedge B_s)), \tag{7.2}$$

where each A_i is a relational formula and each B_i is atomic (either a relational formula or an equality). As in the case of EIDs, we assume also that each of the x_j's appears in at least one of the A_i's, and that $n \geq 1$, that is, that there is at least one A_i. So far, everything that we have said holds for both EIDs and XEIDs. For EIDs we made the further assumption that the sentence is typed and unirelational. For XEIDs

we make the weaker assumption that the *left-hand side* $A_1 \wedge \cdots \wedge A_n$ is typed and unirelational. For example, sentence (7.1) is an XEID that is not an EID.

Surely, from a practical point of view, the most important example of an interrelational dependency (and of a nontyped dependency) is the *inclusion dependency* [12, 24], or *IND*, of which (7.1) is a special case. It says, intuitively, that the entries in the A column of a relation are a subset of the B entries in the same or another relation. For example, it might say that every manager is an employee. More generally, an IND can say that the projection onto a given m columns in one relation is a subset of the projection onto a given m columns in the same or another relation. If P is 3-ary and Q is 4-ary, then the IND that says that the entries in the first two columns of P (in that order) are a subset of the entries in the fourth and second columns of Q (in that order) can be written

$$(\forall abx)(Pabx \Rightarrow (\exists yz)Qybza).$$

We now define direct product and faithfulness for databases.

As in Example 2.7 of Section 2, the direct product is simply defined relationwise. Thus, if Q is one of the relation names, then the Q relation of the direct product is simply the direct product (under our usual definition) of the Q relations of the components of the direct product.

A database is *relationwise nonempty* if every relation in the database is nonempty. We say that a sentence σ is *upward faithful* (with respect to direct products) if whenever $\langle D_i : i \in I \rangle$ is a family of relationwise nonempty databases such that σ holds for every D_i, then σ holds for $\otimes\langle D_i : i \in I \rangle$. We say that a sentence σ is *downward faithful* (with respect to direct products) if whenever $\langle D_i : i \in I \rangle$ is a family of relationwise nonempty databases such that σ holds for $\otimes\langle D_i : i \in I \rangle$, then σ holds for every D_i. Clearly, σ is faithful if and only if it is both upward and downward faithful.

By a trivial modification to the proof of Theorem 2.6 we can obtain a proof of the following theorem.

THEOREM 7.1. *Let σ be a sentence of the form*

$$(\forall x_1 \cdots x_m)(\phi \Rightarrow (\exists y_1 \cdots y_r)\gamma),$$

where ϕ is a quantifier-free formula and γ is a positive quantifier-free formula. Assume further that ϕ is typed and unirelational. Then σ is downward faithful.

Note that the only difference between the statements of Theorems 2.6 and 7.1 is that in Theorem 2.6 we assume that σ is typed and unirelational, whereas in Theorem 7.1 we make the weaker assumption that the left-hand side ϕ is typed and unirelational.

THEOREM 7.2. *Every XEID is faithful.*

PROOF. By the transformation at the beginning of the proof of Theorem 2.1, we can transform every XEID into a sentence of the form mentioned in Horn's Theorem (Theorem 2.4). Thus, by Horn's Theorem, every XEID is upward faithful. By Theorem 7.1, every XEID is downward faithful. Thus every XEID is faithful, which was to be shown. □

Note that XEIDs, like EIDs, have the property that they are domain independent. This means that if two databases have the same tuples in corresponding relations, but possibly distinct domains of attributes, then the two databases agree on XEIDs. That is, given an XEID σ, either both databases obey σ or both disobey σ. Also,

XEIDs, like EIDs, have the property that they are automatically true about empty databases (databases with no tuples).

Let Σ be a set of sentences and σ a single sentence. We say that σ is a *logical consequence (with respect to relationwise nonempty databases)* of Σ if every relationwise nonempty database that obeys Σ also obeys σ.

