Information-Based Distance Measures and the Canonical Reflection of View Updates

Stephen J. Hegner
Umeå University
Department of Computing Science
SE-901 87 Umeå, Sweden
hegner@cs.umu.se
http://www.cs.umu.se/~hegner

Abstract

For the problem of reflecting an update on a database view to the main schema, the constant-complement strategies are precisely those which avoid all update anomalies, and so define the gold standard for well-behaved solutions to the problem. However, the families of view updates which are supported under such strategies are limited, so it is sometimes necessary to go beyond them, albeit in a systematic fashion. In this work, an investigation of such extended strategies is initiated for relational schemata. The approach is to characterize the information content of a database instance, and then require that the optimal reflection of a view update to the main schema embody the least possible change of information. The key property is identified to be *strong monotonicity* of the view, meaning that view insertions may always be reflected as insertions to the main schema, and likewise for deletions. In that context it is shown that for insertions and deletions, an optimal update, entailing the least change of information, exists and is unique up to isomorphism for wide classes of constraints.

1 Introduction

1.1 The limitations of constant complement The problem of reflecting view updates to the main schema of a database system is a difficult one whose solution invariably involves compromise. The constant-complement approach [BS81] avoids all so-called update anomalies [Heg04], and so is the gold standard for well-behaved strategies. On the other hand, it is also quite conservative regarding the updates which it admits. A short example will help to illustrate the scope of this approach. Let $\mathbf{E}_0$ be the relational schema consisting of the single relation symbol $R[ABC]$, constrained by the join dependency $\bowtie[AB, BC]$, and let $\Pi^{\mathbf{E}_0}_{AB}$ be the view whose single relation symbol is $R_{AB}[AB]$ and whose morphism $\pi^{\mathbf{E}_0}_{AB}$ is the projection of $R[ABC]$ onto AB. In the constant-complement strategy, all updates to $\Pi^{\mathbf{E}_0}_{AB}$ must hold a so-called *complementary view* fixed. The natural complement to $\Pi^{\mathbf{E}_0}_{AB}$ is the view $\Pi^{\mathbf{E}_0}_{BC}$, defined by the projection of $R[ABC]$ onto $R_{BC}[BC]$. It is easy to see that the updates to $\Pi^{\mathbf{E}_0}_{AB}$ which hold $\Pi^{\mathbf{E}_0}_{BC}$ fixed are precisely those which hold the projection onto B fixed. Thus, for example, if

$\{R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0), R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1)\}$ is the current instance of $\mathbf{E}_0$, so that the instance of the view schema is $\{R_{AB}(\mathrm{a}_0, \mathrm{b}_0), R_{AB}(\mathrm{a}_1, \mathrm{b}_1)\}$, then insertion of $R_{AB}(\mathrm{a}_2, \mathrm{b}_1)$ is realized by inserting $R(\mathrm{a}_2, \mathrm{b}_1, \mathrm{c}_1)$ into the instance of the main schema. Unfortunately, even so simple an update as inserting $R_{AB}(\mathrm{a}_2, \mathrm{b}_2)$ into $\Pi_{AB}^{\mathbf{E}_0}$ is not supported, since the projection onto B cannot be held fixed under such an update.

Of course, there is a reason for this limitation. In order to insert $R_{AB}(\mathrm{a}_2, \mathrm{b}_2)$ into the view, it is necessary to insert some tuple of the form $R(\mathrm{a}_2, \mathrm{b}_2, c_x)$ into the main schema, with information about c_x not visible within the view. Such an insertion would violate the principle that views be encapsulated with respect to the updates which are allowed, in the sense that the effect of all such updates be contained entirely within the view itself. It is precisely the constant-complement strategy which guarantees this sort of encapsulation [Heg04, Sec. 1.2]. Nevertheless, there are certainly situations in which it is desirable, if not necessary, to lift this limitation in a controlled manner. The goal of this paper is to develop an extension to the constant-complement strategy which admits a wider class of view updates while preserving as many of the desirable properties of the original strategy. In particular, the following three properties are regarded as uncompromisable.

Invariance of admissibility: The admissibility of a view update must depend only upon the current view instance, and not upon the instance of the main schema which gave rise to it.

Canonicity of reflections: All allowable reflections of a view update to the main schema must be equivalent up to some natural notion of isomorphism.

Reflection of monotonicity: If the view mapping is monotonic, then every insertion (resp. deletion) on the view must be reflected as an insertion (resp. deletion) on the main schema.

These conditions are all natural extensions of that which is expected from a constant complement strategy. Invariance of admissibility is always satisfied by a constant-complement strategy; see (cc:1) of [Heg04, Sec. 1.2]. While the reflection defined by a constant-complement strategy may in principle depend upon the choice of complement, it has been shown that it is in fact independent of that choice when the view morphism is monotonic [Heg04, Thm. 4.3] [Heg08c, Cor. 4.24]. Reflection of monotonicity is similarly guaranteed in wide variety of circumstances; see [Heg04, Def. 3.1 (upt:6)] and [Heg08c].

Much of the existing work on this problem, such as [DB82], [Kel85], [Lan90], [BL97], and [BL98], focuses upon translation of view updates via the relational algebra. As such, while they provide useful insight into commonly occurring problem instances, and all support the simple update problem of inserting $R_{AB}(\mathrm{a}_2, \mathrm{b}_2)$ sketched above, they do not provide a unified theory of how and under which circumstances view updates could be reflected. While such a detailed comparison is a very interesting topic, it must be left to a different paper.

1.2 Database repairs, distance measures, and information content More closely related to the approach developed here is one which has been developed in the logic-programming community – database repair. Roughly, in the repair problem, a database M is given which is inconsistent with respect to a set Ψ of constraints, the task being to "repair" M to a consistent version M' which obeys the constraints in Ψ. It is straightforward to recast a view-update problem in this context, at least in principle. Extending the example of 1.1,

let $\mathbf{E}_0'$ be the schema which augments $\mathbf{E}_0$ with the relation symbol $R_{AB}[AB]$ together with its defining constraint $(\forall x)(\forall y)(R_{AB}(x,y) \Leftrightarrow (\exists z)(R(x,y,z)))$. Thus, the current instance of this schema is $M_{\text{old}} = \{R(\mathrm{a}_0,\mathrm{b}_0,\mathrm{c}_0), R(\mathrm{a}_1,\mathrm{b}_1,\mathrm{c}_1), R_{AB}(\mathrm{a}_0,\mathrm{b}_0), R_{AB}(\mathrm{a}_1,\mathrm{b}_1)\}$. The "defective" new instance which includes the inserted view tuple $R_{AB}(\mathrm{a}_2,\mathrm{b}_2)$ but not the corresponding update to the main schema is $M'_{\text{def}} = M_{\text{old}} \cup \{R_{AB}(\mathrm{a}_2,\mathrm{b}_2)\}$. The task is to repair M'_{def} to be a legal instance, *i.e.*, one which satisfies the integrity constraints of $\mathbf{E}_0$, subject to the condition that the instance of R_{AB} is held constant. A ranking function measures the quality of the of various solutions. For any two sets S_1 and S_2, let $\mathsf{SymDiff}\langle S_1, S_2\rangle$ denote their symmetric difference $(S_1 \setminus S_2) \cup (S_2 \setminus S_1)$, and let $\mathsf{Card}(S_i)$ denote the cardinality of S_i. Two principal measures are *subset ranking* in which M_{new} is preferred to M'_{new} if $\mathsf{SymDiff}\langle M_{\text{old}}, M_{\text{new}}\rangle \subseteq \mathsf{SymDiff}\langle M_{\text{old}}, M'_{\text{new}}\rangle$ and the *count ranking* in which M_{new} is preferred to M'_{new} if $\mathsf{Card}(\mathsf{SymDiff}\langle M_{\text{old}}, M_{\text{new}}\rangle) \leq \mathsf{Card}(\mathsf{SymDiff}\langle M_{\text{old}}, M'_{\text{new}}\rangle)$. Under either of these measures, the optimal solutions are precisely those of the form $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,v)\rangle$, with v any allowable value for the domain C. See [ADNB06] for further details, as well as a comprehensive list of other papers which employ related approaches.

A drawback of both of these measures is that tuple similarity is an all-or-nothing affair — all tuples which are not identical are equally different from one another. Consequently, $R(\mathrm{a}_0,\mathrm{b}_0,\mathrm{c}_1)$ is just as different from $R(\mathrm{a}_0,\mathrm{b}_0,\mathrm{c}_0)$ as is $R(\mathrm{a}_1,\mathrm{b}_1,\mathrm{c}_1)$. Recently, more sophisticated distance measures have been proposed [ADB07], some of which are based upon (pseudo-)distance measures of individual tuples, such as those proposed in [Hut97] or [NC97]. These tuple-based measures may then be extended to sets of tuples via measures such as that of Eiter and Mannila [EM97], which defines the distance between database instances M_1 and M_2 in terms of the distances between tuples to be

$$\mathsf{Dist}\langle M_1, M_2\rangle = \frac{1}{2} \cdot \left(\sum_{t_1 \in M_1} \min_{t_2 \in M_2} \mathsf{Dist}\langle t_1, t_2\rangle + \sum_{t_2 \in M_2} \min_{t_1 \in M_1} \mathsf{Dist}\langle t_1, t_2\rangle \right).$$

Using a model of distance between tuples such as that of [NC97], which defines such distance in terms of how much each term of one tuple differs from the corresponding term in the other, and assuming that distinct terms have positive distance from one another, it follows that $\mathsf{Dist}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_1), R(\mathrm{a}_1,\mathrm{b}_1,\mathrm{c}_1)\rangle < \mathsf{Dist}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_2), R(\mathrm{a}_1,\mathrm{b}_1,\mathrm{c}_1)\rangle$ and so $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_1)\rangle$ is preferred to $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_2)\rangle$ in support of the view update request $\mathsf{Insert}\langle R_{AB}(\mathrm{a}_2,\mathrm{b}_2)\rangle$. Similarly, $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_0)\rangle$ is preferred to $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_2)\rangle$, while $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_0)\rangle$ and $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_1)\rangle$ are of equal preference.

Despite the obvious attractiveness of such tuple-based metrics from both a mathematical and an aesthetic point of view, in this paper it is argued that quite a different type of metric is more appropriate — one which in fact prefers $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_2)\rangle$ to both $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_1)\rangle$ and $\mathsf{Insert}\langle R(\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_0)\rangle$ — precisely the opposite of that which the above tuple-based metric advises. The idea is to measure the information content of database instances relative to a set of sentences, and to prefer updates which involve less change of information. More precisely, relative to a set Φ of sentences, the *information content* $\mathsf{Info}\langle M, \Phi\rangle$ of the database instance M is the set of all sentences in Φ which are satisfied by M. The *information distance* relative to Φ between database instances M_1 and M_2 is then $\Delta\langle(M_1, M_2), \Phi\rangle = \mathsf{SymDiff}\langle\mathsf{Info}\langle M_1, \Phi\rangle, \mathsf{Info}\langle M_2, \Phi\rangle\rangle$. The information distance between M_1 and M_2 is thus not a number but rather the set of formulas of Φ on which M_1 and M_2 differ. Preference of repairs is then defined in the obvious way, with changing M_1 to M_2 preferred to changing M_1 to M_3 if $\Delta\langle(M_1, M_2), \Phi\rangle \subseteq \Delta\langle(M_1, M_3), \Phi\rangle$.

The utility of this approach depends, of course, upon a suitable choice for Φ. If Φ chosen to be the set of all atoms for the schema, then information distance reduces to subset ranking as defined above. The most suitable choice in many situations is to let Φ be the set of all sentences in the language of the schema which are existential (no universal quantification), positive (no negation of any kind), and conjunctive (no disjunction), and which employ at most the constant symbols which occur in the formulation of the update itself — those which occur in the current instances of the main schema and the view as well as those which occur in the proposed new instance of the view. In the context of repairs, if the proposed update is an insertion, then this amounts to just the constant symbols which occur in the database to be repaired. On the other hand, if deletion of tuples is also allowed, then the constant symbols in tuples to be deleted must also be included. Updates will be formalized in detail in the core of this paper; for now, it suffices to write $\mathsf{ConstSym}(\mathbf{u})$ for the set of all constant symbols which occur in the update request $\mathbf{u}$. Then, using the notation to be introduced in the next section, the sentences of importance are $\mathsf{WFS}(\mathbf{D}, \exists\wedge+, \mathsf{ConstSym}(\mathbf{u}))$, with $\mathbf{D}$ the database schema under consideration.

Returning to the example based upon $\mathbf{E}_0'$, which augments $\mathbf{E}_0$ with the relation symbol $R_{AB}[AB]$ together with its defining constraint, insertion of $R_{AB}(\mathrm{a}_2, \mathrm{b}_2)$ into the view requires that the sentence $(\exists z)(R(\mathrm{a}_2, \mathrm{b}_2, z))$ be true in the main schema. This may be satisfied by binding z to any available constant and adding the corresponding tuple. However, if it is bound to a constant in $\mathsf{ConstSym}(M_{\mathrm{def}'}) = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{a}_2, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1\}$, then additional sentences will appear in the information content of the new instance. For example, if z is bound to c_1, then the sentence $R_{AB}(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_1)$ will also be in the information content of the new instance, as would the sentence $(\exists z)(R(\mathrm{a}_1, \mathrm{b}_1, z) \wedge R(\mathrm{a}_2, \mathrm{b}_2, z))$. Neither would be true were $R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2)$ inserted instead. Thus, to add the least possible information relative to sentences which only involve constants which are already used, z must be bound to a constant which does not occur in $\mathsf{ConstSym}(M_{\mathrm{def}'})$. Intuitively, this corresponds to binding it to a generic constant, and not one which also plays some other rôle in the database.

By itself, this information-based approach does not ensure tuple minimality. The insertion of both $R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2)$ and $R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_3)$ into M_{old} produces exactly the same added information as does the insertion of either alone, since each tuple adds precisely $(\exists z)(R(\mathrm{a}_2, \mathrm{b}_2, z))$ to the information content. To remedy this, tuple minimality (*i.e.*, minimality with respect to subset ranking) is also required. Thus, an optimal repair, whenever it exists, must be tuple minimal as well as minimal with respect to information change.

1.3 Further example To illustrate the ideas of information-based optimization of updates more completely, a slightly more complex example is presented. Let $\mathbf{E}_1$ be the relational schema with relations $R[ABC]$ and $S[CD]$, constrained by the inclusion dependency $R[C] \sqsubseteq S[C]$. Regard a database as a set of ground atoms over the associated logic. For example, $M_{00} = \{R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0), R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1), S(\mathrm{c}_0, \mathrm{d}_0), S(\mathrm{c}_1, \mathrm{d}_1)\}$ is such a database. Now, let K be a set of constants in the underlying logical language, regarded as domain elements for this schema. For information content, the base set Φ of sentences is $\mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K)$, the set of all positive (*i.e.*, no negation, explicit or implicit), existential, and conjunctive sentences in the language of the schema $\mathbf{E}_1$ which involve at most the constant symbols in K. Relative to this set, the information content of M is the set of all sentences in $\mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K)$ which are implied by M. The central step is to choose K properly. Using the notation to be introduced in 3.2, this

information content is denoted $\mathsf{Info}\langle M, \mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K)\rangle$.

A *cover* for this information content is a subset $\Psi \subseteq \mathsf{Info}\langle M, \mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K)\rangle$ such that Ψ and $\mathsf{Info}\langle M, \mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K)\rangle$ are logically equivalent. For $K_{00} = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{c}_0, \mathrm{c}_1, \mathrm{d}_0, \mathrm{d}_1\}$, the set of all constant symbols of M_{00}, the set M_{00} itself is clearly a cover for $\mathsf{Info}\langle M_{00}, \mathsf{WFS}(\mathbf{E}_3, \exists\wedge+, K_{00})\rangle$. On the other hand, with $K'_{00} = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0, \mathrm{b}_1, \mathrm{c}_0, \mathrm{d}_0\}$, a cover for $\mathsf{Info}\langle M_{00}, \mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K'_{00})\rangle$ is $\{R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0), S(\mathrm{c}_0, \mathrm{d}_0), (\exists x)(\exists y)(R(\mathrm{a}_1, \mathrm{b}_1, x)\wedge S(x, y))\}$. Note that the constants in $K_{00} \setminus K'_{00}$ have been replaced by existentially quantified variables.

To see how this idea is useful in the context of view updates, let $\Pi^{\mathbf{E}_1}_{R_{AB}[AB]} = (R_{AB}[AB], \pi^{\mathbf{E}_1}_{R[AB]})$ be the view of $\mathbf{E}_1$ which projects $R[ABC]$ onto $R_{AB}[AB]$ and which drops the relation S entirely. Consider M_{00} to be the initial instance of schema $\mathbf{E}_1$; its image instance in the view is then $N_{00} = \{R_{AB}(\mathrm{a}_0, \mathrm{b}_0), R_{AB}(\mathrm{a}_1, \mathrm{b}_1)\}$. Now, suppose that the view update $\mathsf{Insert}\langle R_{AB}(\mathrm{a}_2, \mathrm{b}_2)\rangle$ is requested, so that $N_{01} = N_{00} \cup \{R_{AB}(\mathrm{a}_2, \mathrm{b}_2)\}$ is the desired new view instance, and consider $M_{01} = M_{00} \cup \{R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2), S(\mathrm{c}_2, \mathrm{d}_2)\}$ as a proposed reflection to the main schema $\mathbf{E}_1$. Relative to its entire set $K_{01} = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{a}_2, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_2, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_2\}$ of constant symbols, a cover for $\mathsf{Info}\langle M_{01}, \mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K_{01})\rangle$ is just M_{01} itself. Similarly, for $M_{02} = M_{00} \cup \{R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_3), S(\mathrm{c}_3, \mathrm{d}_3)\}$ with $K_{02} = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{a}_2, \mathrm{b}_0, \mathrm{b}_1, \mathrm{b}_2, \mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_3, \mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_3\}$ a cover for $\mathsf{Info}\langle M_{02}, \mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K_{02})\rangle$ is just M_{02} itself. However, relative to K_{00}, which consists of the constant symbols found in elements of M_{00}, $\mathsf{Info}\langle M_{01}, \mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K_{00})\rangle = \mathsf{Info}\langle M_{02}, \mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K_{00})\rangle = M_{00} \cup \{(\exists x)(\exists y)(R(\mathrm{a}_2, \mathrm{b}_2, x)\wedge S(x, y))\}$. Denote this set of sentences by I_1. This recaptures formally that the proposed updates M_{01} and M_{02} are identical up to a renaming of the new constants. The utility of information measure is that it provides a means to recapture this idea formally.

Now, consider the alternative solution $M_{03} = M_{00} \cup \{R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_3), S(\mathrm{c}_3, \mathrm{d}_1)\}$ to this view-update problem. A cover for $\mathsf{Info}\langle M_{03}, \mathsf{WFS}(\mathbf{E}_1, \exists\wedge+, K_{00})\rangle$ is $I_3 = M_{00} \cup \{(\exists x)(R(\mathrm{a}_2, \mathrm{b}_2, x)\wedge S(x, \mathrm{d}_1))\}$, which is strictly stronger than I_1, *i.e.*, $I_2 \models I_1$, since $(\exists x)(R(\mathrm{a}_2, \mathrm{b}_2, x)\wedge S(x, \mathrm{d}_1)) \models (\exists x)(\exists y)(R(\mathrm{a}_2, \mathrm{b}_2, x)\wedge S(x, y)))$, but not conversely. Thus, relative to the information measure defined by K_{00}, M_{03} adds more information to M_{00} than does M_{01} or M_{02}. Similarly, $M_{04} = M_{00} \cup \{R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_0))\}$ adds more information than does M_{01} or M_{02}, since a cover for its information content is just M_{04} itself, which is stronger than I_1, since $R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_0)\wedge S(\mathrm{c}_0, \mathrm{d}_0) \models (\exists x)(\exists y)(R(\mathrm{a}_2, \mathrm{b}_2, x)\wedge S(x, y)))$, but not conversely.

The first and primary measure of quality of a reflected update is the change of information content which it induces. Under this measure, M_{01} and M_{02} are equivalent, and both are superior to either of M_{03} or M_{04}. However, this measure alone is not quite adequate. Rather, there is an additional measure of quality which must be taken into account. To illustrate, consider the proposed solution $M_{05} = M_{01} \cup M_{02} = M_{00} \cup \{R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_3), S(\mathrm{c}_2, \mathrm{d}_2), S(\mathrm{c}_3, \mathrm{d}_3)\}$ to this update problem. It has the same information content, I_1, relative to K_{00}, as do M_{01} and M_{02}. The information measure cannot distinguish the insertion of two new tuples with completely new constants from the insertion of just one. However, it is clear that M_{05} should be considered inferior to both M_{01} and M_{02} as a solution to the given update problem, since it is a proper superset of each. Therefore, a second criterion of quality is invoked; namely that no solution whose set of changes is a proper superset of those of another can be considered to be superior. In the terminology introduced earlier in this section, the update must be minimal under subset ranking, and not just under count ranking. For example, consider again the proposed solution M_{04}. From a strict counting point of view, M_{04} involves fewer changes than do M_{01} or M_{02}. However, neither M_{01} nor M_{02} is a superset of M_{04}. Thus, the superiority of M_{01}

and M_{02} is not contradicted. In other words, only solutions which are *tuple minimal*, in the sense that no proper subset of the changes is also an admissible solution, are permitted.