THEOREM 7.3. *Let Σ be a set of XEIDs, and let $\Sigma^*_{nonempty}$ be the set of XEIDs that are logical consequences (with respect to relationwise nonempty databases) of Σ. Then there is a database that obeys $\Sigma^*_{nonempty}$ and no other XEIDs.*

PROOF. In Theorem 3.1, let $\mathscr{S}$ be the set of all XEIDs, let a "model" be a relationwise nonempty database, and let $\oplus$ be the direct product $\oplus$. Theorem 7.2 says that Theorem 3.1(a) then holds. So, by Theorem 3.1, we know that (b) holds. This is exactly what was to be shown. □

We now show that Theorem 7.3 would be false if we were to substitute "logical consequence" for "logical consequence (with respect to relationwise nonempty database)." Thus, Theorem 7.3 shows the existence of "Armstrong-like" databases in the presence of XEIDs: *Armstrong-like*, rather than *Armstrong*, because we are using "logical consequence (with respect to relationwise nonempty databases)" rather than simply "logical consequence." Our next example (Example 7.4) shows that although there is an Armstrong-like database in the presence of XEIDs (Theorem 7.3), there is not necessarily an Armstrong database.

Example 7.4. Let P and Q refer to unary relations. Let σ_1 be the XEID $(\forall x)(Px \Rightarrow Qx)$, and let σ_2 be the XEID $\forall x(Qx \Rightarrow \exists y Py)$. Let Σ be the empty set $\emptyset$. We now show that $\Sigma \models (\sigma_1 \vee \sigma_2)$. Let D be a database (with P and Q among the relation names). If the P relation of D is empty, then σ_1 holds for D. If the P relation of D is nonempty, then σ_2 holds for D. Thus $\Sigma \models (\sigma_1 \vee \sigma_2)$. However, $\Sigma \not\models \sigma_1$ and $\Sigma \not\models \sigma_2$. So, there is no Armstrong database for Σ, since each database D obeys either σ_1 or σ_2, neither of which is in Σ^*. □

The reason that Theorem 7.3 holds as it stands but fails if we were to substitute "logical consequence" for "logical consequence (with respect to relationwise nonempty databases)" is that the analog of Lemma 3.2 fails. Thus, in Example 7.4 the XEID σ_2 is not a logical consequence of Σ, but it *is* a logical consequence (with respect to relationwise nonempty databases) of Σ.

However, we now show that if we restrict our attention to XEIDs that are unirelational, then there are Armstrong databases. Like EIDs, unirelational XEIDs deal with only one relation; however, unlike EIDs, they need not be typed (only the left-hand side needs to be typed).

THEOREM 7.5. *Let Σ be a set of unirelational XEIDs, and let Σ^* be the set of unirelational XEIDs that are logical consequences of Σ. Then there is a database that obeys Σ^* and no other unirelational XEIDs.*

PROOF. Define Σ_R to be those XEIDs in Σ that contain the relation symbol R. If σ is a unirelational XEID that contains the relation symbol R, then it is easy to see that $\Sigma \models \sigma$ if and only if $\Sigma_R \models \sigma$. It follows easily that an Armstrong database for Σ can be obtained by taking the collection of individual Armstrong relations, one for each R. We conclude the proof by showing that Σ_R has an Armstrong relation, that is, that there is a relation that obeys precisely those unirelational XEIDs involving R that are logical consequences of Σ_R. This follows from Theorem 7.2, just as Corollary 3.3 followed from Theorem 2.1. □

Fagin and Vardi [27] show that if we restrict our attention to INDs and FDs, then there are *not* necessarily Armstrong databases. However, they show that if we restrict our attention to INDs and to "standard" FDs (FDs for which the left-hand side is nonempty), then there *are* necessarily Armstrong databases.

We say that a database is *finite* if each of its relations is finite. We note that an analogous theorem to Theorem 4.1 (which deals with the existence of finite Armstrong relations) holds about the existence of finite Armstrong-like databases. Earlier we mentioned that an interesting special case of Theorem 4.1 occurs when the only dependencies of interest are FDs, MVDs, EMVDs, JDs, and EJDs; similarly, an interesting special case of this analogous theorem occurs when the only dependencies of interest are those just mentioned, along with INDs involving attributes for the relations in the database. A similar comment applies, in the case of unirelational XEIDs, for the existence of finite Armstrong databases. Also, a similar comment applies for Fagin and Vardi's result [27], mentioned above, about finite Armstrong databases in the presence of INDs and standard FDs.