The main modelling premise of this paper is that the quality of a view update can be measured by the amount of change in information content which it induces, and so an optimal reflection of a view update request is one which is both tuple minimal and which induces the least amount of change of information content. Under this premise, both M_{01} and M_{02} are superior to each of M_{03}, M_{04}, and M_{05}. Furthermore, since M_{01} and M_{02} induce the same change in information content, they are equivalent. In Section 3, it is established that, under suitable conditions, all such optimal solutions are equivalent, up to a renaming of the constant symbols. In Section 4, it is established, again under suitable conditions, that for insertions, a minimal solution (in terms of change of information content) must be optimal. These conditions include in particular schemata constrained by a very wide class of dependencies called generalized Horn dependencies.

In summary, there are two conditions which must be met for optimality of a proposed update reflection u. First, it must be *tuple minimal*, in that there can be no other solution whose set of changes is a proper subset of those of u. Second, it must induce a *least change of information* relative to a specific set of sentences. This approach applies also to deletions and updates which involve both insertion and deletion, and this generality is incorporated into the formalism which is presented.

1.4 Further issues Despite the connection to database repair just presented, the primary focus of this paper is not to present yet another measure for such repairs, but rather to present a unified approach to the support of updates on traditional relational schemata which pays particular attention to the requirements of invariance of admissibility, canonicity of reflections, and reflection of monotonicity. Such an approach requires that careful choices be made regarding the class of schemata and views which are supported. Foremost, schemata or view mappings which allow disjunction preclude canonicity of reflections in most cases. For example, let $\mathbf{E}_2$ have the three unary relation symbols $R[A]$, $S[A]$, and $T[A]$, subject to the constraint $(\forall x)(T(x) \Leftrightarrow (R(x) \vee S(x)))$, and let Ω_T be the view which contains only $T[A]$. It is easy to see that canonicity of reflections can never be satisfied. Consider the database $M_{\text{old}} = \{R(\text{a}_0), S(\text{a}_0), T(\text{a}_0)\}$, with the insertion request $\mathsf{Insert}\langle T(\text{a}_1)\rangle$. A minimal solution would insert either $\{R(\text{a}_1), T(\text{a}_1)\}$ or else $\{(S(\text{a}_1), T(\text{a}_1)\}$; to insert $\{R(\text{a}_1), S(\text{a}_1), T(\text{a}_1)\}$ would not be minimal. For this reason, attention in this work is restricted to relational schemata which are restricted by Horn constraints, such as the XEIDs of Fagin [Fag82].

Unfortunately, even within contexts which involve at most functional dependencies (FDs) and projections, two sorts of problems may occur for insertions. First of all, an attempted reflection to the main schema of a view update may introduce new tuples in the main schema, called *orphan tuples*, whose images are visible in the view. This phenomenon is illustrated via a concrete example in 4.8. Secondly, an insertion to a given view instance may be possible for some instances of the main schema which map to it, but not others. This phenomenon is illustrated via a concrete example in 4.12. It is shown that these anomalies may be remedied by requiring that the view be *strongly monotonic* — that is, that every deletion to the view may be reflected as a deletion to the main schema, and every insertion to the view may be reflected as an insertion to the main schema. Simple conditions which guarantee strong monotonicity for projections of relations constrained by FDs and inclusion dependencies are developed.

At first thought, it might appear that the management of deletions would be simpler than that of insertions, since generic values need not be created. However, there is a quite different type of complication which arises. Specifically, Horn-style dependencies of the form $A_1 \wedge A_2 \wedge \ldots \wedge A_n \Rightarrow B$ are disjunction free with respect to insertions, but not deletions. Roughly, to delete B minimally, it is necessary to delete one of the A_i, but generally not all. One might therefore be lead to propose *weak optimal realizations*, as illustrated in the example of 6.4 and defined formally in 6.5, in which all of the contributing A_i's are deleted. Unfortunately, as illustrated in 6.9, it is not even possible in general to delete them all and obtain a correct solution. Therefore, attention is focused upon schemata whose tuple-generating dependencies are of the form $A \Rightarrow B$, with just one assertion in the head of the rule. Fortunately, even with such a restriction, many common situations, particularly schemata constrained by functional and inclusion dependencies, and views defined by projection, are supported.

The most difficult cases surround updates which involve both insertions and deletions. In general, the information-based approach forwarded here does not provide optimal solutions to such requests, and so that topic must remain a subject for future work.

This article is a based upon [Heg08a], but has been completely reworked to address some shortcomings in that preliminary version.

2 The Relational Model

The results of this paper are formulated within the relational model, and familiarity with its standard notions, as presented in references such as [PDGV89] and [AHV95], is assumed. Nevertheless, there are aspects which must be formulated with particular care. Most important are the need to take all relational schemata over the same domain, with the same constant symbols, and the need to express databases themselves as sets of ground atoms. For this reason, the central ideas which are special to this formulation are presented in this section.

2.1 Relational contexts and constant interpretations A relational context contains the logical information which is shared amongst the schemata and database mappings. Formally, a relational context $\mathcal{D}$ consists of a finite nonempty set $\mathcal{A}_\mathcal{D}$ of attribute names, a countable set $\mathsf{Vars}(\mathcal{D})$ of variables, and for each $A \in \mathcal{A}_\mathcal{D}$, a countable set $\mathsf{Const}_\mathcal{D}(A)$ of constant symbols, with $\mathsf{Const}(\mathcal{D}) = \bigcup\{\mathsf{Const}_\mathcal{D}(A) \mid A \in \mathcal{A}_\mathcal{D}\}$. The variables in $\mathsf{Vars}(\mathcal{D})$ are further partitioned into two disjoint sets; a countable set $\mathsf{GenVars}(\mathcal{D}) = \{x_0, x_1, x_2, \ldots\}$ of *general variables*, and special $\mathcal{A}_\mathcal{D}$-indexed set $\mathsf{AttrVars}(\mathcal{D}) = \{x_A \mid A \in \mathcal{A}_\mathcal{D}\}$ of *attribute variables*. The latter is used in the definition of interpretation mappings; see 2.6 for details. Lowercase letters at the end of the alphabet, such as v, w, x, y, and z, as well as subscripted instances using these names, will also be used as general variables.

Databases are represented as ground atoms, as elaborated in 2.2 below. Therefore, it is necessary that each domain element, in the sense of a logical structure for a first-order language, [Mon76, Def. 11.1], be bound to a unique constant symbol. Formally, a *constant interpretation* for the relational context $\mathcal{D}$ is a pair $\mathcal{I} = (\mathsf{Dom}_\mathcal{I}, \mathsf{IntFn}_\mathcal{I})$ in which $\mathsf{Dom}_\mathcal{I}$ is a countably infinite set, called the *domain* of $\mathcal{I}$, and $\mathsf{IntFn}_\mathcal{I} : \mathsf{Const}(\mathcal{D}) \to \mathsf{Dom}_\mathcal{I}$ is a bijective function, called the *interpretation function* of $\mathcal{I}$. This effectively stipulates the following two well-known conditions [GN87, p. 120]:

Domain closure: $(\forall x \in \mathsf{Vars}(\mathcal{D}))(\bigvee_{\mathrm{a} \in \mathsf{Const}(\mathcal{D})} x = \mathrm{a})$ $(\mathsf{DCA}(\mathcal{D}))$

Unique naming: $(\neg(a = b))$ for distinct $\mathrm{a}, \mathrm{b} \in \mathsf{Const}(\mathcal{D})$ $(\mathsf{UNA}(\mathcal{D}))$

Since there are countably many constant symbols, the domain-closure axiom is not a finite disjunction. This is not a problem however, since it is never used in a context in which a first-order constraint is necessary. Rather, the assignment of a constant to each variable is taken to be part of the context in which this work is carried out.

As a notational convention, from this point on, unless stated otherwise, fix a relational context $\mathcal{D}$ and a constant interpretation $\mathcal{I} = (\mathsf{Dom}_{\mathcal{I}}, \mathsf{IntFn}_{\mathcal{I}})$ for it.

2.2 Tuples and databases An *unconstrained relational schema* over $(\mathcal{D}, \mathcal{I})$ is a pair $\mathbf{D} = (\mathsf{Rels}(\mathbf{D}), \mathsf{Ar}_{\mathbf{D}})$ in which $\mathsf{Rels}(\mathbf{D})$ is a finite set of relational symbols and $\mathsf{Ar}_{\mathbf{D}} : \mathsf{Rels}(\mathbf{D}) \to \mathbf{2}^{\mathcal{A}_{\mathcal{D}}}$ a function which assigns an *arity*, a set of distinct attributes from $\mathcal{A}_{\mathcal{D}}$, to each $R \in \mathsf{Rels}(\mathbf{D})$.

An *R-atom* is a function $t : \mathsf{Ar}_{\mathbf{D}}(R) \to \mathsf{Const}(\mathcal{D}) \cup \mathsf{Vars}(\mathcal{D})$ with the property that $t[A] \in \mathsf{Const}_{\mathcal{D}}(A) \cup \mathsf{GenVars}(\mathcal{D}) \cup \{x_A\}$; in other words, all terms, constant and variable, are of the appropriate type. A *ground R-atom* has the additional property that it contains no variables; *i.e.*, $t[A] \in \mathsf{Const}_{\mathcal{D}}(A)$. The set of all R-atoms (resp. ground R-atoms) is denoted $\mathsf{Atoms}(R, \mathbf{D})$ (resp. $\mathsf{GrAtoms}(R, \mathbf{D})$).

A *$\mathbf{D}$-atom* is an R-atom for some $R \in \mathsf{Rels}(\mathbf{D})$; the set of all such atoms is denoted $\mathsf{Atoms}(\mathbf{D})$. A *ground atom* is defined to be a ground R-atom for some $R \in \mathsf{Rels}(\mathbf{D})$, with the set of all such atoms denoted $\mathsf{GrAtoms}(\mathbf{D})$. An *atom database for* $\mathbf{D}$ is a finite subset of $\mathsf{GrAtoms}(\mathbf{D})$, with the set of all atom databases for $\mathbf{D}$ denoted $\mathsf{DB}(\mathbf{D})$. In this work, ground atoms are also called *tuples*.

It is convenient to be able to recover the associated relation name from a tuple, and so *tagging* is employed, in which tuples are marked with the relation name. Formally, this is accomplished by introducing a new attribute $\mathsf{RName} \notin \mathcal{A}_{\mathcal{D}}$, and then regarding an R-atom not as a function t just on $\mathsf{Ar}_{\mathbf{D}}(R)$, but rather as one on $\{\mathsf{RName}\} \cup \mathsf{Ar}_{\mathbf{D}}(R)$ with the property that $t[\mathsf{RName}] = R$. Tagging of R-atoms will be used from this point on throughout the paper. Nevertheless, in writing such atoms, the more conventional notation $R(a_1, a_2, \ldots, a_n)$ will be used in lieu of the technically more correct $(R, a_1, a_2, \ldots, a_n)$, although tags will be used in formal constructions. To be completely pedantic, this entails introducing a new attribute name $\mathsf{RNames} \in \mathcal{A}_{\mathcal{D}}$ with $\mathsf{Const}_{\mathcal{D}}(\mathsf{RNames}) = \mathsf{Rels}(\mathbf{D})$, and these constant values used only for the attribute RNames. Furthermore, in any atom, the value for the RNames attribute must be a constant, never a variable. Since this is a logically inessential tactic whose full formal treatment is tedious but routine, the details will not be elaborated further.

There is a third type of atom which will be of use in defining constraints, the equality atom. Formally, an *equality atom* is of one of the forms $(x_i = x_j)$, $(x_i = \mathrm{a}_j)$, or $(\mathrm{a}_i = \mathrm{a}_j)$, for $x_i, x_j \in \mathsf{GenVars}(\mathcal{D})$ and $\mathrm{a}_i, \mathrm{a}_j \in \mathsf{Const}(\mathcal{D})$. The set of all equality $\mathcal{D}$-atoms is denoted $\mathsf{EqAtoms}(\mathcal{D})$. Equality atoms which equate two constants; *e.g.*, $(\mathrm{a}_i = \mathrm{a}_j)$ are called *ground equality atoms*; note that the truth value of such atoms is predetermined by the unique naming assumption. All other equality atoms; *e.g.*, those of the forms $(x_i = x_j)$ or $(x_i = \mathrm{a}_j)$, are called *variable equality atoms*. The set of all variable equality atoms is denoted $\mathsf{VarEqAtoms}(\mathcal{D})$. Note that the definitions of equality atoms depend only upon the relational context $\mathcal{D}$, and not upon the specific schema $\mathbf{D}$.

2.3 Formulas and constraint classes The first-order language associated with the relational schema $\mathbf{D}$ is defined in the natural way; however, it is useful to introduce some notation which identifies particular sets of formulas. Define $\mathsf{WFF}(\mathbf{D})$ to be the set of all well-formed first-order formulas with equality in the language whose set of relational symbols is $\mathsf{Rels}(\mathbf{D})$, whose set of constant symbols is $\mathsf{Const}(\mathcal{D})$, and which contains no non-nullary function symbols. The variables are those of $\mathcal{D}$; these formulas are typed to the extent that for $A \in \mathcal{A}_{\mathcal{D}}$, a term in a position of type A must lie in $\mathsf{GenVars}(\mathcal{D}) \cup \{x_A\} \cup \mathsf{Const}_{\mathcal{D}}(A)$.

A *constraint class* $\mathcal{C}$ identifies a subset of $\mathsf{WFF}(\mathbf{D})$, denoted $\mathsf{WFF}(\mathbf{D}, \mathcal{C})$. For this paper, the two most important constraint classes are $\exists\wedge+$ and $\mathsf{GrAtoms}$, defined as follows.

- $\mathsf{WFF}(\mathbf{D}, \exists\wedge+)$ is the subset of $\mathsf{WFF}(\mathbf{D})$ in which in which only existential quantification is allowed, and the only logical connective which is allowed is conjunction ($\wedge$). These formulas define the so-called *conjunctive queries* [CGT90, Sec. 4.2]. It is not necessary to allow the equality predicate in such formulas, since equality can always be expressed by simply using the same name for the two atoms which are equated.
- $\mathsf{WFF}(\mathbf{D}, \mathsf{GrAtoms})$ is just $\mathsf{GrAtoms}(\mathbf{D})$.

$\mathsf{WFF}(\mathbf{D}, \mathcal{C})$ may be trimmed further by limiting the constant symbols which may occur in it. Specifically, if $S \subseteq \mathsf{Const}(\mathcal{D})$, then $\mathsf{WFF}(\mathbf{D}, \mathcal{C}, S)$ denotes the formulas in $\mathsf{WFF}(\mathbf{D})$ which involve only constant symbols from S.

Each of these classes may be limited to sentences; *i.e.*, formulas without free variables. $\mathsf{WFS}(\mathbf{D})$
(resp. $\mathsf{WFS}(\mathbf{D}, \mathcal{C})$, resp. $\mathsf{WFS}(\mathbf{D}, \mathcal{C}, S)$) denotes the subset of $\mathsf{WFF}(\mathbf{D})$ (resp. $\mathsf{WFF}(\mathbf{D}, \mathcal{C})$, resp. $\mathsf{WFF}(\mathbf{D}, \mathcal{C}, S)$) consisting of sentences.

Let $\Phi \subseteq \Psi \subseteq \mathsf{WFS}(\mathbf{D})$. The *closure* of Φ in Ψ, denoted $\mathsf{Closure}\langle\Phi, \Psi\rangle$, is $\{\varphi \in \Psi \mid \Phi \models \varphi\}$. A *cover* for Φ relative to Ψ is a subset $\Phi' \subseteq \Phi$ with $\mathsf{Closure}\langle\Phi', \Psi\rangle = \mathsf{Closure}\langle\Phi, \Psi\rangle$. A *minimal cover* Ψ' has the property that none of its proper subsets is itself a cover.

Finally, the symbol $\bot$ will be used to denote the sentence which is always false.

2.4 Atomic models Even though databases are represented as sets of ground atoms, and not as interpretations in the usual logical sense, it is still essential to have an appropriate notion of model for a given sentence. This is relatively straightforward; a model for a sentence φ is a database which is consistent with both φ and the unique-naming axioms. There is one complication, however. In representing a database as a set of $\mathbf{D}$-atoms, the closed-world assumption is implicit. On the other hand, to express what it means for such a representation to satisfy an arbitrary sentence in $\mathsf{WFS}(\mathbf{D})$, it is necessary to state explicitly which atoms are not true as well. Formally, for $M \in \mathsf{DB}(\mathbf{D})$, define the *diagram* of M to be $\mathsf{Diagram}_{\mathbf{D}}(M) = M \cup \{\neg t \mid t \in \mathsf{GrAtoms}(\mathbf{D}) \setminus M\}$. Now, say that $M \in \mathsf{DB}(\mathbf{D})$ is an *atomic $\mathfrak{I}$-model* of $\varphi \in \mathsf{WFS}(\mathbf{D})$ if $\mathsf{Diagram}_{\mathbf{D}}(M) \cup \{\varphi\} \cup \mathsf{UNA}(\mathcal{D})$ is consistent. $\mathsf{AtMod}_{\mathfrak{I}}(\varphi)$ denotes the set of all atomic $\mathfrak{I}$-models of φ, with $\mathsf{AtMod}_{\mathfrak{I}}(\Phi) = \bigcap\{\mathsf{AtMod}_{\mathfrak{I}}(\varphi) \mid \varphi \in \Phi\}$ for $\Phi \subseteq \mathsf{WFS}(\mathbf{D})$. Since only atomic $\mathfrak{I}$-models will be considered in this paper, the simple term *model* will be used as a synonym for *atomic $\mathfrak{I}$-model*.

2.5 Schemata with constraints and constrained databases To obtain full relational schemata, constraints are added to the unconstrained schemata of 2.2. Formally, a *relational*

schema over $(\mathcal{D}, \mathcal{I})$ is a triple $\mathbf{D} = (\mathsf{Rels}(\mathbf{D}), \mathsf{Ar}_{\mathbf{D}}, \mathsf{Constr}(\mathbf{D}))$ in which $(\mathsf{Rels}(\mathbf{D}), \mathsf{Ar}_{\mathbf{D}})$ is an unconstrained relational schema over $(\mathcal{D}, \mathcal{I})$ and $\mathsf{Constr}(\mathbf{D}) \subseteq \mathsf{WFS}(\mathbf{D})$ is the set of *dependencies* or *constraints* of $\mathbf{D}$.

Define the *legal* (or *constrained*) *databases* $\mathsf{LDB}(\mathbf{D})$ of $\mathbf{D}$ to be $\mathsf{AtMod}_{\mathcal{I}}(\mathsf{Constr}(\mathbf{D}))$.

Although $\mathsf{Constr}(\mathbf{D})$ is allowed to be an infinite set, it will always be assumed that $\mathsf{Constr}(\mathbf{D})$ is *constant finite*; that is, that all of the sentences in $\mathsf{Constr}(\mathbf{D})$ together contain only a finite number of distinct constant symbols.

2.6 Database morphisms and views Let $\mathbf{D}_1$ and $\mathbf{D}_2$ be relational schemata over $(\mathcal{D}, \mathcal{I})$. There are two fundamental ways to represent a database morphism $f : \mathbf{D}_1 \to \mathbf{D}_2$ in the relational context. On the one hand, such a morphism may be represented as a function $f : \mathsf{DB}(\mathbf{D}_1) \to \mathsf{DB}(\mathbf{D}_2)$, using expressions from the relational algebra. On the other hand, by providing an interpretation formula $f^R \in \mathsf{WFF}(\mathbf{D}_1)$ for each $R \in \mathsf{Rels}(\mathbf{D}_2)$, the morphism may be represented using the relational calculus [JAK82]. The equivalence of these two representations is one of the classical results of relational database theory [PDGV89, Sec. 2.4-2.6]. The interpretation formulation is taken as the base definition for views in this work. Formally, given $R \in \mathsf{Rels}(\mathbf{D}_2)$, an *interpretation* for R into $\mathbf{D}_1$ is a $\varphi \in \mathsf{WFF}(\mathbf{D}_1)$ in which precisely the variables $\{x_A \mid A \in \mathsf{Ar}_{\mathbf{D}}(R)\}$ are free, and in which x_A is used to mark the position in the formula which is bound to attribute A. Since each position in the view relation is associated with a distinct attribute, one variable per attribute suffices. The set of all interpretations of R into $\mathbf{D}_1$ is denoted $\mathsf{Interp}(R, \mathbf{D}_1)$. A *syntactic morphism* $f : \mathbf{D}_1 \to \mathbf{D}_2$ is a family $f = \{f^R \mid R \in \mathsf{Rels}(\mathbf{D}_2) \text{ and } f^R \in \mathsf{Interp}(R, \mathbf{D}_1)\}$. The morphism f is said to be *of class* $\exists\wedge+$ if $f^R \in \mathsf{WFF}(\mathbf{D}, \exists\wedge+)$ for each $R \in \mathsf{Rels}(\mathbf{D}_2)$.

Let $t \in \mathsf{Atoms}(R, \mathbf{D}_2)$. The *substitution* of t into f, denoted $\mathsf{Substf}\langle f, t\rangle$, is the formula in $\mathsf{WFF}(\mathbf{D}_1)$ obtained by substituting, into f^R, $t[A]$ for x_A, for each $A \in \mathsf{Ar}_{\mathbf{D}}(R)$. Note that if t is a ground atom, then $\mathsf{Substf}\langle f, t\rangle \in \mathsf{WFS}(\mathbf{D}_1)$.

For $M \in \mathsf{DB}(\mathbf{D}_1)$, define $f(M) = \{t \in \mathsf{GrAtoms}(\mathbf{D}_2) \mid M \in \mathsf{AtMod}_{\mathcal{I}}(\mathsf{Substf}\langle f, t\rangle)\}$. f is called an LDB-*morphism* if it maps legal databases to legal databases; formally, an LDB-morphism has the property that $f(M) \in \mathsf{LDB}(\mathbf{D}_2)$ for each $M \in \mathsf{LDB}(\mathbf{D}_1)$. When no qualification is given, *database morphism* will always mean LDB-morphism.

Let $\mathbf{D}$ be a relational schema over $(\mathcal{D}, \mathcal{I})$. A *(relational) view* of $\mathbf{D}$ is a pair $\Gamma = (\mathbf{V}, \gamma)$ in which $\mathbf{V}$ is a relational schema over $(\mathcal{D}, \mathcal{I})$ and $\gamma : \mathbf{D} \to \mathbf{V}$ is an LDB-morphism which is furthermore LDB-*surjective* in the sense that for every $N \in \mathsf{LDB}(\mathbf{V})$, there is an $M \in \mathsf{LDB}(\mathbf{D})$ with $\gamma(M) = N$. Surjectivity is required because the instance of the view must always be determined by the instance of the main schema $\mathbf{D}$. The view $\Gamma = (\mathbf{V}, \gamma)$ is said to be *of class* $\exists\wedge+$ precisely in the case that γ has that property.

In order to illustrate these ideas, a simple example is in order. Consider again the schema $\mathbf{E}_0$ and the view $\Pi^{\mathbf{E}_0}_{AB}$ of 1.1. The view mapping $\pi^{\mathbf{E}_0}_{AB}$ is expressed as an interpretation via the formula $(\pi^{\mathbf{E}_0}_{AB})^{R_{AB}} = (\exists z)(R(x_A, x_B, z))$. Note in particular how x_A and x_B are used to mark the appropriate attributes. For $t = R_{AB}(\mathrm{a}_0, \mathrm{b}_0)$, $\mathsf{Substf}\langle \pi^{\mathbf{E}_0}_{AB}, t\rangle = (\exists z)(R(\mathrm{a}_0, \mathrm{b}_0, z))$, while for $t = R_{AB}(x_0, x_1)$, $\mathsf{Substf}\langle \pi^{\mathbf{E}_0}_{AB}, t\rangle = (\exists z)(R(x_0, x_1, z))$.

Occasionally, it will be useful to separate the quantifiers from the rest of the formula of an interpretation γ^R. To this end, define $\overline{\gamma}^R$ to be that which is left when the quantifier prefix is removed from γ^R. For example, in the above, $(\overline{\pi^{\mathbf{E}_0}_{AB}})^{R_{AB}} = (R(x_A, x_B, z))$.

2.7 Notation — extracting constant symbols and variables For X an entity (for example, an atom, a formula, a database, etc.), or a set of entities, $\mathsf{ConstSym}(X)$ denotes the set of all $a \in \mathsf{Const}(\mathcal{D})$ which occur in X. Furthermore, for $\mathbf{D}$ a database schema, $\mathsf{ConstSym}_{\mathbf{D}}(X) = \mathsf{ConstSym}(X \cup \mathsf{Constr}(\mathbf{D}))$, and for $\Gamma = (\mathbf{V}, \gamma)$ a view of $\mathbf{D}$, $\mathsf{ConstSym}_{\mathbf{\Gamma}}(X) = \mathsf{ConstSym}(X \cup \mathsf{Constr}(\mathbf{D}) \cup \mathsf{Constr}(\mathbf{V})) \cup \mathsf{ConstSym}(\gamma)$, where $\mathsf{ConstSym}(\gamma)$ is the set of all constant symbols which occur in the defining interpretation formulas associated with γ.

Similarly, $\mathsf{Vars}(X)$ denotes the set of all variables which occur in X. This will not be formalized further, but the meaning should always be unambiguous.

2.8 Notation for inclusion dependencies It is assumed that the reader is familiar with the relational model and the standard dependencies which have been studied in that context. Here only some notation and terminology is clarified. First, $R[X] \sqsubseteq S[Y]$ (note the squared subset symbol) will be used to denote the inclusion dependency (IND) which states that the projection onto attributes X of relation R is a subset of the projection onto attributes Y of relation S. Second, a *unary inclusion dependency*, abbreviated *UIND*, is one in which each of X and Y consist of a single attribute.

3 Information and Canonical Models

The theory of support for view updates which is forwarded in this paper is based upon a duality between a set of sentences defining information content and canonical models for such information. In this section, that duality is developed in detail.

3.1 Notational convention Throughout the rest of this paper, unless stated specifically to the contrary, take $\mathbf{D}$ to be a relational schema over $(\mathcal{D}, \mathcal{I})$. The notation Υ will be used as an abbreviation for $\mathsf{WFS}(\mathbf{D}, \exists\wedge+)$, and Υ_K will be used as an abbreviation for $\mathsf{WFS}(\mathbf{D}, \exists\wedge+, K)$. Furthermore, in the context of a set of the form $\mathsf{WFS}(\mathbf{D}, \exists\wedge+, K)$, if no further information is given, K will be taken to be an arbitrary subset of $\mathsf{Const}(\mathcal{D})$.

3.2 Information content and Φ-equivalence Let $\Phi \subseteq \mathsf{WFS}(\mathbf{D})$ and let $M \in \mathsf{DB}(\mathbf{D})$. The *information content* of M relative to Φ is the set of all sentences in Φ which are true for M. More precisely, $\mathsf{Info}\langle M, \Phi\rangle = \{\varphi \in \Phi \mid M \in \mathsf{AtMod}_{\mathcal{I}}(\varphi)\}$. For $\varphi \in \mathsf{WFS}(\mathbf{D})$, $\mathsf{Info}\langle M, \varphi\rangle$ denotes $\mathsf{Info}\langle M, \{\varphi\}\rangle$. M_1 and M_2 are Φ-*equivalent* if they have the same information content relative to Φ; *i.e.*, $\mathsf{Info}\langle M_1, \Phi\rangle = \mathsf{Info}\langle M_2, \Phi\rangle$.

The semantics of conventional databases are based upon the closed-world assumption — all assertions which cannot be established to be true are taken to be false. Thus, intuitively, information content should be monotone; that is, for any $M_1, M_2 \in \mathsf{DB}(\mathbf{D})$ if $M_1 \subseteq M_2$, then $\mathsf{Info}\langle M_1, \Phi\rangle \subseteq \mathsf{Info}\langle M_2, \Phi\rangle$. However, this is manifestly false for most choices of Φ. Indeed, if φ holds in M_2 but not M_1, then $\neg\varphi$ holds in M_1 but not M_2. Thus, if there is some $\varphi \in \Phi$ for which $\neg\varphi \in \Phi$ as well, Φ cannot be information monotone.

Formally, it is best to begin by defining monotonicity for individual sentences. To be precise, the sentence $\varphi \in \mathsf{WFS}(\mathbf{D})$ is *information monotone* if for any $M_1, M_2 \in \mathsf{DB}(\mathbf{D})$ if $M_1 \subseteq M_2$, then $\mathsf{Info}\langle M_1, \varphi\rangle \subseteq \mathsf{Info}\langle M_2, \varphi\rangle$. The set $\Phi \subseteq \mathsf{WFS}(\mathbf{D})$ is then said to be *information monotone* if each $\varphi \in \Phi$ has this property. It is easy to see that any $\varphi \in \mathsf{WFS}(\mathbf{D})$ which

does not involve negation, when expressed entirely in terms of the connectives $\wedge$, $\vee$, and $\neg$, is information monotone. Connectives such as $\Rightarrow$ involve negation implicitly, and so sentences which involve implication need not be (and usually are not) information monotone.

In this work, there are two families of information-monotone sentences which are of central importance. The first is $\mathsf{WFS}(\mathbf{D}, \mathsf{GrAtoms})$. It is easy to see that $\mathsf{Info}\langle M, \mathsf{WFS}(\mathbf{D}, \mathsf{GrAtoms})\rangle = M$ for any $M \in \mathsf{DB}(\mathbf{D})$, so that the information content of a database relative to $\mathsf{WFS}(\mathbf{D}, \mathsf{GrAtoms})$ is just that database itself. Although trivial in its characterization, this case is nonetheless important. The second important family of information monotone sentences is $\mathsf{WFS}(\mathbf{D}, \exists\wedge+, K)$ for a given $K \subseteq \mathsf{Const}(\mathcal{D})$, and is far less trivial in its characterization.

3.3 Tuple-minimal models Let $\Phi \subseteq \mathsf{WFS}(\mathbf{D})$. and let $M \in \mathsf{AtMod}_{\mathcal{I}}(\Phi)$. M is a *tuple-minimal model* of Φ if for any $M' \in \mathsf{AtMod}_{\mathcal{I}}(\Phi)$ with $M' \subseteq M$, it must be that $M' = M$. The set of all tuple-minimal models of Φ is denoted $\mathsf{MinAtMod}_{\mathcal{I}}(\Phi)$. For $\varphi \in \mathsf{WFS}(\mathbf{D})$, $\mathsf{MinAtMod}_{\mathcal{I}}(\varphi)$ is shorthand for $\mathsf{MinAtMod}_{\mathcal{I}}(\{\varphi\})$.

For $\Phi \subseteq \mathsf{WFS}(\mathbf{D})$ and $\varphi \in \mathsf{WFS}(\mathbf{D})$, say that Φ *minimally entails* φ, written $\Phi \models_{\mathsf{min}} \varphi$, if $\mathsf{MinAtMod}_{\mathcal{I}}(\Phi) \subseteq \mathsf{AtMod}_{\mathcal{I}}(\varphi)$. In other words, Φ minimally entails φ if every tuple-minimal model of Φ is also a model (not necessarily minimal) of φ.

3.4 Fully Reduced $\exists\wedge+$-families The concept of a minimal cover for a set Φ of sentences is well known and has already been recalled in 2.3. In the context of sentences in Υ, there is a stronger notion which is central to the development of the ideas presented here. To motivate this idea, let $\Xi = \{R(\mathrm{a}_1, \mathrm{a}_2), R(\mathrm{a}_2, \mathrm{a}_3), (\exists x_1)(\exists x_2)(\exists x_3)(R(x_1, x_2) \wedge R(x_2, \mathrm{a}_3) \wedge R(\mathrm{a}_3, x_3))\}$. It is easy to see that Ξ is a minimal cover of itself, in that none of its proper subsets is equivalent to it. However, it is also clear that $\xi_0 = (\exists x_1)(\exists x_2)(\exists x_3)(R(x_1, x_2) \wedge R(x_2, \mathrm{a}_3) \wedge R(\mathrm{a}_3, x_3))$ may be replaced with $(\exists x_3)(R(\mathrm{a}_3, x_3))$ while retaining logical equivalence. In other words, conjuncts may be removed from one of the sentences while preserving information content.

To formalize this notion, let $\varphi = (\exists x_1) \ldots (\exists x_m)(A_1 \wedge \ldots \wedge A_n) \in \Upsilon$ have at least two conjuncts, and for any i, $1 \leq i \leq n$, define $\mathsf{Reduction}\langle \varphi, A_i\rangle$ to be the sentence obtained by removing A_i as a conjunct from φ, and removing any quantifier term which is no longer used as well. For example, $\mathsf{Reduction}\langle \xi_0, R(x_1, x_2)\rangle = (\exists x_2)(\exists x_3)(R(x_2, \mathrm{a}_3) \wedge R(\mathrm{a}_3, x_3))$ and $\mathsf{Reduction}\langle \mathsf{Reduction}\langle \xi_0, R(x_1, x_2)\rangle, R(x_2, \mathrm{a}_3)\rangle = (\exists x_3)(R(\mathrm{a}_3, x_3))$. Call $\Phi \subseteq \Upsilon$ *conjunct reduced* if for no $\varphi \in \Phi$ with at least two conjuncts is there a conjunct A_i of φ with $(\Phi \setminus \varphi) \cup \{\mathsf{Reduction}\langle \varphi, A_i\rangle\}$ logically equivalent to Φ. Call Φ *fully reduced* if it is both conjunct reduced and a minimal cover of itself. In the above example, $\{R(\mathrm{a}_1, \mathrm{a}_2), R(\mathrm{a}_2, \mathrm{a}_3), (\exists x_3)(R(\mathrm{a}_3, x_3))\}$ is fully reduced.

The goal is to establish that by substituting distinct constants for each variable in a finite, fully reduced family of sentences, a canonical model of those sentences is obtained. Thus, in the above example, $\{R(\mathrm{a}_1, \mathrm{a}_2), R(\mathrm{a}_2, \mathrm{a}_3), (R(\mathrm{a}_3, \mathrm{b}_1))\}$ would be such a model, with b_1 a "generic" constant. To render all of this formal, some additional notions are necessary.

3.5 Armstrong models in an information-monotone context Let $\Psi \subseteq \mathsf{WFS}(\mathbf{D})$ and let $\Phi \subseteq \Psi$. Informally, an Armstrong model for Φ relative to Ψ is a model of Φ which satisfies only those constraints of Ψ which are implied by Φ. More formally, an *Armstrong model* for Φ relative to Ψ is an $M \in \mathsf{AtMod}_{\mathcal{I}}(\Phi)$ with the property that for any $\psi \in \Psi$, if $M \in \mathsf{AtMod}_{\mathcal{I}}(\psi)$,

then $\mathsf{AtMod}_{\mathcal{I}}(\Phi) \subseteq \mathsf{AtMod}_{\mathcal{I}}(\psi)$. A *tuple-minimal Armstrong model* for Φ relative to Ψ is an Armstrong model with the property that no proper subset is an Armstrong model for Φ relative to Ψ. In general, a tuple-minimal Armstrong model M of Φ relative to Ψ need not be a tuple-minimal model of Φ, since there may be an $M' \subsetneq M$ which is a non-Armstrong model of Φ. However, if Ψ is information monotone, it is easy to see that this cannot happen, so every tuple-minimal Armstrong model must in fact be a minimal model. Armstrong models have been studied extensively for database dependencies; see, for example, [Fag82] and [FV83].

In the current context, for a given finite, fully reduced set $\Phi \subseteq \Upsilon$, a suitably constructed Armstrong model for Φ relative to Υ_K for a given K with $\mathsf{ConstSym}(\Phi) \subseteq K$ will serve as a canonical representation for insertions with generic constants, as sketched in the introduction. To proceed further, a special representation is useful.

3.6 Representation of $\exists\wedge+$-sentences as sets of D-atoms There is an alternative syntactic representation for formulas in Υ which will be used in that which follows. Specifically, for $\varphi \in \Upsilon$ define $\mathsf{AtRep}(\varphi)$ to be the set of all atoms which occur as conjuncts in φ. For example, if $\varphi = (\exists x_1)(\exists x_2)(\exists x_3)(R(x_1, \mathrm{a}) \wedge R(x_1, \mathrm{b}) \wedge S(x_2, a) \wedge T(x_2, x_3))$ then $\mathsf{AtRep}(\varphi) = \{R(x_1, \mathrm{a}), R(x_1, \mathrm{b}), S(x_2, \mathrm{a}), T(x_2, x_3)\}$.

This representation is dual to that used in theorem-proving contexts in classical artificial intelligence [GN87, 4.1]. Here the variables are existentially quantified and the atoms are conjuncts of one another; in the AI setting the atoms are disjuncts of one another and the variables are universally quantified.

3.7 Substitutions Let $V = \{x_1, x_2, \ldots, x_n\} \subseteq \mathsf{GenVars}(\mathcal{D})$. A *substitution* for V (in $\mathcal{D}$) is a function $s : V \to \mathsf{Const}(\mathcal{D}) \cup \mathsf{GenVars}(\mathcal{D})$. If $s(x_i) = \tau_i$ for $i \in \{1, 2, \ldots, n\}$, following (somewhat) standard notation this substitution is $\{\tau_1/x_1, \tau_2/x_2, \ldots, \tau_n/x_n\}$ [CL73, Sec. 5.3] and will be used here, although the reader is cautioned that some authors write $\{x_1/\tau_1, x_2/\tau_2, \ldots, x_n/\tau_n\}$ instead [GN87, 4.2].

Let $\varphi \in \Upsilon$ with $\mathsf{Vars}(\varphi) \subseteq V$. Call s *correctly typed* for φ if for each $t \in \mathsf{AtRep}(\varphi)$ and each $A \in \mathsf{Ar}_{\mathbf{D}}(t[\mathsf{RName}])$, if $t[A] \in \mathsf{Vars}(\mathbf{D})$ then $s(t[A]) \in \mathsf{Const}_{\mathcal{D}}(A) \cup \mathsf{GenVars}(\mathcal{D})$. Define $\mathsf{Subst}(\varphi, s)$ to be the set of atoms obtained by substituting $s(x_i)$ for x_i in $\mathsf{AtRep}(\varphi)$. For example, with $s = \{\mathrm{a}_1/x_1, \mathrm{a}_2/x_2, \mathrm{a}_3/x_3\}$ and $\mathsf{AtRep}(\varphi) = \{R(x_1, \mathrm{a}), R(x_1, \mathrm{b}), S(x_2, \mathrm{a}), T(x_2, x_3)\}$, $\mathsf{Subst}(\varphi, s) = \{R(a_1, \mathrm{a}), R(a_1, b), S(\mathrm{a}_2, \mathrm{a}), T(\mathrm{a}_2, \mathrm{a}_3)\}$.

If $s(x_i) \in \mathsf{Const}(\mathcal{D})$ for each $x_i \in V$, s is called a *constant substitution*. In this case, $\mathsf{Subst}(\varphi, s)$ is a set of ground atoms.

Now let $\Phi \subseteq \Upsilon$ be a finite set. A *constant substitution set* for Φ is a Φ-indexed set $S = \{s_\varphi \mid \varphi \in \Phi\}$ of substitutions, with s_φ a constant substitution for $\mathsf{Vars}(\varphi)$. For K a finite set with $\mathsf{ConstSym}(\Phi) \subseteq K \subseteq \mathsf{Const}(\mathcal{D})$, S is *free for* $\langle \Phi, K \rangle$ if each s_φ is correctly typed for φ, injective, $s_\varphi(x_i) \notin K$ for any $\varphi \in \Phi$ and $x_i \in V$, and, furthermore, for any distinct $\varphi_1, \varphi_2 \in \Phi$, $s_{\varphi_1}(\mathsf{Vars}(\varphi_1)) \cap s_{\varphi_2}(\mathsf{Vars}(\varphi_2)) = \emptyset$.

With S free for $\langle \Phi, K \rangle$, the *Armstrong model defined by* $\langle \Phi, K, S \rangle$ is obtained by applying the substitution s_φ to φ for each $\varphi \in \Phi$. Formally, $\mathsf{ArmMod}\langle \Phi, K, S \rangle = \bigcup \{\mathsf{Subst}(\varphi, s_\varphi) \mid \varphi \in \Phi\}$. Of course, this terminology is a bit presumptuous, as it has not yet been established that $\mathsf{ArmMod}\langle \Phi, K, S \rangle$ is in fact an Armstrong model of anything; this will be rectified in 3.9 below.

3.8 Constant endomorphisms Informally, an endomorphism on $\mathcal{D}$ is a function which renames constants. More formally, an *endomorphism* on $\mathcal{D}$ is a function $h : \mathsf{Const}(\mathcal{D}) \to \mathsf{Const}(\mathcal{D})$ which preserves attribute types, in the precise sense that for each $A \in \mathcal{A}_\mathcal{D}$ and each a $\in \mathsf{Const}_\mathcal{D}(A)$, $h(\mathrm{a}) \in \mathsf{Const}_\mathcal{D}(A)$. If h is additionally a bijection, then it is called an *automorphism* of $\mathcal{D}$. For $K \subseteq \mathsf{Const}(\mathcal{D})$, call h *K-invariant* if $h(\mathrm{a}) = \mathrm{a}$ for all a $\in K$.

Given a database schema $\mathbf{D}$, an endomorphism on $\mathcal{D}$ induces a mapping from $\mathsf{GrAtoms}(\mathbf{D})$ to itself given by sending $t \in \mathsf{GrAtoms}(\mathbf{D})$ to the tuple t' with $t'[\mathsf{RName}] = t[\mathsf{RName}]$ and $t'[A] = t[h(A)]$ for all $A \in \mathsf{Ar}_{\mathbf{t}[\mathsf{RName}]}$. This mapping on atoms will also be represented by h, as will the induced mapping from $\mathsf{DB}(\mathbf{D})$ to itself given by $M \mapsto \{h(t) \mid t \in M\}$.

The following theorem establishes that $\mathsf{ArmMod}\langle\Phi, K, S\rangle$ is a weak sort of initial model for Φ, in the sense that for any other database M which satisfies Φ, there is an endomorphism $h : \mathsf{ArmMod}\langle\Phi, K, S\rangle \to M$ which holds K constant. On the other hand, it is not an initial model for Φ in the traditional categorical sense [HS73, §7], since h need not be unique.