We close this section by noting that the class of XEIDs is closed under conjunction. That is, the conjunction of two XEIDs (and hence, by induction, the conjunction of any finite number of XEIDs) is equivalent to an XEID. The proof is identical to Beeri and Vardi's proof [9] that the class of EIDs is closed under conjunction. Thus, assume that σ and σ' are XEIDs; we shall show that the sentence $\sigma \wedge \sigma'$ is equivalent to an XEID τ. Let σ be the XEID

$$(\forall x_1 \cdots x_m)(\phi \Rightarrow (\exists y_1 \cdots y_r)\psi),$$

and let σ' be the XEID

$$(\forall x'_1 \cdots x'_{m'})(\phi' \Rightarrow (\exists y'_1 \cdots y'_{r'})\psi').$$

Assume further that no variable appears both in σ and in σ'. Let τ be the XEID

$$(\forall x_1 \cdots x_m x'_1 \cdots x'_{m'})((\phi \wedge \phi') \Rightarrow (\exists y_1 \cdots y_r y'_1 \cdots y'_{r'})(\psi \wedge \psi')).$$

We claim that the conjunction $\sigma \wedge \sigma'$ is equivalent to τ. It is very easy to see that $(\sigma \wedge \sigma') \models \tau$. The proof of the opposite direction (that $\tau \models \sigma$ and $\tau \models \sigma'$) depends in a rather subtle way on the fact that the left-hand sides ϕ and ϕ' are each typed and unirelational. This proof is an amusing exercise for the curious reader.

8. *Summary*

We have introduced implicational and embedded implicational depencencies, in an attempt to find a natural class of "dependencies" for relations in a relational database. We have shown that these dependencies are all faithful with respect to direct product, although slight variations are not necessarily faithful. The existence of Armstrong relations in the presence of these dependencies follows from the faithfulness property. We have shown, in fact, that the existence of Armstrong relations is equivalent to faithfulness with respect to *some* operator. We have shown that it is possible for infinite Armstrong relations to exist without finite Armstrong relations existing, and we have given conditions that guarantee the existence of finite Armstrong relations.

We have shown that the projection of an implicational dependency class is again an implicational dependency class (although the projection of a functional dependency class is not necessarily a functional dependency class).

Finally, we have introduced *extended* embedded implicational dependencies, which, unlike ordinary embedded implicational dependencies, may be interrelational and nontyped. This class includes the inclusion dependency, which says, for example,

that every manager is an employee. We have shown the existence of an Armstrong-like database in the presence of extended embedded implicational dependencies.

ACKNOWLEDGMENTS. The author is grateful to M. Brooks, S. Ginsburg, S. Givant, R. Hull, R. McKenzie, H. R. Strong, J. D. Ullman, M. Y. Vardi, and R. L. Vaught for helpful comments.