3.9 Theorem — Characterization of tuple-minimal Armstrong models *Let $\Phi \subseteq \Upsilon$ be finite and fully reduced, let K be a finite set with $\mathsf{ConstSym}(\Phi) \subseteq K \subseteq \mathsf{Const}(\mathcal{D})$, and let S be a constant substitution set which is free for Φ. Then the following hold.*

(a) *For any $M \in \mathsf{DB}(\mathbf{D}) \cap \mathsf{AtMod}_\mathfrak{I}(\Phi)$, there is a K-invariant endomorphism h on $\mathcal{D}$ with $h(\mathsf{ArmMod}\langle\Phi, K, S\rangle) \subseteq M$.*

(b) *$\mathsf{ArmMod}\langle\Phi, K, S\rangle$ is a tuple-minimal Armstrong model for Φ relative to Υ_K.*

(c) *If $M \in \mathsf{DB}(\mathbf{D})$ is any other tuple-minimal Armstrong model for Φ relative to Υ_K, then there is a $\mathsf{ConstSym}(\Phi)$-invariant automorphism h on $\mathcal{D}$ with $h(\mathsf{ArmMod}\langle\Phi, K, S\rangle) = M$.*

PROOF: To establish (a), let $M \in \mathsf{Mod}_\mathfrak{I}(\Phi)$, and for each $\varphi \in \Phi$, let M_φ be a minimal subset of M with $M_\varphi \in \mathsf{Mod}_\mathfrak{I}(\varphi)$. Let V_φ denote the set of variables of $s_\varphi \in S$. It is easy to see that there must be a constant substitution s'' with $\mathsf{Vars}(s'') = V_\varphi$ and $\mathsf{Subst}(\varphi, s'') = M_\varphi$. Indeed, there is trivially a constant substitution with $\mathsf{Subst}(\varphi, s'') \subseteq M_\varphi$, but if the subset inclusion were proper, M_φ would not be tuple minimal.

Now define $h : s_\varphi(V_\varphi) \to s''(V_\varphi)$ by $\mathrm{a} \mapsto s''(s_\varphi^{-1}(\mathrm{a}))$. Since s_φ is injective, h is well defined. Since $s_{\varphi_1}(\mathsf{Vars}(\varphi_1)) \cap s_{\varphi_2}(\mathsf{Vars}(\varphi_2)) = \emptyset$ for distinct $\varphi_1, \varphi_2 \in \Phi$, there are no conflicts in this definition of h. Finally, extend h to be the identity on all a $\in \mathsf{Const}(\mathcal{D})$ which are not covered by the above definition. The result is a endomorphism on $\mathcal{D}$ which satisfies $h(\mathsf{ArmMod}\langle\Phi, K, S\rangle) \subseteq M$.

For (b), first observe that $\mathsf{ArmMod}\langle\Phi, K, S\rangle$ is a model of Φ just by construction. It is furthermore easy to see that since Φ is fully reduced, it is tuple minimal. Indeed, if any tuple $t \in \mathsf{ArmMod}\langle\Phi, K, S\rangle$ could be removed, then the corresponding conjunct could be removed from the $\varphi \in \Phi$ associated with t, contradicting the fact that Φ is fully reduced. To show that $\mathsf{ArmMod}\langle\Phi, K, S\rangle$ is an Armstrong model, let $\psi \in \Upsilon$ with $\mathsf{ArmMod}\langle\Phi, K, S\rangle \in \mathsf{Mod}_\mathfrak{I}(\psi)$, and let $M \in \mathsf{Mod}_\mathfrak{I}(\Phi)$. In view of (a), there is an endomorphism h on $\mathcal{D}$ with $h(\mathsf{ArmMod}\langle\Phi, K, S\rangle) \subseteq M$. In view of Lyndon's theorem [Mon76, Thm. 25.22], which states that satisfaction of sentences not involving negation is closed under endomorphic images, it follows that $M \in \mathsf{Mod}_\mathfrak{I}(\psi)$ also. Hence, $\Phi \models \psi$, and so $\mathsf{ArmMod}\langle\Phi, K, S\rangle$ is an Armstrong model of Φ.

To show (c), let M be any other tuple-minimal Armstrong model for Φ relative to Υ_K. In the above construction for the proof of (a), the resulting h must be surjective (else M would

not be tuple minimal), and it must be injective (since there must also be an endomorphism in the opposite direction, and both $\mathsf{ArmMod}\langle\Phi, K, S\rangle$ and M are finite, by assumption). Hence, h is an automorphism. □

It is easy to see that the endomorphism h need not be unique. For example, if $\mathbf{D}$ has the single unary relation symbol $R[A]$, and $\Phi = \{(\exists x)(R(x))\}$, then $M_1 = \{R(\mathrm{a})\}$ is a minimal Armstrong model, while $M_2\{R(\mathrm{b}), R(\mathrm{c})\}$ is an Armstrong model which is not tuple minimal. There are two endomorphisms from M_1 to M_2, $h_1 : \mathrm{a} \mapsto \mathrm{b}$ and $h_2 : \mathrm{a} \mapsto \mathrm{c}$.

In some ways, the construction given above is similar to the construction of the universal solutions of [FKMP05, Def. 2.4], in that both are based upon similar notions of endomorphism (there termed *homomorphism*). However, those universal solutions are not required to be tuple minimal. On the other hand, they are not limited to positive sentences, but rather apply to the more general class of XEIDs [Fag82].

The existence of a (finite) Armstrong model for a set of Φ is guaranteed under fairly simple circumstances; all that is necessary is that Φ have a finite cover.

3.10 Lemma *Let $\Phi \subseteq \Upsilon_K$. Then Φ admits a tuple-minimal Armstrong model with respect to Υ_K iff Φ admits a finite cover relative to Υ_K.*

PROOF: If Φ admits a finite cover, then Φ admits a tuple-minimal Armstrong model with respect to Υ_K by 3.9(b). Conversely, if Φ is not finite and has no finite cover, then for any positive integer n, there is a $\varphi \in \Phi$ which contains at least n distinct conjuncts and which is not equivalent to any finite subset of Υ_K, each of whose elements contains fewer than n conjuncts. An Armstrong model must thus contain at least n tuples. Since n may be chosen arbitrarily large, it follows that no such finite model can exist. □

3.11 Canonical models Let K be a finite subset of $\mathsf{Const}(\mathcal{D})$. In (a)-(c) and (e) below, take $\Phi \subseteq \Upsilon_K$ as well.

(a) Φ is $\mathbf{D}$-*consistent* if $\mathsf{AtMod}_{\mathcal{J}}(\Phi) \cup \mathsf{LDB}(\mathbf{D}) \neq \emptyset$.

Thus, Φ is $\mathbf{D}$-consistent if there is some legal database which satisfies Φ. Such a database must also satisfy the sentences in $\mathsf{Constr}(\mathbf{D})$; the total set of sentences which it must satisfy is the extended information, expressed formally as follows.

(b) Define the *extended information* of Φ with respect to Υ_K to be $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi, \Upsilon_K\rangle = \{\varphi \in \Upsilon_K \mid \Phi \cup \mathsf{Constr}(\mathbf{D}) \models \varphi\}$.

Note that if Φ is not $\mathbf{D}$-consistent, then $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi, \Upsilon_K\rangle = \Upsilon_K$. Also note that, equivalently, $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi, \Upsilon_K\rangle = \{\varphi \in \Upsilon_K \mid (\forall M \in \mathsf{LDB}(\mathbf{D}))((M \in \mathsf{AtMod}_{\mathcal{J}}(\Phi)) \Rightarrow (M \in \mathsf{AtMod}_{\mathcal{J}}(\varphi)))\}$ whenever Φ is $\mathbf{D}$-consistent. In other words, $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi, \Upsilon_K\rangle$ is the set of all sentences in Υ_K which are true in every $M \in \mathsf{LDB}(\mathbf{D}) \cap \mathsf{AtMod}_{\mathcal{J}}(\Phi)$.

Since the databases of this paper are finite, consistency is not enough. Rather, Φ together with $\mathsf{Constr}(\mathbf{D})$ must admit a finite model. In view of 3.10, this property is equivalent to Φ having a finite cover. Formally, this is recaptured as follows.

(c) Φ *extends finitely* to $\mathbf{D}$ with respect to Υ_K if $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi, \Upsilon_K\rangle$ has a finite cover with respect to Υ_K.

(d) The schema $\mathbf{D}$ *admits finite extensions* with respect to Υ_K if every finite and $\mathbf{D}$-consistent $\Phi \subseteq \Upsilon_K$ extends finitely to $\mathbf{D}$ with respect to Υ_K.

(e) A *canonical database* for Φ in $\mathbf{D}$ with respect to Υ_K is a tuple-minimal Armstrong model M for $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi, \Upsilon_K\rangle$ with respect to Υ_K.

Observe that, in view of 3.9(c), canonical databases are unique up to automorphism. $\mathbf{D}$ admits canonical databases conditionally if there is a canonical model whenever the extended information is finite, with unconditional extension requiring further that this finiteness condition always be satisfied.

(f) The schema $\mathbf{D}$ *admits canonical models conditionally* with respect to Υ_K if for every $\Phi \subseteq \Upsilon_K$ which extends finitely to $\mathbf{D}$ with respect to Υ_K, every canonical database with respect to Υ_K is in $\mathsf{LDB}(\mathbf{D})$.

(g) The schema $\mathbf{D}$ *admits canonical models unconditionally* if $\mathbf{D}$ admits finite extensions with respect to Υ_K and every canonical database with respect to Υ_K is in $\mathsf{LDB}(\mathbf{D})$.

These existence conditions are characterized precisely in the following lemma.

3.12 Lemma *Continue with the notation of 3.11 above.*

(a) $\mathbf{D}$ *admits canonical models conditionally with respect to* Υ_K *iff for every* $\mathbf{D}$*-consistent* $\Phi \subseteq \Upsilon_K$, $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi, \Upsilon_K\rangle \models_{\mathsf{min}} \varphi$ *for every* $\varphi \in \mathsf{Constr}(\mathbf{D})$,

(b) $\mathbf{D}$ *admits canonical models unconditionally with respect to* Υ_K *iff it admits canonical models conditionally and* $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi, \Upsilon_K\rangle$ *has a finite cover relative to* Υ_K.

PROOF: Both parts follow immediately from 3.10. □

3.13 Example — Canonical models conditionally but not unconditionally It is not the case that every schema which admits canonical models conditionally admits them unconditionally. For example, let the schema $\mathbf{E}_2$ have three relational symbols $R_1[A]$, $R_2[AB]$, and $R_3[AB]$, with the inclusion dependencies $R_1[A] \subseteq R_2[A]$, $R_2[A] \subseteq R_3[A]$, and $R_3[B] \subseteq R_2[B]$. Let $M_1 = \{R_1(\mathrm{a}_0), R_2(\mathrm{a}_0, \mathrm{b}_0), R_3(\mathrm{a}_0, \mathrm{b}_0), R_1(\mathrm{a}_1)\}$, let $K = \{\mathrm{a}_0, \mathrm{a}_1, \mathrm{b}_0\}$, and note that $M_1 \subseteq \mathsf{WFS}(\mathbf{E}_3, \exists\wedge+, K)$, since databases are taken to be sets of ground atoms.

In $\mathsf{XInfo}_{\mathbf{E}_3}\langle M_1, \mathsf{WFS}(\mathbf{E}_3, \exists\wedge+, K)\rangle$, a tuple of the form $R_2(\mathrm{a}_1, \mathrm{b}_1)$ must be present, which implies that one of the form $R_3(\mathrm{a}_2, \mathrm{b}_1)$ must be present as well, which in turn implies that one of the form $R_2(\mathrm{a}_2, \mathrm{b}_2)$ must be present, and so forth. If the constant symbols which are introduced to satisfy the dependencies, are always new ones which have not been used previously, then this construction proceeds indefinitely. In terms of the construction of the extended information $\mathsf{XInfo}_{\mathbf{D}}\langle M_1, \mathsf{WFS}(\mathbf{E}_3, \exists\wedge+, K)\rangle$, it is not difficult to see that an infinite increasing sequence $\langle\varphi_0, \varphi_1, \varphi_2, \ldots, \varphi_i, \ldots\rangle$ of sentences arises, as shown in Fig. 1, with φ_{i+1} strictly longer than φ_i and furthermore not a consequence of $\{\varphi_0, \varphi_1, \ldots, \varphi_i\}$. Thus, $\mathsf{XInfo}_{\mathbf{E}_3}\langle M_1, \mathsf{WFS}(\mathbf{E}_3, \exists\wedge+, K)\rangle$ cannot have a finite cover. If the sequence is terminated, by choosing, say, $b_2 = b_1$, then an additional sentence beyond those in $\mathsf{XInfo}_{\mathbf{E}_2}\langle M_1, \mathsf{WFS}(\mathbf{E}_2, \exists\wedge+, K)\rangle$ is included, and so the resulting database is not Armstrong with respect to $\mathsf{WFS}(\mathbf{E}_3, \exists\wedge+, K)$. On the other hand, the database $M_2 = \{R_1(\mathrm{a}_0), R_2(\mathrm{a}_0, \mathrm{b}_0), R_3(\mathrm{a}_0, \mathrm{b}_0)\}$ already satisfies every constraint in $\mathsf{XInfo}_{\mathbf{E}_3}\langle M_2, \mathsf{WFS}(\mathbf{E}_2, \exists\wedge+, K)\rangle$, and so is a

$$(\exists x_1)(R_2(\mathrm{a}_1, x_1))$$
$$(\exists x_1)(\exists x_2)(R_2(\mathrm{a}_1, x_1) \wedge R_3(x_1, x_2))$$
$$(\exists x_1)(\exists x_2)(\exists x_3)(R_2(\mathrm{a}_1, x_1) \wedge R_3(x_1, x_2) \wedge R_2(x_3, x_2))$$
$$(\exists x_1)(\exists x_2)(\exists x_3)(\exists x_4)(R_2(\mathrm{a}_1, x_1) \wedge R_3(x_1, x_2) \wedge R_2(x_3, x_2) \wedge R_3(x_3, x_4))$$
$$\vdots$$

Figure 1: A strictly increasing sequence of information sentences

tuple-minimal Armstrong model of itself. Thus, the terminology *conditionally* is justified; $\mathbf{E}_2$ admits canonical models for some sets of sentences, but not for others. In 3.22, conditions under which canonical models are always admitted unconditionally are identified.

3.14 Example — Canonical models and positive disjunction While the definitions of 3.11 apply to any relational database schema, further restrictions must be imposed to render them meaningful. Consider again the schema $\mathbf{E}_2$ of Section 1, with the three unary relation symbols $R[A]$, $S[A]$, and $T[A]$, subject to the constraint $(\forall x)(R(x) \Leftrightarrow (S(x) \vee T(x)))$. For $M_1 = \{R(\mathrm{a}_0)\}$ and $K = \{\mathrm{a}_0\}$, $\mathsf{XInfo}_{\mathbf{E}_2}\langle M_1, \mathsf{WFS}(\mathbf{E}_2, \exists\wedge+, K)\rangle = M_1$, yet $M_1 \notin \mathsf{LDB}(\mathbf{E}_2)$. The problem is that the "full" extended information, relative to $\mathsf{WFS}(\mathbf{E}_2)$ is $\{R(\mathrm{a}_0), S(\mathrm{a}_0) \vee T(\mathrm{a}_0)\}$, but the disjunction $S(\mathrm{a}_0) \vee T(\mathrm{a}_0)$ does not lie in $\mathsf{WFS}(\mathbf{E}_2, \exists\wedge+)$. Hence the canonical database for M_1 with respect to $\mathsf{WFS}(\mathbf{E}_2, \exists\wedge+)$ is not in $\mathsf{LDB}(\mathbf{E}_2)$. It is clear that the notion of a canonical database is not really meaningful in the presence of such disjunctions. Rather, attention must be restricted to Horn dependencies, which avoid such positive disjunction and which are described below.

3.15 Generalized Horn dependencies The vast majority of relational database dependencies which have been considered over the years belong to a general class of logical formulas called *Horn clauses*. Originally presented as a characterization of formulas which are true under direct products [Hor51], they are more generally central to the modelling canonical instances in computer science [Mak87]. In the context of database dependencies, the following form is used, with each A_i and each B_i an atom.

$$\text{(GHD)} \qquad (\forall x_1)(\forall x_2)\ldots(\forall x_m)((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \Rightarrow (\exists y_1)(\exists y_2)\ldots(\exists y_r)(B_1 \wedge B_2 \wedge \ldots \wedge B_s))$$

In this work, such dependencies will be allowed in their most general form, which will be called *generalized Horn dependencies*, or *GHDs*. The only restrictions are the following.

(ghd-1) Each GHD is in fact a sentence, so that each variable lies within the scope of a quantifier.

(ghd-2) $\{x_1, x_2, \ldots, x_m\} \cap \{y_1, y_2, \ldots, y_r\} = \emptyset$.

(ghd-3) Each x_i occurs in some A_j; no universally quantified variable occurs only in a B_j.

(ghd-4) $A_i \in \mathsf{Atoms}(\mathbf{D})$ for each i.

(ghd-5) If $s > 0$, then either each $B_i \in \mathsf{Atoms}(\mathbf{D})$, in which the sentence is *tuple generating* (also called a *TGHD*), or else $s = 1$, $r = 0$, and $B_1 \in \mathsf{VarEqAtoms}(\mathcal{D}) \cup \{\bot\}$, in which case

the sentence is called *equality generating* (also called an *EGHD*). If $B_1 = \bot$, the sentence is called a *mutual exclusion*. ($\bot$ may be thought of as an equality which can never be satisfied, such as (a = b).)

As a convenient notation, $\mathsf{GHD}(\mathbf{D})$ will be used to denote the set of all GHDs on $\mathbf{D}$, with $\mathsf{TGHD}(\mathbf{D})$ (resp. $\mathsf{EGHD}(\mathbf{D})$) the subset consisting of the tuple-generating (resp. equality-generating) sentences. Mutual exclusions will be regarded as special cases of EGHDs in which there is no atom on the right-hand side.

The GHDs are a generalization of the XEIDs of Fagin [Fag82, Sec. 7], and are essentially the source-to-target dependencies of [FKMP05, Def. 2.1]. As such, the GHDs encompass virtually all classes of database constraints which have been studied, including in particular functional and inclusion dependencies. In contrast to XEIDs, the left-hand side need be neither unirelational nor typed. Of course, the more stringent requirement on XEIDs is made for a reason — XEIDs enjoy the property of possessing Armstrong models [Fag82, Thm. 3.1] which GHDs do not. Although they were used as the general class of dependency in [Heg08a], it turns out that this property of possessing Armstrong models is only required of sentences in $\mathsf{WFS}(\mathbf{D}, \exists\wedge+)$ which are used to characterize the information content of database, and not for more general Horn clauses which are used to characterize the underlying constraints. Therefore, there is no need to enforce the additional requirements of XEIDs.

GHDs also generalize XEIDs in a less essential way — there is no requirement that n be greater than zero, although both are not allowed to be zero in the same clause. If $n = 0$, a sentence in $\mathsf{WFS}(\mathbf{D}, \exists\wedge+)$ is obtained; thus, $\mathsf{WFS}(\mathbf{D}, \exists\wedge+) \subseteq \mathsf{GHD}(\mathbf{D})$. As a specific example of such a constraint, consider $(\exists y_1)(\exists y_2)(R(y_1, y_2))$, which states that the relation instance for R is always nonempty. Additionally, constant symbols are allowed in a GHD. For example, a constraint of the form $(\forall x_1)(\forall x_2)(R(x_1, x_2) \Rightarrow S(x_1, x_2, \nu)$ might assert that for every tuple in R, a similar tuple, padded with the constant ν in the third position (with ν representing a null value, for example), is required. Finally, an unsatisfiable right-hand side is allowed. If $B_1 = \bot$, a statement of the form $(\forall x_1)(\forall x_2)\ldots(\forall x_m)((A_1\wedge A_2\wedge\ldots\wedge A_n) \Rightarrow \bot)$ is obtained, with $\bot$ denoting the identically false assertion. As noted in (ghd-5) above, such as sentence is called a mutual exclusion. An example is the antisymmetry constraint $(\forall x_1)(\forall x_2)((R(x_1, x_2)\wedge R(x_2, x_1)) \Rightarrow \bot$.

The idea of forward chaining on databases by applying Horn-style rules is well known [DG84], and forms one of the cornerstones of logic programming [Llo87]. However, in the current context, rather than reasoning on ground atoms, it is essential to apply forward chaining on information — that is, on sentences in Υ_K. This idea is addressed via notions of information associated with TGHDs, as developed below.

3.16 Information inference for TGHDs In so-called forward chaining in classical propositional logic, given a Horn clause of the form $A_1\wedge A_2\wedge\ldots\wedge A_n \Rightarrow B$, if all of the A_i's are known to be true, the rule may "fire" and assert that B is also true. This idea also works for TGHDs operating on ground atoms; a ground substitution is applied to the left-hand side, and if all of the resulting ground atoms are true, the sentence obtained by applying the same substitution to the right-hand side must be true, and further substitutions may then be applied to identify the ground atoms which it implies. More generally, however, when the pool of knowledge consists of sentences in $\mathsf{WFS}(\mathbf{D}, \exists\wedge+)$, the left-hand side and right-hand sides are coupled via variables. In other words, a rule fires for specific bindings of variable common to both sides. It is therefore

not possible to separate the conclusion (*i.e.*, the right-hand side) from the hypotheses (*i.e.*, the left-hand side). The solution is a rather simple one. When a rule of the above form fires, rather than concluding simply B from $A_1 \wedge A_2 \wedge \ldots \wedge A_n$, the conjunction $A_1 \wedge A_2 \wedge \ldots \wedge A_n \wedge B$ is deduced as a new fact. In this way, any binding of quantified variables which occurred during the inference process are preserved.

Let φ be a GHD of the form (GHD) of 3.15 above, and let s be a substitution on $\{x_1, x_2, \ldots, x_n\}$. Call s *GHD-compatible with* φ if $s(x_i) \neq y_j$ for any indices i and j, and assume that s has this property. GHD compatibility ensures only that a substitution does not rename a universally quantified variable to coincide with one which is existentially quantified in the original formula. The *left-hand-side information* of φ with respect to s is the sentence obtained by applying s to the left-hand side of φ. Formally, $\mathsf{LHSinfo}\langle\varphi, s\rangle$ is the sentence in $\mathsf{WFS}(\mathbf{D}, \exists\wedge+)$ obtained from $(\forall x_1)(\forall x_2)\ldots(\forall x_m)(A_1 \wedge A_2 \wedge \ldots \wedge A_n)$ as follows.

(lhs-i) For $i \in \{1, 2, \ldots, m\}$, if $s(x_i) = v$ and $v \in \mathsf{GenVars}(\mathcal{D})$, replace $(\forall x_i)$ with $(\exists v)$.

(lhs-ii) For $i \in \{1, 2, \ldots, m\}$, if $s(x_i) \in \mathsf{Const}(\mathcal{D})$, delete $(\forall x_i)$.

(lhs-iii) For $i \in \{1, 2, \ldots, n\}$, replace each A_i with the sole element of $\mathsf{Subst}(A_i, s)$.

Note in particular that universally quantified variables become existentially quantified. The existentially quantified versions represent a single but unspecified binding on those formerly universally quantified positions. Now, the *left-plus-right-hand-side information* of φ with respect to s, denoted $\mathsf{LRHSinfo}\langle\varphi, s\rangle$, is the sentence obtained from

$$(\forall x_1)(\forall x_2)\ldots(\forall x_m)(\exists y_1)(\exists y_2)\ldots(\exists y_r)(A_1 \wedge A_2 \wedge \ldots \wedge A_n \wedge B_1 \wedge B_2 \wedge \ldots \wedge B_s)$$

by following the steps (i)-(iii) above and, in addition to the following step.

(rhs) Replace each B_i with $\mathsf{Subst}(B_i, s)$.

For example, if $\varphi = (\forall x_1)(\forall x_2)(R(x_1, x_2) \Rightarrow (\exists y)(S(x_1, x_2, y)))$ and $s = \{\mathrm{a}_1/x_1, x/x_2\}$ then $\mathsf{LHSinfo}\langle\varphi, s\rangle = (\exists x)(R(\mathrm{a}_1, x))$ and $\mathsf{LRHSinfo}\langle\varphi, s\rangle = (\exists x)(\exists y)(R(\mathrm{a}_1, x) \wedge S(\mathrm{a}_1, x, y))$.

The following lemma states formally the intuition that if $\mathsf{LHSinfo}\langle\varphi, s\rangle$ is satisfied, and the rule φ holds, then $\mathsf{LRHSinfo}\langle\varphi, s\rangle$ holds as well. It is an immediate consequence of the above definitions.

3.17 Lemma *Let $\varphi \in \mathsf{TGHD}(\mathbf{D})$ and let $M \in \mathsf{DB}(\mathbf{D})$. Then $M \in \mathsf{AtMod}_{\mathfrak{I}}(\varphi)$ iff for every GHD-compatible substitution s on $\mathsf{Vars}(\varphi)$, if $\mathsf{LHSinfo}\langle\varphi, s\rangle \in \mathsf{Info}\langle M, \Upsilon_K\rangle$, then $\mathsf{LRHSinfo}\langle\varphi, s\rangle \in \mathsf{Info}\langle M, \Upsilon_K\rangle$ as well.* □

The case of EGHDs must be handled a bit differently, since the result of a deduction is not a new sentence in Υ_K but rather a restriction on existing sentences. Therefore, the crucial idea is to consider the effect of *reducing* a given sentence $\psi \in \Upsilon_K$ by an EGHD φ, with the latter possibly forcing certain terms of ψ to be equal.

3.18 Information associated with an EGHD Let φ be an EGHD of the form (GHD) of 3.15 above, and let $\psi \in \Upsilon_K$. Here ψ is a sentence to which the φ will be applied. If ψ can be unified with the left-hand side of φ, then the equality defined by the right-hand side of φ must be applied to φ.

For this to work, a suitable substitution must be applied, so s be any substitution on $\{x_1, x_2, \ldots, x_n\}$, the variables of φ. (In an EGHD, there are no existentially quantified variables of the form y_i in (GHD).) Note further that the GHD-compatibility property on a substitution does not apply here, since every EGHD is a universal sentence. However, the result of applying s to the left-hand side of φ must match ψ, so that the rule can fire. Formally, call s *φ-compatible* for ψ if $\mathsf{AtRep}(\mathsf{LHSinfo}\langle\varphi, s\rangle) \subseteq \mathsf{AtRep}(\psi)$. Thus, s is φ-compatible if, when applied to the left-hand side of φ, the conjuncts which are obtained (after removing any quantifiers) are a subset of those of ψ. Let $\mathsf{RHSinfo}\langle\varphi, s\rangle$ denote the equality atom obtained by applying the substitution s to B_1, the right-hand side of φ. (If φ is a mutual exclusion, then $\mathsf{RHSinfo}\langle\varphi, s\rangle = \bot$.) The *reduction of ψ by* $\mathsf{RHSinfo}\langle\varphi, s\rangle$, denoted $\mathsf{Reduction}\langle\psi, \mathsf{RHSinfo}\langle\varphi, s\rangle\rangle$ is defined by cases as follows.

(red-i) If $\mathsf{RHSinfo}\langle\varphi, s\rangle$ is of the form $(a_i = a_j)$ with $i \neq j$, or $\mathsf{RHSinfo}\langle\varphi, s\rangle = \bot$, then $\mathsf{Reduction}\langle\psi, \mathsf{RHSinfo}\langle\varphi, s\rangle\rangle = \bot$, the identically false assertion.

(red-ii) If $\mathsf{RHSinfo}\langle\varphi, s\rangle$ is of the form $(a_i = a_i)$ or $(x_i = x_i)$, then $\mathsf{Reduction}\langle\psi, \mathsf{RHSinfo}\langle\varphi, s\rangle\rangle = \psi$.

(red-iii) If $\mathsf{RHSinfo}\langle\varphi, s\rangle$ is of the form $(x_i = \mathrm{a}_j)$ or $(\mathrm{a}_j = x_i)$, then $\mathsf{Reduction}\langle\psi, \mathsf{RHSinfo}\langle\varphi, s\rangle\rangle$ is obtained by substituting a_j for x_i in ψ and removing the quantifier $(\exists x_i)$.

(red-iv) If $\mathsf{RHSinfo}\langle\varphi, s\rangle$ is of the form $(x_i = x_j)$ with $i \neq j$, then $\mathsf{Reduction}\langle\psi, \mathsf{RHSinfo}\langle\varphi, s\rangle\rangle$ is obtained by substituting x_i for x_j in ψ and removing the quantifier $(\exists x_j)$. (Note that the quantifier $(\exists x_i)$ must also be present in this case, and is not removed.)

For example, if $\varphi = (\forall x_1)(\forall x_2)(\forall x_3)((R(x_1, x_2) \wedge R(x_1, x_3)) \Rightarrow (x_2 = x_3))$ and $\psi = (\exists x_1)(\exists x_2)(R(\mathrm{a}_1, x_1) \wedge R(\mathrm{a}_1, x_2) \wedge S(x_1, x_2))$ then for $s = \{(\mathrm{a}_1/x_1, x_1/x_2, x_2/x_3)\}$, $\mathsf{LHSinfo}\langle\varphi, s\rangle = (\exists x_1)(\exists x_2)(R(\mathrm{a}_1, x_1) \wedge R(\mathrm{a}_1, x_2))$ and so is φ-compatible. $\mathsf{RHSinfo}\langle\varphi, s\rangle = (x_1 = x_2)$, and so $\mathsf{Reduction}\langle\psi, \mathsf{RHSinfo}\langle\varphi, s\rangle\rangle = (\exists x_1)(\exists x_2)(R(\mathrm{a}_1, x_1) \wedge R(\mathrm{a}_1, x_1) \wedge S(x_1, x_1))$. On the other hand, if $\psi = (\exists x_1)(R(x_1, \mathrm{a}_1) \wedge R(x_1, \mathrm{a}_2)$ then $s = \{x_1/x_1, \mathrm{a}_1/x_2, \mathrm{a}_2/x_3\}$ is φ-compatible but $\mathsf{RHSinfo}\langle\varphi, s\rangle = (a_1 = a_2)$. and so $\mathsf{Reduction}\langle\psi, \mathsf{RHSinfo}\langle\varphi, s\rangle\rangle = \bot$.

The following lemma is a routine consequence of the above.

3.19 Lemma *Let $\varphi \in \mathsf{EGHD}(\mathbf{D})$, let $M \in \mathsf{DB}(\mathbf{D})$, and let Ψ be a cover for $\mathsf{Info}\langle M, \Upsilon_K\rangle$ with respect to Υ_K. Then $M \in \mathsf{AtMod}_\mathbb{J}(\varphi)$ iff for every $\psi \in \Psi$ and every substitution s which is φ-compatible for ψ, $\mathsf{Reduction}\langle\psi, \mathsf{RHSinfo}\langle\varphi, s\rangle\rangle \in \mathsf{Info}\langle M, \Upsilon_K\rangle$ as well.* □

Finally, the main result, that schemata constrained by GHDs always admit canonical models conditionally, may be established.

3.20 Theorem — Conditional existence of canonical models *If $\mathsf{Constr}(\mathbf{D}) \subseteq \mathsf{GHD}(\mathbf{D})$, then for any finite $K \subseteq \mathsf{Const}(\mathcal{D})$, $\mathbf{D}$ admits canonical models conditionally with respect to Υ_K.*

PROOF: Let $\Phi \subseteq \mathsf{WFS}(\mathbf{D}, \exists\wedge+, K)$ have the property that it extends finitely to $\mathbf{D}$ for Υ_K. First, let $\varphi \in \mathsf{Constr}(\mathbf{D})$ be a TGHD, and let s be a GHD-compatible substitution for φ. It follows directly from 3.17 and the definition of XInfo (3.11(b)) that whenever $\mathsf{LHSinfo}\langle\varphi, s\rangle \in$

$\mathsf{XInfo}_{\mathbf{D}}\langle\Phi,\Upsilon_K\rangle$, then $\mathsf{LRHSinfo}\langle\varphi,s\rangle \in \mathsf{XInfo}_{\mathbf{D}}\langle\Phi,\Upsilon_K\rangle$ as well. A second application of 3.17 establishes that $M \in \mathsf{MinAtMod}_{\mathcal{J}}(\varphi)$ for every tuple-minimal Armstrong model M of $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi,\Upsilon_K\rangle$ with respect to Υ_K.

The proof for $\varphi \in \mathsf{Constr}(\mathbf{D})$ an EGHD is similar. Choose $\psi \in \Phi$ and let s be a φ-compatible substitution for ψ. In view of 3.19 and the definition of XInfo, it must be the case that $\mathsf{Reduction}\langle\psi,\mathsf{RHSinfo}\langle\varphi,s\rangle\rangle \in \mathsf{XInfo}_{\mathbf{D}}\langle\Phi,\Upsilon_K\rangle$ as well. Again, a second application of 3.19 establishes that $M \in \mathsf{MinAtMod}_{\mathcal{J}}(\varphi)$ for every tuple-minimal Armstrong model M of $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi,\Upsilon_K\rangle$ with respect to Υ_K. □

3.21 Weakly acyclic TGHDs To admit canonical models unconditionally, it is necessary to ensure that infinite increasing sequences in $\mathsf{XInfo}_{\mathbf{D}}\langle\Phi,\Upsilon_K\rangle$, as illustrated in 3.13, do not occur. Such infinite models are related to cycles in the tuple-generating dependencies. In [FKMP05, Def. 3.7], the notion of a *weakly acyclic set of TGDs* is developed, and it is shown [FKMP05, Thm. 3.9] that the chase procedure always terminates when the dependencies are limited to an acyclic set of of TGDs together with EGDs. The TGDs and EGDs of [FKMP05] differ only in relatively minor ways from the TGHDs and EGHDs of this paper; in particular, the result extends directly to TGDs. The details will not be worked out, but the following result is noted for completeness.

3.22 Corollary — (to 3.20) *If* $\mathsf{Constr}(\mathbf{D})$ *is finite and consists of a weakly acyclic set of TGHDs, together with any set of EGHDs, then then for any finite* $K \subseteq \mathsf{Const}(\mathcal{D})$, $\mathbf{D}$ *admits canonical models unconditionally with respect to* Υ_K.

PROOF: Combine the result of 3.20 with [FKMP05, Thm. 3.9]. □

4 Optimal Reflection of Insertions

The focus is now turned to the problem of characterizing optimal reflections of insertions into the view schema. Roughly, the idea is to reflect the information which must be added to the main schema and then construct an Armstrong model (with respect to Υ_K for a suitable K) of that information together with the current instance of the main schema. There are, of course, details to be developed and pitfalls to be avoided, all of which are discussed in this section. First of all, some basic definitions surrounding updates and information are developed.

4.1 Notational convention Throughout the rest of this paper, unless stated specifically to the contrary, take $\Gamma = (\mathbf{V},\gamma)$ to be a relational view of $\mathbf{D}$ of class $\exists\wedge+$.

4.2 Updates and reflections An *update* on $\mathbf{D}$ is a pair $(M_1,M_2) \in \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D})$. M_1 is the current instance, and M_2 the new instance. It is an *insertion* if $M_1 \subseteq M_2$, and a *deletion* if $M_2 \subseteq M_1$.

To describe the situation surrounding an update request on Γ, it is sufficient to specify the current instance M_1 of the main schema and the desired new instance N_2 of the view schema $\mathbf{V}$. The current instance of the view can be computed as $\gamma(M_1)$; it is only the new instance M_2 of the main schema (subject to $N_2 = \gamma(M_2)$) which must be obtained from an update strategy. Formally, an *update request* from Γ to $\mathbf{D}$ is a pair (M_1,N_2) in which $M_1 \in \mathsf{LDB}(\mathbf{D})$ (the old

instance of the main schema) and $N_2 \in \mathsf{LDB}(\mathbf{V})$ (the new instance of the view schema). If $\gamma(M_1) \subseteq N_2$, it is called an *insertion request*, and if $N_2 \subseteq \gamma(M_1)$, it is called a *deletion request*. Collectively, insertion requests and deletion requests are termed *unidirectional update requests*. A *realization* of (M_1, N_2) along Γ is an update (M_1, M_2) on $\mathbf{D}$ with the property that $\gamma(M_2) = N_2$. The update (M_1, M_2) is called a *reflection* (or *translation*) of the view update $(\gamma(M_1), N_2)$. The set of all realizations of (M_1, N_2) along Γ is denoted $\mathsf{UpdRealiz}\langle(M_1, N_2), \Gamma\rangle$. The subset of $\mathsf{UpdRealiz}\langle(M_1, N_2), \Gamma\rangle$ consisting of insertions (resp. deletions) is denoted $\mathsf{InsRealiz}\langle(M_1, N_2), \Gamma\rangle$ (resp. $\mathsf{DelRealiz}\langle(M_1, N_2), \Gamma\rangle$.

4.3 Update difference and optimal reflections The update difference of an update (M_1, M_2) on $\mathbf{D}$ with respect to a set $\Sigma \subseteq \mathsf{WFS}(\mathbf{D})$ is a measure of how much M_2 differs from M_1 in terms of satisfaction of the sentences of Σ. Formally, the *positive (Δ^+), negative (Δ^-), and total (Δ) update differences* of (M_1, M_2) with respect to Σ are defined as follows:

$$\begin{aligned}
\Delta^+\langle(M_1, M_2), \Sigma\rangle &= \mathsf{Info}\langle M_2, \Sigma\rangle \setminus \mathsf{Info}\langle M_1, \Sigma\rangle \\
\Delta^-\langle(M_1, M_2), \Sigma\rangle &= \mathsf{Info}\langle M_1, \Sigma\rangle \setminus \mathsf{Info}\langle M_2, \Sigma\rangle \\
\Delta\langle(M_1, M_2), \Sigma\rangle &= \Delta^+\langle(M_1, M_2), \Sigma\rangle \cup \Delta^-\langle(M_1, M_2), \Sigma\rangle
\end{aligned}$$

Note that, given $\varphi \in \Delta\langle(M_1, M_2), \Sigma\rangle$, it is always possible to determine whether $\varphi \in \Delta^+\langle(M_1, M_2), \Sigma\rangle$ or $\varphi \in \Delta^-\langle(M_1, M_2), \Sigma\rangle$ by checking whether or not $M_1 \in \mathsf{AtMod}_{\mathfrak{I}}(\varphi)$. Given an update request (M_1, N_2), the quality of a realization (M_1, M_2) is measured by its update difference. Formally, let $\Sigma \subseteq \mathsf{WFS}(\mathbf{D})$, let (M_1, N_2) be an update request from Γ to $\mathbf{D}$, let $T \subseteq \mathsf{UpdRealiz}\langle(M_1, N_2), \Gamma\rangle$, and let $(M_1, M_2) \in T$.

(a) (M_1, M_2) is *minimal* in T with respect to Σ if for any $(M_1, M_2') \in T$, if $\Delta\langle(M_1, M_2'), \Sigma\rangle \subseteq \Delta\langle(M_1, M_2), \Sigma\rangle$, then $\Delta\langle(M_1, M_2'), \Sigma\rangle = \Delta\langle(M_1, M_2), \Sigma\rangle$.

(b) (M_1, M_2) is *least* in T with respect to Σ if for all $(M_1, M_2') \in T$, $\Delta\langle(M_1, M_2), \Sigma\rangle \subseteq \Delta\langle(M_1, M_2'), \Sigma\rangle$.

4.4 Update classifiers An *update classifier* for $\mathbf{D}$ is simply a set Σ of information-monotone sentences. In this work, the set Σ will always be taken to be either $\mathsf{GrAtoms}(\mathbf{D})$ or else $\mathsf{WFS}(\mathbf{D}, \exists\wedge+, K) = \Upsilon_K$ for an appropriate set K of constants.

Let (M_1, N_2) be an update request from Γ to $\mathbf{D}$, let $T \subseteq \mathsf{UpdRealiz}\langle(M_1, N_2), \Gamma\rangle$, and let $(M_1, M_2) \in T$.

(a) (M_1, M_2) is $\langle\Upsilon_K, T\rangle$-*admissible* if it is minimal in T with respect to both Υ_K and $\mathsf{GrAtoms}(\mathbf{D})$.

(b) (M_1, M_2) is $\langle\Upsilon_K, T\rangle$-*optimal* if it is $\langle\Upsilon_K, T\rangle$-admissible and least in T with respect to Σ.

Roughly, (M_1, M_2) is admissible if no other realization is better, and it is optimal if it is better than all others, up to the equivalence defined by Σ. Observe that if some update request is $\langle\Upsilon_K, T\rangle$-optimal, then all $\langle\Upsilon_K, T\rangle$-admissible update requests are $\langle\Upsilon_K, T\rangle$-optimal.

As a notational shorthand, if $T = \mathsf{InsRealiz}\langle(M_1, N_2), \Gamma\rangle$ (resp. $T = \mathsf{DelRealiz}\langle(M_1, N_2), \Gamma\rangle$), that is, if T is the set of all possible insertions (resp. deletions) which realize (M_1, N_2), then $\langle\Upsilon_K, T\rangle$-admissible and $\langle\Upsilon_K, T\rangle$-optimal will be abbreviated to $\langle\Upsilon_K, \uparrow\rangle$-admissible and $\langle\Upsilon_K, \uparrow\rangle$-optimal (resp. $\langle\Upsilon_K, \downarrow\rangle$-admissible and $\langle\Upsilon_K, \downarrow\rangle$-optimal).

For $\Sigma = \mathsf{GrAtoms}(\mathbf{D})$, admissibility reduces to minimality in the sense of symmetric difference of sets as sketched in 1.2. More concretely, given an update request (M_1, N_2), a realization (M_1, M_2) is $\langle \mathsf{GrAtoms}(\mathbf{D}), T\rangle$-admissible if for no other realization $(M_1, M_2') \in T$ is it the case that $\mathsf{SymDiff}\langle M_1, M_2'\rangle \subseteq \mathsf{SymDiff}\langle M_1, M_2\rangle$. Similarly, (M_1, M_2) is $\langle \mathsf{GrAtoms}(\mathbf{D}), T\rangle$-optimal if for every other realization (M_1, M_2'), $\mathsf{SymDiff}\langle M_1, M_2\rangle \subseteq \mathsf{SymDiff}\langle M_1, M_2'\rangle$. Minimality with respect to $\mathsf{GrAtoms}(\mathbf{D})$ is referred to as *tuple minimality*, in harmony with the terminology already introduced for Armstrong models in Section 3.

The major theme of this paper is that tuple minimality, by itself, is not sufficient to characterize optimal updates. Rather, optimality with respect to Υ_K for a suitably chosen set K of constants is also essential. This issue is next addressed in detail.