REFERENCES

1 AHO, A.V., BEERI, C., AND ULLMAN, J.D. The theory of joins in relational databases. *ACM Trans. Database Syst. 4,* 3 (Sept. 1979), 297–314.
2. ARMSTRONG, W.W Dependency structures of database relationships. *Proc. IFIP 74.* North Holland, Amsterdam, 1974, pp. 580–583.
3. ARMSTRONG, W.W., AND DELOBEL, C. Decompositions and functional dependencies in relations. *ACM Trans. Database Syst 5,* 4 (Dec. 1980), 404–430.
4. BEERI, C. Personal communication, 1979.
5. BEERI, C , AND BERNSTEIN, P.A. Computational problems related to the design of normal form relational schemas *ACM Trans. Database Syst. 4,* 1 (Mar. 1979), 30–59.
6. BEERI, C., DOWD, M., FAGIN, R., AND STATMAN, R On the structure of Armstrong relations for functional dependencies. To appear in *J. ACM.*
7. BEERI, C , FAGIN, R., AND HOWARD, J.H. A complete axiomatization for functional and multivalued dependencies in database relations. Proc. Int. ACM-SIGMOD Conf. on Management of Data, Toronto, Ont., Can., 1977, pp. 17–61.
8. BEERI, C., AND VARDI, M.Y. A proof procedure for data dependencies. Tech. Rep., Hebrew Univ. of Jerusalem, Jerusalem, Israel, Aug. 1980.
9 BEERI, C., AND VARDI, M.Y. Formal systems for tuple and equality-generating dependencies. Tech. Rep , Hebrew Univ. of Jerusalem, Jerusalem, Israel, Apr. 1981
10. BEERI, C., AND VARDI, M.Y. The implication problem for data dependencies. In *Proc. 8th Int. Conf. on Automata, Languages, and Programming,* Lecture Notes in Computer Science 115, Springer-Verlag, New York, 1981, pp. 73–85
11. BROOKS, M. Determining correctness by testing. Ph.D Dissertation, Stanford Univ., Stanford, Calif., 1980.
12. CASANOVA, M A., FAGIN, R., AND PAPADIMITRIOU, C Inclusion dependencies and their interaction with functional dependencies. Proc. 1st ACM SIGACT-SIGMOD Conf. on Principles of Database Systems, Los Angeles, Calif., 1982, pp. 171–176.
13. CHANG, C C., AND KEISLER, H.J *Model Theory.* North Holland, Amsterdam, 1973.
14. CODD, E.F. Further normalization of the database relational model. In *Data Base Systems,* Courant Computer Science Symposia 6, R Rustin, Ed., Prentice Hall, Englewood Cliffs, N.J., 1971, pp. 33–64.
15. CODD, E.F. Relational completeness of data base sub-languages. In *Data Base Systems,* Courant Computer Science Symposium, R. Rustin, Ed , Prentice Hall, Englewood Cliffs, N.J. 1971, pp. 65–98
16. COOPER, E.C. On the expressive power of query languages. Tech. Rep. TR-14-80, Center for Research in Computing Technology, Harvard Univ., Cambridge, Mass., 1980.
17. DEMOLOMBE, R. A syntactical characterization of a subset of definite and safe formulas. Tech. Rep., ONERA-CERT, Toulouse, France, Sept. 1981.
18. DEMOLOMBE, R., AND NICOLAS, J.-M. On the characterization of "valid" formulas for database querying. Tech. Rep., ONERA-CERT, Toulouse, France, Sept. 1981.
19. DI PAOLA, R. The recursive unsolvability of the decision problem for the class of definite formulas *J ACM 16,* 2 (Apr. 1969), 324–327.
20. ENDERTON, H.B. *A Mathematical Introduction to Logic.* Academic Press, New York, 1972.
21 FAGIN, R. Multivalued dependencies and a new normal form for relational databases. *ACM Trans. Database Syst 2,* 3 (Sept. 1977), 262–278.
22. FAGIN, R. Functional dependencies in a relational database and propositional logic. *IBM J. Res. Devel. 21,* 6 (Nov. 1977), 534–544.
23 FAGIN, R. Horn clauses and database dependencies. Proc. 12th Ann. ACM Symp. on Theory of Computing, Los Angeles, Calif , 1980, pp. 123–134 (extended abstract)
24 FAGIN, R. A normal form for relational databases that is based on domains and keys. *ACM Trans. Database Syst. 6,* 3 (Sept 1981), 387–415
25. FAGIN, R Armstrong databases. Proc. 7th IBM Symp. on Mathematical Foundations of Computer Science, Kanagawa, Japan, May 1982.

26. FAGIN, R., MAIER, D., ULLMAN, J.D., AND YANNAKAKIS, M. Tools for template dependencies. To appear in *SIAM J. Comput.*
27. FAGIN, R., AND VARDI, M.Y. Armstrong databases for functional and inclusion dependencies. Res. Rep. RJ3500, IBM Research Laboratory, San Jose, Calif., June 1982.
28. GINSBURG, S., AND ZAIDDAN, S.M. Properties of functional-dependency families. *J. ACM 29*, 3 (July 1982), 678–698.
29. GRANT, J., AND JACOBS, B.E. On the family of generalized dependency constraints. *J. ACM 29*, 4 (Oct. 1982), 986–997.
30. GRÄTZER, G. *Universal Algebra*, 2nd ed. Springer-Verlag, New York, 1979.
31. GUREVICH, Y., AND LEWIS, H.R. The inference problem for template dependencies. Proc. 1st ACM SIGACT-SIGMOD Symp. on the Principles of Database Systems, Los Angeles, Calif., 1982, pp. 221–229.
32. HORN, A. On sentences which are true of direct unions of algebras *J. Symbolic Logic 16* (1951), 14–21.
33. HULL, R. Implicational dependency and finite specification. Tech. Rep., Univ. of Southern California, Los Angeles, Calif., 1981.
34. KANELLAKIS, P.C. On the computational complexity of cardinality constraints in relational databases. *Inf. Proc. Lett. 11*, 2 (Oct. 1980), 98–101.
35. KEISLER, H.J. Some applications of infinitely long formulas. *J. Symbolic Logic 30*, 3 (Sept. 1965), 339–349.
36. KUHNS, J.L. Answering questions by computer: A logical study. Tech. Rep RM-5428-PR, Rand Corp., Santa Monica, Calif., Dec. 1967.
37. MAIER, D., MENDELZON, A., AND SAGIV, Y. Testing implications of data dependencies. *ACM Trans. Database Syst. 4*, 4 (Dec. 1979), 455–469.
38. MENDELZON, A.O., AND MAIER, D. Generalized mutual dependencies and the decomposition of database relations. Proc. 5th Int. Conf. on Very Large Data Bases, Rio de Janeiro, Brazil, 1979, pp. 75–82.
39. NICOLAS, J.-M. First-order logic formalization for functional, multivalued, and mutual dependencies. Proc. ACM SIGMOD Int Conf. on Management of Data, Austin, Texas, 1978, pp. 40–46.
40. NICOLAS, J.-M. Logic for improving integrity checking in relational databases. To appear in *Acta Inf.*
41. PARADAENS, J. Transitive dependencies in a database scheme. Tech Rep R387, MBLE, Brussels, Belgium, 1979.
42. PARADAENS J., AND JANSSENS, D. Decompositions of relations: a comprehensive approach. In *Advances in Data Base Theory*, Vol. 1, H. Gallaire, J. Minker, and J-M. Nicolas, Eds., Plenum Publishing, New York, 1981.
43. PARKER, D.S., AND PARSAYE-GHOMI, K. Inference involving embedded multivalued dependencies and transitive dependencies Proc. Int. ACM-SIGMOD Conf on Management of Data, Los Angeles, Calif., 1980, pp. 52–57.
44. RISSANEN, J. Theory of relations for databases—A tutorial survey. *Proc. 7th Symp. on Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science 64, J. Winkowski, Ed., Springer-Verlag, New York, pp. 537–551.
45. SADRI, F. Personal communication.
46. SADRI, F, AND ULLMAN, J.D. Template dependencies: A large class of dependencies in relational databases and its complete axiomatization. *J. ACM 29*, 2 (Apr. 1982), 363–372.
47. SAGIV, Y., AND WALECKA, S.F. Subset dependencies and a completeness result for a subclass of embedded multivalued dependencies. *J. ACM 29*, 1 (Jan. 1982), 103–117.
48. SAGIV, Y., DELOBEL, C., PARKER, D.S. JR., AND FAGIN, R. An equivalence between relational database dependencies and a fragment of propositional logic. *J. ACM 28*, 3 (July 1981), 435–453.
49. SHOENFIELD, J.R. *Mathematical Logic.* Addison-Wesley, Reading, Mass., 1967.
50. SILVA, A.M., AND MELKANOFF, M.A. A method for helping discover the dependencies of a relation. In *Advances in Data Base Theory*, Vol. 1, H. Gallaire, J. Minker, and J.-M. Nicolas, Eds., Plenum Publishing, New York, 1981.
51. SLAGLE, J.R., AND KONIVER, D. Finding resolution proofs using duplicate goals in AND/OR trees. *Inf. Sci. 4* (1971), 313–342.
52. STATMAN, R. Private communication.
53. TARSKI, A. Contributions to the theory of Models I. *Nederl. Akad. Wetensch. Proc. Ser. A 57* (1954), 572–588 (Indag. Math., vol 16).
54 ULLMAN, J.D. *Principles of Database Systems.* Computer Science Press, Woodland Hills, Calif., 1980.
55. VARDI, M.Y. The decision problem for database dependencies. *Inf. Proc. Lett. 12*, 5 (Oct 1981), 251–254.

56. VARDI, M.Y. Private communication
57 VARDI, M.Y. The implication and finite implication problems for typed template dependencies. Proc 1st ACM SIGACT-SIGMOD Conf. on Principles of Database Systems, Los Angeles, Calif, 1982, pp. 230–238.
58. YANNAKAKIS, M. Private communication.
59. YANNAKAKIS, M., AND PAPADIMITRIOU, C. Algebraic dependencies. Proc. 21st IEEE Symp. on Foundations of Computer Science, Syracuse, N.Y., 1980, pp. 328–332. To appear in *J. Comp. Syst. Sci*
60. ZLOOF, M.M. Query-by-example. *Proc. 1975 AFIPS NCC*, Vol. 44 AFIPS Press, Arlington, Va., 1975, pp 431–438.

RECEIVED MARCH 1980; REVISED DECEMBER 1980; ACCEPTED JUNE 1981