4.5 The constants associated with an update request In reflecting an update from a view to the main schema, the use of new constants in generic models is crucial. For a constant to be "new", it is not sufficient that it merely not appear in the current or proposed view instance, or the current instance of the main schema. Rather, it must not appear in any constraint or defining formula for the main schema or view. For example, referring back to the example of the null-value constraint $(\forall x_1)(\forall x_2)(R(x_1, x_2) \Rightarrow S(x_1, x_2, \nu)$ of 3.15, it would be inappropriate to use ν as a generic constant, since it already has another meaning within the global context of all databases. Thus, the pool of generic constants must also exclude any which occur in constraints or in the defining formulas for views. Specifically, the following sequence of definitions leads to the acceptable pool of generic constants.

First of all, define the constant symbols of the schema $\mathbf{D}$ to be those of its constraints.

(a) $\mathsf{ConstSym}(\mathbf{D}) = \mathsf{ConstSym}(\mathsf{Constr}(\mathbf{D}))$

Next, define the constant symbols of the view $\Gamma = (\mathbf{V}, \gamma)$ to be those of the main schema $\mathbf{D}$, together with those of both the view schema $\mathbf{V}$ and the view mapping γ.

(b) $\mathsf{ConstSym}(\Gamma) = \mathsf{ConstSym}(\mathsf{Constr}(\mathbf{D})) \cup \mathsf{ConstSym}(\mathsf{Constr}(\mathbf{V})) \cup \mathsf{ConstSym}(\gamma)$

The constant symbols of an update request u are those of its instances.

(c) For $u = (M_1, N_2)$ an update request from Γ to $\mathbf{D}$, $\mathsf{ConstSym}(u) = \mathsf{ConstSym}(M_1) \cup \mathsf{ConstSym}(N_2)$.

Finally, define the *total constant set of* u, denoted $\mathfrak{C}_u$, to be the constant symbols of u together with those of Γ.

(d) $\mathfrak{C}_u = \mathsf{ConstSym}(\Gamma) \cup \mathsf{ConstSym}(u)$.

Note that $\mathsf{ConstSym}(N_2) \subseteq \mathsf{ConstSym}(M_1) \cup \mathsf{ConstSym}(\gamma) \cup \mathsf{ConstSym}(\mathbf{V})$, so the following more compact representation is also valid.

(d′) $\mathfrak{C}_u = \mathsf{ConstSym}(\Gamma) \cup \mathsf{ConstSym}(M_1)$.

Since database schemata, even those of views, are always taken to be constant finite (see 2.5), it follows that $\mathfrak{C}_u$ is always a finite set. Since $\mathsf{Const}_{\mathcal{D}}(A)$ is infinite for every $A \in \mathcal{A}_{\mathcal{D}}$, it follows that there are always infinitely many "available" constant symbols for each attribute A which do not lie in $\mathfrak{C}_u$. The set $\mathfrak{C}_u$ will thus be taken to be the set of constant symbols which may not be used as generic values in constructing Armstrong models.

4.6 Information lifting Let $N \in \mathsf{DB}(\mathbf{V})$. The *information lifting* of N along Γ, denoted $\mathsf{InfoLift}\langle N, \Gamma\rangle$, is the minimum information in $\mathsf{WFS}(\mathbf{D}, \exists\wedge+)$ which any $M \in \mathsf{DB}(\mathbf{D})$ with $\gamma(M) = N$ must have. Formally, this is recaptured as follows.

$$\mathsf{InfoLift}\langle N, \Gamma\rangle = \{\mathsf{Substf}\langle \gamma, t\rangle \mid t \in N\}$$

Note that $\mathsf{InfoLift}\langle N, \Gamma\rangle \subseteq \Upsilon_K$ with $K = \mathsf{ConstSym}(N_2) \cup \mathsf{ConstSym}(\gamma)$.

Given an insertion request $u = (M_1, N_2)$ from Γ to $\mathbf{D}$, and let $M_2 \in \mathsf{InsRealiz}\langle (M_1, N_2), \Gamma\rangle$. The least information which M_2 must have is $\mathsf{InfoLift}\langle N_2, \Gamma\rangle \cup M_1$, closed up under the constraints in $\mathsf{Constr}(\mathbf{D})$. This is recaptured formally as

$$\mathsf{LeastRefl}\langle u, \Gamma\rangle = \mathsf{XInfo}_{\mathbf{D}}\langle M_1 \cup \mathsf{InfoLift}\langle N_2, \Gamma\rangle, \Upsilon_{\mathfrak{C}_u}\rangle$$

and is called the *least reflection of u along Γ*.

In the context of the example of 1.3, with $u = (M_{00}, N_{01})$, the information lifting $\mathsf{InfoLift}\langle N_{01}, \Pi^{\mathbf{E}_1}_{R'[AB]}\rangle = \{(\exists x_1)(\exists x_2)(\exists x_3)(R(\mathrm{a}_0, \mathrm{b}_0, x_1) \wedge R(\mathrm{a}_1, \mathrm{b}_1, x_2) \wedge R(\mathrm{a}_2, \mathrm{b}_2, x_3))\}$, while the least reflection $\mathsf{LeastRefl}\langle u, \Pi^{\mathbf{E}_1}_{R'[AB]}\rangle = M_{00} \cup \{(\exists x_1)(\exists x_2)(R(\mathrm{a}_2, \mathrm{b}_2, x_1) \wedge (S(x_1, x_2)))\}$.

Using the construction of 3.9, a tuple-minimal Armstrong model of $\mathsf{LeastRefl}\langle u, \Pi^{\mathbf{E}_1}_{R'[AB]}\rangle$ is obtained by replacing the variables by distinct and new constants; for example $M_{01} = M_{00} \cup \{R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2), S(\mathrm{c}_2, \mathrm{d}_2)\}$.

4.7 Proposition — Characterization of optimal insertions *Assume that* $\mathbf{D}$ *admits canonical models conditionally, and let* $u = (M_1, N_2)$ *be an insertion request from* Γ *to* $\mathbf{D}$. *Then* (M_1, M_2) *is a* $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$*-optimal realization of* u *iff the following two conditions hold.*

(i) M_2 *is a tuple-minimal Armstrong model of* $\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle$ *with respect to* $\Upsilon_{\mathfrak{C}_u}$.

(ii) $\gamma(M_2) = N_2$.

PROOF: Certainly, if (i) and (ii) hold, then M_2 is a $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$-optimal realization of u. On the other hand, if M_2 is a $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$-optimal realization of u, then (ii) holds trivially. Since $\mathbf{D}$ admits canonical models conditionally, if $\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle$ admits a finite cover, then a tuple-minimal Armstrong model of that set is a a $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$-optimal realization of u, just by construction. If $\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle$ does not admit a finite cover, then, as illustrated in 3.13, it must contain an infinite increasing sequence $\langle \varphi_0, \varphi_1, \varphi_2, \ldots\rangle$ of sentences with $\mathsf{AtMod}_{\mathfrak{I}}(\bigcup\{\varphi_i \mid 0 \leq i \leq k\}) \not\subseteq \mathsf{AtMod}_{\mathfrak{I}}(\varphi_k)$ for any $k > 0$, and so it cannot have a finite Armstrong model. Thus, any $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$-admissible realization must satisfy some sentence not in $\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle$, and so no $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$-optimal realization can exist. □

4.8 Example — Orphan tuples Even in the case that $\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle$ satisfies condition (i) of 4.7, it may not satisfy condition (ii); that is, it may not be the case that $\gamma(M_2) = N_2$. The problem lies with so-called *orphan tuples*, which are illustrated via the following example.

Let $\mathbf{E}_4$ be the schema having the single relation symbol $R[AB]$, constrained by the dependency $(\exists x_1)(\exists x_2)(R(x_1, x_2))$ which simply asserts that the instance of R is nonempty. It is immediate that this dependency is a TGHD (with $m = n = 0$ in the pattern (GHD) of 3.15). The view $\Pi^{\mathbf{E}_4}_{A+B} = (\mathbf{W}_4, \pi^{\mathbf{E}_4}_{A+B})$ has two relation symbols $R_A[A]$ and $R_B[B]$, defined by the obvious projections $\pi^{\mathbf{E}_4}_A$ and $\pi^{\mathbf{E}_4}_B$. The only constraints on the view schema are

$(\exists x_1)(R_A(x_1))$ and $(\exists x_1)(R_B(x_1))$, so $\mathsf{Constr}(\mathbf{W}_4)$ consists of GHDs as well. Now with $M_1 = \{R(\mathrm{a}_0, \mathrm{b}_0), R(\mathrm{a}_1, \mathrm{b}_1)\}$ the instance of $\mathbf{E}_4$, $\pi^{\mathbf{E}_4}_{A+B}(M_1) = N_1 = \{R_A(\mathrm{a}_0), R_A(\mathrm{a}_1), R_B(\mathrm{b}_0), R_b(\mathrm{b}_1)\}$. Let N_2 be the view instance obtained by inserting $R_A(\mathrm{a}_2)$ into N_1, and define $u = (M_1, N_2)$. Then $\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle = M_1 \cup (\exists x_1)(R(\mathrm{a}_2, x_1))$. A canonical model M_2 of this least reflection is of the form $M_1 \cup \{R(\mathrm{a}_2, \mathrm{b}_2)\}$, with $\mathrm{b}_2 \notin \mathfrak{C}_u$, so $\pi^{\mathbf{E}_4}_{A+B}(M_2) = N_2 \cup \{R_B(\mathrm{b}_2)\}$. Here $R_B(\mathrm{b}_2)$ is termed an *orphan tuple*; it represents newly inserted information which has made its way back to the view. The tuple can be made to "disappear" by replacing b_2 with an existing value for attribute B, say b_1. However, in that case, while the resulting instance $M_2' = M_1 \cup \{R(\mathrm{a}_2, \mathrm{b}_1)\}$ does map to N_2 under $\pi^{\mathbf{E}_4}_{A+B}$, $\mathsf{Info}\langle M_2, \Upsilon_{\mathfrak{C}_u}\rangle$ is a proper subset of $\mathsf{Info}\langle M_2', \Upsilon_{\mathfrak{C}_u}\rangle$, and so M_2' is not $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$-optimal. Thus, (M_1, M_2') is a $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$-admissible solution to the update request (M_1, N_2), but it is not optimal. This problem cannot be made to disappear via clever formulation; in this example, there are no optimal solutions. Fortunately, orphan tuples can be ruled out by requiring that Γ reflect deletions, as described below.

4.9 Reflection of deletions The view $\Gamma = (\mathbf{V}, \gamma)$ *reflects deletions* if every deletion request (M_1, N_2) from Γ to $\mathbf{D}$ admits a realization which is itself a deletion.

In the example of 4.8 above, $\Pi^{\mathbf{E}_4}_{A+B}$ does not reflect deletions. There is no realization of the update request (M_1', N_2) which is a deletion, since with $M_1 = \{R(\mathrm{a}_0, \mathrm{b}_0), R(\mathrm{a}_1, \mathrm{b}_1)\}$ the instance of the main schema $\mathbf{E}_4$, there is no way to realize the deletion of $R_2(\mathrm{b}_1)$ from the view instance N_1 as a deletion from M_1; $R(\mathrm{a}_1, \mathrm{b}_1)$ must be deleted, which would also remove $R_1(\mathrm{a}_1)$.

4.10 Lemma — Reflection of deletions implies no orphan tuples *Assume that* $\mathbf{D}$ *admits canonical models conditionally, and let* $u = (M_1, N_2)$ *be an insertion request from* Γ *to* $\mathbf{D}$ *for which* $\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle$ *admits a tuple-minimal Armstrong model* M_2 *with respect to* $\Upsilon_{\mathfrak{C}_u}$. *Then, if* Γ *reflects deletions,* $\gamma(M_2) = N_2$.

PROOF: It is clear that $N_2 \subseteq \gamma(M_2)$. By the definition of reflection of deletions, there is an $M_2' \in \mathsf{LDB}(\mathbf{D})$ with $M_2' \subseteq M_2$ and $\gamma(M_2') = N_2$. However, since M_2 is already tuple minimal, it follows that $M_2' = M_2$. Thus, $\gamma(M_2) \setminus N_2 = \emptyset$, and so $\gamma(M_2) = N_2$, as required. □

Finally, it can be established that for views with reflect deletions, condition (ii) of 4.7 is superfluous.

4.11 Proposition *Assume that* Γ *reflects deletions, that* $\mathbf{D}$ *admits canonical models conditionally, and let* $u = (M_1, N_2)$ *be an insertion request from* Γ *to* $\mathbf{D}$. *Then* (M_1, M_2) *is a* $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$*-optimal realization of* u *iff* M_2 *is a tuple-minimal Armstrong model of* $\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle$ *with respect to* $\Upsilon_{\mathfrak{C}_u}$.

PROOF: The proof follows immediately from 4.7 and 4.10. □

4.12 Example — Dependence upon the instance of the main schema There is a another issue which arises in applying 4.7 and 4.11; namely, that whether or not an optimal insertion exists may depend upon M_1 and not simply the view instance $\gamma(M_1)$. To illustrate this phenomenon, consider the schema $\mathbf{E}_5$ containing the single relation symbol $R[ABCDE]$ governed by the FDs in $\mathcal{F} = \{A \to D, B \to E, DE \to C\}$. The view to be updated is the projection onto ABC; it contains the single relation symbol $R_{ABC}[ABC]$, and

is represented more formally as $\Pi^{\mathbf{E}_5}_{ABC} = (\mathbf{W}_5, \pi^{\mathbf{E}_5}_{ABC})$. It is easy to see that $\mathsf{Constr}(\mathbf{W}_5) = \{AB \to C\}$, so that the view is constrained by FDs alone. Each of the two instances $M_{20} = \{R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0, \mathrm{d}_0, \mathrm{e}_0), R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_1, \mathrm{e}_1)\}$, and $M_{21} = \{R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0, \mathrm{d}_0, \mathrm{e}_0), R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_1, \mathrm{e}_0)\}$ of $\mathbf{E}_5$ maps to the view instance $N_1 = \{R_{ABC}(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0), R_{ABC}(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1)\}$ under $\pi^{\mathbf{E}_5}_{ABC}$. Consider the view update which inserts the tuple $R_{ABC}(\mathrm{a}_0, \mathrm{b}_1, \mathrm{c}_2)$, so that the desired new view instance is $N_2 = N_1 \cup \{R_{ABC}(\mathrm{a}_0, \mathrm{b}_1, \mathrm{c}_2)\}$. For convenience, write $u_{20} = (M_{20}, N_2)$ and $u_{21} = (M_{21}, N_2)$. Then $\mathsf{LeastRefl}\langle u_{20}, \Upsilon_{\mathfrak{C}_u}\rangle$ is given by $M'_{20} = M_{20} \cup R(\mathrm{a}_0, \mathrm{b}_1, \mathrm{c}_2, \mathrm{d}_0, \mathrm{e}_1)$, and it is easy to see that $\gamma(M'_{20}) = N_2$, as desired. On the other hand, $\mathsf{LeastRefl}\langle u_{21}, \Upsilon_{\mathfrak{C}_{u'}}\rangle$ does not exist. Indeed, the FDs stipulate that it would need to be $M'_{21} = M_{21} \cup R(\mathrm{a}_0, \mathrm{b}_1, \mathrm{c}_2, \mathrm{d}_0, \mathrm{e}_0)$, but the presence of both $R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{c}_0, \mathrm{d}_0, \mathrm{e}_0)$ and $R(\mathrm{a}_0, \mathrm{b}_1, \mathrm{c}_2, \mathrm{d}_0, \mathrm{e}_0)$ in the same instance violates the FD $DE \to C$. In the case that the instance of $\mathbf{E}_5$ is M'_1, there is no insertion which will realize the insertion of $R_{ABC}(\mathrm{a}_0, \mathrm{b}_1, \mathrm{c}_2)$ into $\gamma(M'_1) = N_1$. To realize this update, tuples of M'_1 must either be deleted or else altered.

This example thus shows that the principles of *reflection of monotonicity* and *invariance of admissibility*, as defined in Sec. 1, cannot always be realized simultaneously, even when the reflections for certain databases of the main schema are very well behaved. Fortunately, this phenomenon can be ruled out by requiring that Γ reflect insertions, as described below.

4.13 Reflection of insertions The view $\Gamma = (\mathbf{V}, \gamma)$ *reflects insertions* if every insertion request (M_1, N_2) from Γ to $\mathbf{D}$ which admits a realization admits one which is itself a insertion.

In the example of 4.12 above, $\Pi^{\mathbf{E}_5}_{ABC}$ does not reflect insertions, since there is no reflection of the insertion request (M'_1, N_2) which is itself an insertion.

4.14 Observation *If Γ reflects insertions, then for every insertion request $u = (M_1, N_2)$ from Γ to $\mathbf{D}$, $\mathsf{LDB}(\mathbf{D}) \cap \mathsf{AtMod}_{\mathfrak{J}}(\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle)$ is nonempty.*

PROOF: Since $\mathsf{LeastRefl}\langle u, \Upsilon_{\mathfrak{C}_u}\rangle$ is the least information which any realization of u which is an insertion must contain, it must be consistent if any insertion has that property. □

4.15 Strong monotonicity of views In view of 4.11 and 4.14, it is clear that for the reflection of updates to be well behaved, a view should reflect both deletions and insertions. Because of the importance of this property, it is given a special name. Call Γ *strongly monotonic* if it reflects both insertions and deletions.

Finally, conditions which guarantee the existence of optimal insertions may be established.

4.16 Theorem *If $\mathbf{D}$ admits canonical models unconditionally with respect to Υ_K and Γ is strongly monotonic, then every insertion request u from Γ to $\mathbf{D}$ admits a $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$-optimal realization.*

PROOF: The proof follows from 4.11 and 4.14. □

4.17 Corollary *If $\mathsf{Constr}(\mathbf{D})$ is finite and consists of a weakly acyclic set of TGHDs, together with any set of EGHDs, and Γ is strongly monotonic, then every insertion request u from Γ to $\mathbf{D}$ has a $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow\rangle$-optimal realization.*

PROOF: The proof follows from 4.16 and 3.22. □

5 Characterization of Strongly Monotonic Views

The characterizations 4.16 and 4.17 lead to the further problem of identifying conditions under which a view is strongly monotonic. In general, this does not appear to be an easy question to answer. However, for schemata constrained by FDs and unary inclusion dependencies (UINDs), and for views defined by projections, it is possible to identify some sufficient conditions which are easily verified in practice.

5.1 Partial dependence and complete sets In the example of 4.12, the problem is that there is a sort of weak dependence of A upon C via $A \to D; D \subseteq DE; DE \to C$, while the Fd $A \to C$ itself does not hold. To obtain strong monotonicity in the context of projections of views constrained by FDs, it is precisely this sort of weak dependence which must not be present without the associated FD also holding. To formalize this idea for a general schema $\mathbf{D}$ which is constrained by FDs, let $R \in \mathsf{Rels}(\mathbf{D})$, let $\mathcal{F}_R$ be a set of FDs on R, and let $A, B \in \mathsf{Ar}_{\mathbf{D}}(R)$.

(a) *A functionally influences B*, denoted $A \dashrightarrow B$, if there is a sequence $\langle A_0, A_1, A_2, \ldots, A_n\rangle$ of elements of $\mathsf{Ar}_{\mathbf{D}}(R)$ with $A = A_0$ and $B = A_k$, and a sequence $X_1 \to A_1, X_2 \to A_2, \ldots, X_k \to A_k$ of FDs in the closure of $\mathcal{F}_R$ with the property that $A_i \in X_{i+1}$ for $i \in \{0, \ldots, k\}$. This may be visualized as follows.

$$A = A_0 \in X_1; X_1 \to A_1; A_1 \in X_2; X_2 \to A_2; \ldots A_{k-1} \in X_k; X_k \to A_k = B$$

Functional influence is weaker than functional dependence. $A \dashrightarrow B$ simply means that the value of A could influence the value of B, subject to information about the values of other attributes. Put another way, if $A \dashrightarrow B$ does not hold, then the value of A cannot influence the value of B via the FDs which hold on the schema.

(b) Call a subset $Y \subseteq \mathsf{Ar}_{\mathbf{D}}(R)$ *complete for* $\mathcal{F}_R$ if if whenever $A, B \in Y$ with $A \dashrightarrow B$, then there is a $Z \subseteq Y$ with $A \in Z$ and $Z \to B \in \mathsf{Closure}\langle \mathcal{F}_R, \mathsf{WFS}(\mathbf{D}, \exists\wedge+)\rangle$.

In other words, completeness states that if $A \dashrightarrow B$ holds in Y, then an FD whose left hand side contains A and whose right-hand side is B also embeds into Y.

5.2 Simple projective views and complete views Informally, $\Gamma = (\mathbf{V}, \gamma)$ is a *simple projective view* of $\mathbf{D}$ if the schema $\mathbf{V}$ consists of at most one projection of each relation symbol of $\mathbf{D}$. Formally, a simple projective view $\Gamma = (\mathbf{V}, \gamma)$ of $\mathbf{D}$ is defined by an injective function $\mathsf{SP}_\Gamma : \mathsf{Rels}(\mathbf{V}) \to \mathsf{Rels}(\mathbf{D})$ with $\mathsf{Ar}_{\mathbf{V}}(R) \subseteq \mathsf{Ar}_{\mathbf{D}}\mathsf{SP}_\Gamma(R)$ for each $R \in \mathsf{Rels}(\mathbf{V})$. $\mathsf{SP}_\Gamma(R)$ is the relation of which R is a projection. The property that SP_Γ be injective; that is, that each relation of $\mathbf{V}$ be the projection of a distinct relation in $\mathbf{D}$, is critical.

Now assume that Γ is a simple projective view and that each $S \in \mathsf{Rels}(\mathbf{D})$ is constrained by a set $\mathcal{F}_S$ of FDs. Call Γ *FD-complete* if for each $R \in \mathsf{Rels}(\mathbf{V})$, the set $\mathsf{Ar}_{\mathbf{V}}(R)$ is complete for $\mathcal{F}_{\mathsf{SP}_\Gamma(R)}$. In other words, for each relation symbol of $\mathbf{V}$, the projected attributes must be complete in the relation of $\mathbf{D}$ from which they originate.

5.3 Proposition *Suppose that $\mathbf{D}$ is constrained solely by FDs, and that Γ is a simple projective view which is FD-complete. Then Γ is strongly monotonic.*

PROOF: Since the relation symbols of $\mathbf{D}$ are independent of one another, it suffices to consider the situation in which $\mathsf{Rels}(\mathbf{D})$ consists of single relation symbol $R[X]$, constrained by FDs $\mathcal{F}_R$, with $\mathsf{Rels}(\mathbf{V})$ consisting of a single relation symbol $R'[Y]$ with $Y \subseteq X$, and $\gamma = \pi_Y^{\mathbf{D}}$ defining the projection of $R[X]$ onto $R'[Y]$.

First of all, since FDs are EGDS, deletions are reflected trivially. Now let $u = (M_1, N_2)$ be an insertion request, and let $P = N_2 \setminus \gamma(M_1)$. Each tuple $t' = R'(\mathrm{a}_1, \mathrm{a}_2, \ldots, \mathrm{a}_m) \in P$ must be lifted to a tuple $t = R(\mathrm{a}_1, \mathrm{a}_2, \ldots, \mathrm{a}_m, \mathrm{b}_1, \mathrm{b}_2, \ldots, \mathrm{b}_k)$ in $\mathbf{D}$, with the b_i's values for the attributes in $X \setminus Y$. In view of the FD-completeness property of Γ, the values of the b_i's cannot be influenced by the values of the a_i's. (Put another way, the classical chase procedure [BV84] will not force the values of the b_i's to match those of any existing tuples.) Thus the insertion will not have the problems which are illustrated in 4.12, and so the update may be realized as an insertion. □

5.4 Examples — FDs and strong monotonicity As noted above, the example of 4.12 is not strongly monotonic. However, if the set $\mathcal{F} = \{A \to D, B \to E, DE \to C\}$ of FDs is replaced by $\mathcal{F}' = \{A \to D, B \to E, D \to C\}$, then $A \to C$ is in the closure of $\mathcal{F}'$, ABC is FD-complete, and the associated projection is strongly monotonic.

5.5 Examples — UINDs and strong monotonicity It is possible to obtain conditions under which simple projective views governed by UINDs are strongly monotonic, although some care is necessary. A few examples will illustrate the central issues.

First of all, let $\mathbf{E}_6$ denote the schema with two binary relation symbols $R_1[AB]$ and $R_2[AB]$, governed by the IND $R_1[AB] \sqsubseteq R_2[AB]$. $R_2[AB]$ is also governed by the FD $A \to B$, and this FD is inherited by $R_1[AB]$ via the IND. The view $\Pi_{R_1}^{\mathbf{E}_6}$ preserves $R_1[AB]$ but discards $R_2[AB]$. Let $M_1 = \{R_2(\mathrm{a}_0, \mathrm{b}_0)\}$ be the current instance of $\mathbf{E}_6$, so that the instance of the view is $\emptyset$. Consider the update request (M_1, N_2) with $N_2 = \{R_1(\mathrm{a}_0, \mathrm{b}_0)\}$. This update can be realized as the insertion of $\{R_1(\mathrm{a}_0, \mathrm{b}_0)\}$ to R in M_1. However, if the instance of $\mathbf{E}_6$ were $M_1' = \{R_2(\mathrm{a}_0, \mathrm{b}_1)\}$ instead, this view insertion would not be realizable as an insertion to the main schema, since the insertion of $R_1(\mathrm{a}_0, \mathrm{b}_0)$ requires the insertion of $R_2(\mathrm{a}_0, \mathrm{b}_0)$, and $\{R_2(\mathrm{a}_0, \mathrm{b}_0), R_2(\mathrm{a}_0, \mathrm{b}_1)\}$ together violate the FD $A \to B$ on R_2. Thus, any reflection of (M_1', N_2) must delete $R_2(\mathrm{a}_0, \mathrm{b}_1)$. Hence $\Pi_{R_1}^{\mathbf{E}_6}$ does not reflect insertions. In general, non-unary INDs are very problematic with respect to strong montonicity, and so in this paper attention will be restricted to UINDs. Note that if the IND above is changed to $R_1[A] \subseteq R_2[A]$, then the problem disappears.

Next, let $\mathbf{E}_7$ denote the schema with the single relation symbol $R[ABC]$, constrained by $R[B] \sqsubseteq R[C]$, and let $\Pi_A^{\mathbf{E}_7} = (\mathbf{W}_7, \pi_A^{\mathbf{E}_7})$ be the view which projects $R[ABC]$ onto $R'[A]$. Let $M_1 = \{R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{b}_0)R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{b}_1)\}$, so that $\pi_A^{\mathbf{E}_7}(M_1) = N_1 = \{R'(\mathrm{a}_0), R'(\mathrm{a}_1)\}$. The view instance $N_2 = N_1 \setminus R'(\mathrm{a}_1) = \{R'(\mathrm{a}_0)\}$ may be realized via the deletion of $R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{b}_1)$ from M_1. On the other hand, for $M_1' = \{R(\mathrm{a}_0, \mathrm{b}_0, \mathrm{b}_1)R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{b}_0)\}$, $\pi_A^{\mathbf{E}_7}(M_1') = N_1$ as well, yet there is no subset of M_1' which maps to N_2 under $\pi_A^{\mathbf{E}_7}$. Hence $\Pi_A^{\mathbf{E}_7}$ does not reflect deletions. A similar problem occurs if the view is taken to be the projection $\Pi_{AB}^{\mathbf{E}_7}$ onto AB or the projection $\Pi_{AC}^{\mathbf{E}_7}$ onto AC. Thus, the second principle to enforce is that for every intrarelational UIND $R[A_1] \subseteq R[A_2]$, if R is projected at all to the view, then the projection must contain $R[A_1A_2]$. Less formally, all intrarelational UINDs must be completely visible in the projection.

Finally, let $\mathbf{E}_8$ be the schema with three relation symbols $R_1[AB]$, $R_2[AB]$, and $R_3[AB]$, constrained by $R_1[A] \sqsubseteq R_2[A]$ and $R_2[B] \sqsubseteq R_3[B]$. The simple projective view Ω_7 contains the pro-

jections $R'_1[A]$ and $R'_3[A]$ of R_1 and R_3, respectively. Let $M_1 = \{R_1(\mathrm{a}_0, \mathrm{b}_0), R_2(\mathrm{a}_0, \mathrm{b}_1), R_3(\mathrm{a}_0, \mathrm{b}_1), R_3(\mathrm{a}_1, \mathrm{b}_0)\}$, so that the view instance is $N_1 = \{R'_1(\mathrm{a}_0), R'_3(\mathrm{a}_0), R'_3(\mathrm{a}_1)\}$. Let $N_2 = \{R'_1(\mathrm{a}_0), R'_3(\mathrm{a}_0)\}$. It is easy to see that the deletion request (M_1, N_2) is realizable by deleting $R_3(\mathrm{a}_1, \mathrm{b}_0)$ from M_1. However, for $M'_1 = \{R_1(\mathrm{a}_0, \mathrm{b}_0), R_2(\mathrm{a}_0, \mathrm{b}_1), R_3(\mathrm{a}_1, \mathrm{b}_1), R_3(\mathrm{a}_0, \mathrm{b}_0)\}$, the deletion request (M'_1, N_2) is not realizable as a deletion, since $R_3(\mathrm{a}_1, \mathrm{b}_1)$ must be deleted, which would violate $R_2[B] \sqsubseteq R_3[B]$. Note that parts of the "chain" $R_1[A] \sqsubseteq R_2[A] \leftrightarrow R_2[B] \sqsubseteq R_3[B] \leftrightarrow R_3[A]$ are not visible in the view, with $\leftrightarrow$ meaning "occurs in the same relation. The final condition to be enforced is that all intermediate entries in such a chain must appear in the view whenever the end points do.

The next task is to formalize all of this.

5.6 UINDs and projective views The set of *unary projections* of $\mathbf{D}$ consists of all expressions of the form $R[A]$ with $R \in \mathsf{Rels}(\mathbf{D})$ and $A \in \mathsf{Ar}_{\mathbf{D}}(R)$. Thus, the unary projections are precisely those which can occur as the left-hand or right-hand side of a UIND. Formally, $\mathsf{UProj}(\mathbf{D}) = \{R[A] \mid R \in \mathsf{Rels}(\mathbf{D}) \text{ and } A \in \mathsf{Ar}_{\mathbf{D}}(R)\}$.

Define $\mathsf{UIND}(\mathbf{D})$ to be the set of all UINDs which are implied by $\mathsf{Constr}(\mathbf{D})$. Say that $R[A]$ *participates* in $\mathsf{UIND}(\mathbf{D})$ if if it appears as either the left-hand side or else the right-hand side of some nontrivial $\varphi \in \mathsf{UIND}(\mathbf{D})$. Here a *nontrivial* UIND is one which is not true in $M \in \mathsf{DB}(\mathbf{D})$; *i.e.*, one which is not of the form $R[A] \sqsubseteq R[A]$. For $R[A_1], R[A_2] \in \mathsf{UProj}(\mathbf{D})$ both participants in $\mathsf{UIND}(\mathbf{D})$ and over the same relation R, write $R[A_1] \leftrightarrow R[A_2]$. Note that $R[A_1] \leftrightarrow R[A_2]$ does not necessarily imply that one of $R[A_1] \sqsubseteq R[A_2]$ or $R[A_2] \sqsubseteq R[A_1]$ holds; $R[A_1]$ and $R[A_2]$ may well participate in distinct UINDs of $\mathbf{D}$.

Define the *UIND-graph* of $\mathbf{D}$, denoted $\mathsf{UGraph}(\mathbf{D})$, to be the directed graph whose vertices are the members of $\mathsf{UProj}(\mathbf{D})$, with an edge from $R_1[A_1]$ to $R_2[A_2]$ iff $R_1[A_1] \neq R_2[A_2]$ and either $R_1[A_1] \sqsubseteq R_2[A_2] \in \mathsf{UIND}(\mathbf{D})$ or else $R_1 = R_2$ and $R_1[A_1] \leftrightarrow R_2[A_2]$. Thus, the UIND-graph recaptures "chains of influence for UINDs, much as functional influence does for FDs.

Finally, assume that $\Gamma = (\mathbf{V}, \gamma)$ is a simple projective view of $\mathbf{D}$. Say that $R[A] \in \mathsf{UProj}(\mathbf{D})$ is *visible in* Γ if there is some $R' \in \mathsf{Rels}(\mathbf{V})$ which is a projection of R under γ. Call Γ *UIND-complete* if for every directed path $\rho = \langle R_1[A_1], R_2[A_2], \ldots, R_k[A_k] \rangle$ in $\mathsf{UGraph}(\mathbf{D})$, if $R_1[A_1]$ and $R_k[A_k]$ are visible in Γ, so too are all intermediate elements in ρ.

The following result, analogous to 5.3, may now be established.

5.7 Proposition *Suppose that* $\mathbf{D}$ *is constrained by UINDs, and that* Γ *is a simple projective view which is UIND-complete. Then* Γ *is strongly monotonic.*

PROOF: The proof is a similar to that of 5.3. The central idea is to observe that all connections between values which are forced by the UINDs are already visible in the view, in the sense that they are deducible from the constraints of the view instance, and so it is completely decidable within the view whether or not an insertion or deletion will violate a UIND. The details are left to the reader. □

5.8 Proposition *Suppose that* $\mathbf{D}$ *is constrained by FDs and UINDs, and that* Γ *is a simple projective view which is both FD-complete and UIND-complete. Then* Γ *is strongly monotonic.*

PROOF: The proof follows from 5.3 and 5.7, together with the classical result that FDs and UINDs have trivial interaction [CKV90]. □

5.9 Theorem *Suppose that* $\mathbf{D}$ *is constrained by FDs and weakly acyclic UINDs, and that* Γ *is a simple projective view which is both FD-complete and UIND-complete. Then every insertion request* u *from* Γ *to* $\mathbf{D}$ *admits a* $\langle \Upsilon_{\mathfrak{C}_u}, \uparrow \rangle$*-optimal realization.*

PROOF: Combine 3.22, 4.17, and 5.8. □

6 Optimal Reflection of Deletions

At first glance, the information-based modelling of deletions to views would appear to be much simpler than that for insertions. Indeed, in large part, the only relevant information-monotone family is $\mathsf{GrAtoms}(\mathbf{D})$. Thus, the following is immediate.

6.1 Observation *Let* $u = (M_1, N_2)$ *be a deletion request from* Γ *to* $\mathbf{D}$, $(M_1, M_2) \in \mathsf{DelRealiz}\langle (M_1, N_2), \Gamma \rangle$, *and let* $K = \mathsf{ConstSym}(u)$.

(a) (M_1, M_2) *is* $\langle \Upsilon_K, \downarrow \rangle$*-admissible iff for all* $(M_1, M_2') \in \mathsf{DelRealiz}\langle (M_1, N_2), \Gamma \rangle$ *with* $M_2 \subseteq M_2'$, *it must be that* $M_2 = M_2'$.

(b) (M_1, M_2) *is* $\langle \Upsilon_K, \downarrow \rangle$*-optimal iff for all* $(M_1, M_2') \in \mathsf{DelRealiz}\langle (M_1, N_2), \Gamma \rangle$, $M_2' \subseteq M_2$. □

6.2 Admissibility and optimality for deletions As the admissibility and optimality of deletions do not depend upon the set of constant symbols in the instances defining the update, a more concise notation may be employed. For $u = (M_1, N_2)$ be a deletion request from Γ to $\mathbf{D}$ and $(M_1, M_2) \in \mathsf{DelRealiz}\langle (M_1, N_2), \Gamma \rangle$, write $\downarrow$*-admissible* as an abbreviation for $\langle \Upsilon_{\mathfrak{C}_u}, \downarrow \rangle$-admissible, and $\downarrow$*-optimal* as an abbreviation for $\langle \Upsilon_{\mathfrak{C}_u}, \downarrow \rangle$-optimal.

It is also worth observing that equality-generating dependencies and mutual-exclusion dependencies are always preserved under deletion, so no special handling of them is required.

6.3 Observation — EGHDs and mutual-exclusion TGHDs preserved under deletion *Let* φ *be either an EGHD or a mutual-exclusion TGHD on* $\mathbf{D}$. *If* $M \in \mathsf{AtMod}_{\mathfrak{I}}(\varphi)$, *then* $M' \in \mathsf{AtMod}_{\mathfrak{I}}(\varphi)$ *for every* $M' \subseteq M$. □

Despite these simplifications, the optimal support of deletions is far from trivial. The root of the problem is that while TGDs are well suited for insertions, they display inherent disjunction in the context of deletion. A simple example will help illustrate.

6.4 Example — Strong monotonicity does not ensure $\downarrow$-optimal realizations Let $\mathbf{E}_9$ be the relational schema with three relation symbols $R[A]$, $S[A]$, and $T[A]$, constrained by the single TGHD $(\forall x)((R(x) \wedge S(x)) \Rightarrow T(x))$, let $\mathbf{G}_9$ be the schema whose single relation symbol is $T'[A]$, and let $\Omega_9 = (\mathbf{G}_9, \omega_9)$ be the view of $\mathbf{G}_9$ with $\omega_9^{T'} = T(x_A)$. In words, the view Ω_9 preserves $T[A]$ (as $T'[A]$) but discards $R[A]$ and $S[A]$ completely.

Clearly Ω_9 is strongly monotonic. However, it does not always admit reflections which are $\downarrow$-optimal. Indeed, let $M_1 = \{R(\mathrm{a}_0), R(\mathrm{a}_1), S(\mathrm{a}_0), T(\mathrm{a}_0)\}$ be the current instance of $\mathbf{E}_9$, so

that $N_1 = \{T'(\mathrm{a}_0)\}$ is the instance of $\mathbf{G}_9$. Let $N_2 = \emptyset$. For $M_2' = \{R(\mathrm{a}_0), R(\mathrm{a}_1)\}$ and $M_2'' = \{S(\mathrm{a}_0), R(\mathrm{a}_1)\}$, each of (M_1, M_2') and (M_1, M_2'') are $\downarrow$-admissible realizations of $u = (M_1, N_2)$ with respect to $\mathsf{WFS}(\mathbf{E}_9, \exists\wedge+)$, and so neither is $\downarrow$-optimal.

Observe that for $M_3 = M_2' \cap M_2'' = \{R(\mathrm{a}_1)\}$, $(M_1, M_3) \in \mathsf{DelRealiz}\langle(M_1, N_2), \Omega_9\rangle$ as well, although it is not minimal. This leads to a weaker "minimax"-style of optimality, in which every tuple which is deleted in some $\downarrow$-admissible realization is deleted. This is formalized as follows.

6.5 Weak $\downarrow$-optimality Let $u = (M_1, N_2)$ be a deletion request from Γ to $\mathbf{D}$.

(a) Define $\mathsf{WeakOpt}_\downarrow\langle u, \Gamma\rangle = \bigcap\{M_3 \mid (M_1, M_3) \in \mathsf{DelRealiz}\langle(M_1, N_2), \Gamma\rangle \text{ and } \downarrow\text{-admissible}\}$.

(b) Call $(M_1, M_2) \in \mathsf{DelRealiz}\langle(M_1, N_2), \Gamma\rangle$ *weakly $\downarrow$-optimal* if $M_2 = \mathsf{WeakOpt}_\downarrow\langle u, \Gamma\rangle$.

In 6.4 above, M_3 is weakly $\downarrow$-optimal but not $\downarrow$-optimal in the sense of 6.2.

To identify conditions under which a weakly $\downarrow$-optimal reflections are admitted, it useful to introduce a new way of viewing the combination of a schema and a view.

6.6 The combined schema induced by a view Rather than regarding the main schema $\mathbf{D}$ and the view schema $\mathbf{V}$ as distinct, it is quite possible to combine them into a single schema, with the view mappings regarded as additional constraints. The alternative representation turns out to be very useful, since all constraints, both those of the schemata and those induced by the view mappings, may be considered at once.

To formalize this idea, it is necessary to assume that $\mathsf{Rels}(\mathbf{D}) \cap \mathsf{Rels}(\mathbf{V}) = \emptyset$. This is not a problem since relations may always be renamed as necessary.

The *combined schema* $\mathsf{CombSch}\langle\mathbf{D}, \Gamma\rangle$ has as its relational symbols $\mathsf{Rels}(\mathbf{D}) \cup \mathsf{Rels}(\mathbf{V})$. The constraints of $\mathsf{CombSch}\langle\mathbf{D}, \Gamma\rangle$ are those in $\mathsf{Constr}(\mathbf{D})$, together with, for each $R \in \mathsf{Rels}(\mathbf{V})$, the *definitional constraint*

$$\text{(DefC)} \qquad (\forall x_{A_1})(\forall x_{A_2}) \ldots (\forall x_{A_m})((R(x_{A_1}, x_{A_2}, \ldots, x_{A_n})) \Leftrightarrow \gamma^R$$

In the above, $\{x_{A_1}, x_{A_2}, \ldots, x_{A_m}\}$ are precisely the attribute variables which occur in the interpretation formula γ^R. It is easier to see the full nature of this constraint when γ^R is expanded into its full form $(\exists x_1)(\exists x_2) \ldots (\exists x_n)(\overline{\gamma}^R(x_{A_1}, x_{A_2}, \ldots, x_{A_m}, x_1, x_2, \ldots, x_n))$. (See 2.6 for a clarification of the notation $\overline{\gamma}^R$.) The complete expansion then becomes

$$\text{(DefC}') \quad (\forall x_{A_1})(\forall x_{A_2}) \ldots (\forall x_{A_m})((R(x_{A_1}, x_{A_2}, \ldots, x_{A_n})) \Leftrightarrow (\exists x_1)(\exists x_2) \ldots (\exists x_n)(\overline{\gamma}^R))(x_{A_1}, x_{A_2}, \ldots, x_{A_m}, x_1, x_2, \ldots, x_n)))$$

The definitional constraint (DefC′) for R may be broken into the *forward component*

$$\text{(DefC-Fwd)} \quad (\forall x_{A_1})(\forall x_{A_2}) \ldots (\forall x_{A_m})(\forall x_1)(\forall x_2) \ldots (\forall x_n) (\overline{\gamma}^R(x_{A_1}, x_{A_2}, \ldots, x_{A_m}, x_1, x_2, \ldots, x_n) \Rightarrow (R(x_1, x_2, \ldots, x_n)))$$

and the *reverse component*

$$\text{(DefC-rev)} \quad (\forall x_{A_1})(\forall x_{A_2}) \ldots (\forall x_{A_m})((R(x_{A_1}, x_{A_2}, \ldots, x_{A_n})) \Rightarrow (\exists x_1)(\exists x_2) \ldots (\exists x_n)(\overline{\gamma}^R))(x_{A_1}, x_{A_2}, \ldots, x_{A_m}, x_1, x_2, \ldots, x_n)))$$

It is easy to see that both the forward and the reverse components are TGHDs. Thus, if $\mathsf{Constr}(\mathbf{D})$ has a cover consisting of GHDs, so too does $\mathsf{Constr}(\mathsf{CombSch}\langle\mathbf{D},\Gamma\rangle)$.

This idea has already been illustrated in 1.2 with $\mathbf{E}'_0$ the combined schema associated with $\mathsf{CombSch}\langle\mathbf{E}_0,\Pi_{AB}^{\mathbf{E}_0}\rangle$. The single definitional constraint is $(\forall x)(\forall y)(R_{AB}(x,y)\Leftrightarrow(\exists z)(R(x,y,z)))$, and this decomposes into the forward constraint $(\forall x)(\forall y)(\forall z)(R(x,y,z)\Rightarrow R_{AB}(x,y))$ and the reverse constraint $(\forall x)(\forall y)(R_{AB}(x,y)\Rightarrow(\exists z)(R(x,y,z)))$. Note that the forward constraint is always universal.

6.7 Universal pairs Call the pair $\langle\mathbf{D},\Gamma\rangle$ *universal* if $\mathsf{Constr}(\mathsf{CombSch}\langle\mathbf{D},\Gamma\rangle)$ consists entirely of universal GHDs. In other words, this means that both the constraints of $\mathbf{D}$ and the view interpretation mappings consist of total dependencies, without any existential quantification. For example, in 6.4, the pair $\langle\mathbf{E}_9,\Omega_9\rangle$ is universal.

Under this assumption of universality, weakly $\downarrow$-optimal solutions exist whenever a solution which is a deletion is possible.

6.8 Proposition *Let $\langle\mathbf{D},\Gamma\rangle$ be a universal pair. Then for every deletion request $u=(M_1,N_2)$ from Γ to $\mathbf{D}$ with $\mathsf{DelRealiz}\langle u,\Gamma\rangle\neq\emptyset$, $\mathsf{WeakOpt}_{\downarrow}\langle u,\Gamma\rangle$ is a weak $\downarrow$-optimal realization of u.*

PROOF: Let $(M_1,M_2)\in\mathsf{DelRealiz}\langle(M_1,N_2),\Gamma\rangle$ and let $t\in N_2$. Since $\langle\mathbf{D},\Gamma\rangle$ is a universal pair, $\mathsf{Subst}(\gamma,t\to)$ must be a conjunction of ground atoms, and these ground atoms must be in every M for which $t\in\gamma(M)$. In particular, each such conjunct must be in $\bigcap\{M_3\mid(M_1,M_3)\in\mathsf{DelRealiz}\langle(M_1,N_2),\Gamma\rangle$ and $\downarrow$-admissible$\}$. Thus $\gamma(\mathsf{WeakOpt}_{\downarrow}\langle u,\Gamma\rangle)=N_2$.

It remains to verify that $\mathsf{WeakOpt}_{\downarrow}\langle u,\Gamma\rangle\in\mathsf{LDB}(\mathbf{D})$. However, it is a very easy exercise to show that all universal Horn sentences (and hence all total TGDs) are preserved under intersection, whence the result. □

6.9 Example — Lack of weak $\downarrow$-optimal realizations It is natural to conjecture that the result of 6.8 extends to situations involving existential quantification. Unfortunately, this is not the case. If non-total TGDs are allowed, relatively simple examples of deletions requests exist which admit $\downarrow$-admissible realizations but no weak $\downarrow$-optimal realization.

Let $\mathbf{E}_{10}$ be the schema with three relational symbols $R[AB]$, $S[BC]$, and $T[BC]$, with the following three constraints.

$$(\forall x_1)(\forall x_2)(\forall x_3)((R(x_1,x_2)\wedge R(x_1,x_3))\Rightarrow S(x_2,x_3))$$
$$(\forall x_1)(\forall x_2)(R(x_1,x_2)\Rightarrow T(x_2,x_2))$$
$$(\forall x_1)(T(x_1,x_1)\Rightarrow(\exists y_1)(R(y_1,x_1)))$$

Let $\Omega_{10}=(\mathbf{G}_{10},\omega_{10})$ be the view which retains the relations of S and T, but discards R. Let $M_1=\{R(\mathrm{a}_0,\mathrm{b}_0),R(\mathrm{a}_0,\mathrm{b}_1),R(\mathrm{a}_1,\mathrm{b}_0),R(\mathrm{a}_1,\mathrm{b}_1),S(\mathrm{b}_0,\mathrm{b}_0),S(\mathrm{b}_0,\mathrm{b}_1),S(\mathrm{b}_1,\mathrm{b}_0),S(\mathrm{b}_1,\mathrm{b}_1),T(\mathrm{b}_0,\mathrm{b}_0),T(\mathrm{b}_1,\mathrm{b}_1)\}$. It is easy to see that $M_1\in\mathsf{LDB}(\mathbf{E}_{10})$. The corresponding view instance $\omega_{10}(M_1)=\{S(\mathrm{b}_0,\mathrm{b}_0),S(\mathrm{b}_0,\mathrm{b}_1),S(\mathrm{b}_1,\mathrm{b}_0),S(\mathrm{b}_1,\mathrm{b}_1),T(\mathrm{b}_0,\mathrm{b}_0),T(\mathrm{b}_1,\mathrm{b}_1)\}$. Let the desired new view instance be $N_2=\{S(\mathrm{b}_0,\mathrm{b}_0),S(\mathrm{b}_1,\mathrm{b}_1),T(\mathrm{b}_0,\mathrm{b}_0),T(\mathrm{b}_1,\mathrm{b}_1)\}$. Thus, the tuples in $P=\{S(\mathrm{b}_1,\mathrm{b}_0),S(\mathrm{b}_0,\mathrm{b}_1)\}$ are to be deleted. It is easy to see that this update admits two $\downarrow$-admissible realizations, one which deletes $P\cup\{R(\mathrm{a}_0,\mathrm{b}_0),R(\mathrm{a}_1,\mathrm{b}_1)\}$ and the other which deletes

$P \cup \{R(\mathrm{a}_0, \mathrm{b}_1), R(\mathrm{a}_1, \mathrm{b}_0)\}$. Thus, the two possibilities for the new instance of $\mathbf{E}_{10}$ are $M_2' = \{R(\mathrm{a}_0, \mathrm{b}_1), R(\mathrm{a}_1, \mathrm{b}_0), S(\mathrm{b}_0, \mathrm{b}_0), S(\mathrm{b}_1, \mathrm{b}_1), T(\mathrm{b}_0, \mathrm{b}_0), T(\mathrm{b}_1, \mathrm{b}_1)\}$ and $M_2'' = \{R(\mathrm{a}_0, \mathrm{b}_0), R(\mathrm{a}_1, \mathrm{b}_1), S(\mathrm{b}_0, \mathrm{b}_0), S(\mathrm{b}_1, \mathrm{b}_1), T(\mathrm{b}_0, \mathrm{b}_0), T(\mathrm{b}_1, \mathrm{b}_1)\}$. However, $M_2' \cap M_2'' = \{S(\mathrm{b}_0, \mathrm{b}_0), S(\mathrm{b}_1, \mathrm{b}_1), T(\mathrm{b}_0, \mathrm{b}_0), T(\mathrm{b}_1, \mathrm{b}_1)\} \notin \mathsf{LDB}(\mathbf{E}_{10})$. Indeed, the existential quantification has introduced an exclusive-or requirement in which $(R(\mathrm{a}_0, \mathrm{b}_0) \wedge R(\mathrm{a}_1, \mathrm{b}_1)) \vee (R(\mathrm{a}_0, \mathrm{b}_1) \wedge R(\mathrm{a}_1, \mathrm{b}_0))$ must hold, but both disjuncts must not hold simultaneously. Furthermore, it is not difficult to see that Ω_{10} reflects deletions. Therefore, a general approach which addresses the disjunction problem for deletions seems impossible.

It should be noted that $\mathsf{Constr}(\mathbf{E}_{10})$ is not typed in the sense of [Fag82, p. 955]. For this paper, whether such an example is possible with such typed constraints is left as an open question. Rather, attention is turned to a more restrictive but nevertheless useful class which does admit full $\downarrow$-optimal realizations.

6.10 Unit-head TGDs, schemata, and views A TGHD of the form (GHD) of 3.15 is called a *unit-head dependency* if $n = 1$; that is, if there is only one atom on the left-hand side of the rule. The most important example of a unit-head dependency is the inclusion dependency. The schema $\mathbf{D}$ is called *unit-head* if $\mathsf{Constr}(\mathbf{D})$ has a cover consisting of EGHDs, mutual-exclusion TGHDs, and unit-head TGHDs.

This idea extends to combined schemata as well. Call the pair $\langle \mathbf{D}, \Gamma \rangle$ *unit head* if $\mathsf{CombSch}\langle \mathbf{D}, \Gamma \rangle$ is unit head. Clearly, $\mathsf{CombSch}\langle \mathbf{D}, \Gamma \rangle$ is unit head iff $\mathbf{D}$ has that property and, for each $R \in \mathsf{Rels}(\mathbf{V})$, the interpretation formula $\overline{\gamma}^R$ consists of a single (not necessarily ground) atom.

All INDs, are unit-head. Furthermore, views which are defined via projection and selection (but not join), are also unit head. Therefore, many practical examples are represented under this class.

The following lemma identifies a critical property of unit-head schemata.

6.11 Lemma *Let $\mathbf{D}$ be a unit-head schema, and let $M, M_1, M_2 \in \mathsf{LDB}(\mathbf{D})$ with $M_i \subseteq M$ for $i \in \{1, 2\}$. Then $M_1 \cup M_2 \in \mathsf{LDB}(\mathbf{D})$ as well.*

PROOF: Without loss of generality, assume that $\mathsf{Constr}(\mathbf{D})$ itself consists of EGDs and unit-head TGHDs. Let $\varphi \in \mathsf{Constr}(\mathbf{D})$ be a TGHD and let s be a constant substitution into the universal variables of φ such that $\mathsf{LHSinfo}\langle \varphi, s \rangle \in M_1 \cup M_2$. Let $i \in \{1, 2\}$ for which $\mathsf{LHSinfo}\langle \varphi, s \rangle \in M_i$. There must then be a tuple $t \in M_i$ which satisfies $\mathsf{RHSinfo}\langle \varphi, s \rangle$, since $M_i \in \mathsf{LDB}(\mathbf{D})$. Thus $M_1 \cup M_2 \in \mathsf{AtMod}_{\mathfrak{I}}(\varphi)$, and so $M_1 \cup M_2$ satisfies every TGHD in $\mathsf{Constr}(\mathbf{D})$. Since $M_1 \cup M_2 \subseteq M$, it follows from 6.3 that $M_1 \cup M_2 \in \mathsf{AtMod}_{\mathfrak{I}}(\psi)$ for every EGHD and mutual-exclusion TGHD in $\mathsf{Constr}(\mathbf{D})$. Thus, $M_1 \cup M_2 \in \mathsf{LDB}(\mathbf{D})$. □

6.12 Theorem *Let $\langle \mathbf{D}, \Gamma \rangle$ be a unit-head pair. Then every deletion request $u = (M_1, N_2)$ from Γ to $\mathbf{D}$ for which $\mathsf{DelRealiz}\langle (M_1, N_2), \Gamma \rangle \neq \emptyset$ admits a unique $\downarrow$-optimal realization.*

PROOF: Given a deletion request $u = (M', N_2)$ from Γ to $\mathbf{D}$, let M_1' and M_2' be $\downarrow$-admissible realizations of u, and let $M_i = M_i' \cup N_2 \in \mathsf{LDB}(\mathsf{CombSch}\langle \mathbf{D}, \Gamma \rangle)$ for $i \in \{1.2\}$. Now just apply 6.11 to establish that $M_1 \cup M_2 \in \mathsf{LDB}(\mathbf{D})$. It thus follows that $M_1 = M_2$, else one of M_1 and

M_2 would not be tuple minimal. Hence the part of M_1 which corresponds to the relations of $\mathbf{D}$ must be the $\downarrow$-optimal solution. □

In the context of the traditional dependencies and view constructions of the relational model, this theorem leads to the following corollary.

6.13 Corollary *Let* $\mathbf{D}$ *be constrained by FDs and INDs, and let* Γ *be defined by projections and selections on the relations of* $\mathbf{D}$. *Then, if* Γ *reflects deletions, every deletion request from* Γ *to* $\mathbf{D}$ *admits a unique* $\downarrow$*-optimal realization.* □

Invoking 5.8 yields the following more focused characterization.

6.14 Corollary *Let* $\mathbf{D}$ *be constrained by FDs and weakly acyclic INDs, and let* Γ *be a simple-projective view which is strongly monotonic. Then every unidirectional update request (i.e., insertion request or deletion request)* u *from* Γ *to* $\mathbf{D}$ *admits a unique optimal realization (*$\langle \Upsilon_{\mathfrak{C}_u}, \uparrow \rangle$*-optimal or* $\downarrow$*-optimal, as the case may be).* □

7 Conclusions and Further Directions

A strategy for the optimal reflection of view updates has been developed, based upon the concept of least information change. The property of strong monotonicity — that view insertions may always be reflected as main-schema insertions and view deletions may always be reflected as main-schema deletions, has been shown to be critical. Under this assumption, and in the context of generalized Horn dependencies, it has been shown that optimal insertions are supported in a reasonable fashion — they are unique up to a renaming of the newly-inserted constants. It has furthermore been shown that optimal deletions are supported under unit-head conditions. Nonetheless, a number of issues remain for future investigation. Among the most important are the following.

Deletion beyond the unit-head context The theory for deletions developed in Section 6 is largely restricted to unit-head pairs. It would be useful to extend these results to a wider class of schemata. As noted at the end of 6.9, it is not known (at least to the author) whether weak $\downarrow$-optimality may be obtained for typed GHDs. This topic warrants further investigation.

Characterization of strong monotonicity for wider classes of schema constraints The characterization in Section 5 of views which are strongly monotonic is limited to simple projections constrained by FDs and UINDs. Since strong monotonicity is central to the support of optimal updates, an investigation into broader characterization would certainly be worthwhile.

Optimization of tuple modification Although the general formulation applies to all types of updates, the results focus almost entirely upon insertions and deletions. Modification of single tuples ("updates" in SQL), on the other hand, are of fundamental importance. With the standard update classification pair introduced in 4.4 and used throughout the paper, only very special cases admit optimal solutions. The difficulty arises from the fact that the framework, which is based entirely upon information content, cannot distinguish between the

process of modifying a tuple and that of deleting it and then inserting a new one. Consequently, both appear as admissible updates, but neither is optimal relative to the other. Further work must therefore look for a way to recapture the distinction between tuple modification and a delete-insert pair.

Application to database components This investigation began as an effort to understand better how updates are propagated between database components, as forwarded in [Heg08b, Sec. 4], but then took on a life of its own as it was discovered that the component-based problems were in turn dependent upon more fundamental issues. Nevertheless, it is important to return to the roots of this investigation — database components. This includes not only the purely autonomous case, as sketched in [Heg08b, Sec. 4], but also the situation in which users cooperate to achieve a suitable reflection, as introduced in [HS07]

Relationship to work in logic programming As already noted in the introduction, the problem of view update has also been studied extensively in the context of deductive databases. The connection between update preference based upon distance measures, as identified in Section 1 and the current approach beg a rapprochement. In addition, the connection between the current work and that of identifying algorithms for finding all possible reflections [BM04] is of interest. Furthermore, some recent work has introduced the idea of using active constraints to establish a preference order on admissible updates [GSTZ03]. Thus, rather than employing a preference based upon information content, one based upon explicit rules is employed. The relationship between such approaches and that of this paper warrants further investigation. Also, there has been a substantial body of work on updates to disjunctive deductive databases [FGM96], in which the extensional database itself consists of a collection of alternatives. The approach of minimizing information change in the disjunctive context deserves further attention as well.

Acknowledgment Much of this research was carried out while the author was a visitor at the Information Systems Engineering Group at the University of Kiel. He is indebted to Bernhard Thalheim and the members of the group for the invitation and for the many discussions which led to this work. Also, the anonymous reviewers made numerous suggestions which (hopefully) have led to a more readable presentation. Thanks are also due to Peggy Schmidt who read the manuscript and made many valuable suggestions.

References

[AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu, *Foundations of Databases*, Addison-Wesley, 1995.

[ADB07] Ofer Arieli, Marc Denecker, and Maurice Bruynooghe, "Distance semantics for database repair," *Ann. Math. Artif. Intell.*, **50**(2007), pp. 389–415.

[ADNB06] Ofer Arieli, Marc Denecker, Bert Van Nuffelen, and Maurice Bruynooghe, "Computational methods for database repair by signed formulae," *Ann. Math. Artif. Intell.*, **46**(2006), pp. 4–37.

[BS81] François Bancilhon and Nicolas Spyratos, "Update semantics of relational views," *ACM Trans. Database Systems*, **6**(1981), pp. 557–575.

[BV84] Catriel Beeri and Moshe Y. Vardi, "A proof procedure for data dependencies," *J. Assoc. Comp. Mach.*, **31**(1984), pp. 718–741.

[BM04] Andreas Behrend and Rainer Manthey, "Update propagation in deductive databases using soft stratification," in: Georg Gottlob, András A. Benczúr, and János Demetrovics, eds., *Advances in Databases and Information Systems, 8th East European Conference, ADBIS 2004, Budapest, Hungary, September 22-25, 2004, Proceesing*, pp. 22–36, Volume 3255 of Lecture Notes in Computer Science, Springer-Verlag, 2004.

[BL97] Fadila Bentayeb and Dominique Laurent, "Inversion de l'algèbre relationnelle et mises à jour," Technical Report 97-9, Université d'Orléans, LIFO, 1997.

[BL98] Fadila Bentayeb and Dominique Laurent, "View updates translations in relational databases," in: *Proc. DEXA '98, Vienna, Sept. 24-28, 1998*, pp. 322–331, 1998.

[CGT90] Stefano Ceri, Georg Gottlob, and Letizia Tanca, *Logic Programming and Databases*, Springer-Verlag, 1990.

[CL73] Chin-Liang Chang and Richard Char-Tung Lee, *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, 1973.

[CKV90] Stavros Cosmadakis, Paris C. Kannelakis, and Moshe Y. Vardi, "Polynomial-time implication problems for unary inclusion dependencies," *J. Assoc. Comp. Mach.*, **37**(1990), pp. 15–46.

[DB82] Umeshwar Dayal and Philip A. Bernstein, "On the correct translation of update operations on relational views," *ACM Trans. Database Systems*, **8**(1982), pp. 381–416.

[DG84] William F. Dowling and Jean H. Gallier, "Linear-time algorithms for testing the satisfiability of propositional Horn clauses," *J. Logic Programming*, **3**(1984), pp. 267–284.

[EM97] Thomas Eiter and Heikki Mannila, "Distance measures for point sets and their computation," *Acta Inf.*, **34**(1997), pp. 109–133.

[Fag82] Ronald Fagin, "Horn clauses and database dependencies," *J. Assoc. Comp. Mach.*, **29**(1982), pp. 952–985.

[FKMP05] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa, "Data exchange: Semantics and query answering," *Theoret. Comput. Sci.*, **336**(2005), pp. 89–124.

[FV83] Ronald Fagin and Moshe Y. Vardi, "Armstrong databases for functional and inclusion dependencies," *Info. Process. Lett.*, **16**(1983), pp. 13–19.

[FGM96] José Alberto Fernández, John Grant, and Jack Minker, "Model theoretic approach to view updates in deductive databases.," *J. Automated Reasoning*, **17**(1996), pp. 171–197.

[GN87] Michael R. Genesereth and Nils J. Nilsson, *Logical Foundations of Artificial Intelligence*, Morgan-Kaufmann, 1987.

[GSTZ03] Sergio Greco, Cristina Sirangelo, Irina Trubitsyna, and Ester Zumpano, "Preferred repairs for inconsistent databases," in: *7th International Database Engineering and Applications Symposium (IDEAS 2003), 16-18 July 2003, Hong Kong, China*, pp. 202–211, IEEE Computer Society, 2003.

[Heg04] Stephen J. Hegner, "An order-based theory of updates for closed database views," *Ann. Math. Art. Intell.*, **40**(2004), pp. 63–125.

[Heg08a] Stephen J. Hegner, "Information-optimal reflections of view updates on relational database schemata," in: Sven Hartmann and Gabriele Kern-Isberner, eds., *Foundations of Information and Knowledge Systems: Fifth International Symposium, FoIKS 2008, Pisa, Italy, February 11-15, 2008, Proceedings*, pp. 112–131, Volume 4932 of Lecture Notes in Computer Science, Springer-Verlag, 2008.

[Heg08b] Stephen J. Hegner, "A model of database components and their interconnection based upon communicating views," in: Hannu Jakkola, Yashui Kiyoki, and Takehiro Tokuda, eds., *Information Modelling and Knowledge Systems XIX*, pp. 79–100, Frontiers in Artificial Intelligence and Applications, IOS Press, 2008.

[Heg08c] Stephen J. Hegner, "Semantic bijectivity and the uniqueness of constant-complement updates in the relatiional context," in: Klaus-Dieter Schewe and Bernhard Thalheim, eds., *International Workshop on Semantics in Data and Knowledge Bases, SDKB 2008, Nantes, France, March 29, 2008, Proceedings*, pp. 172–191, Volume 4925 of Lecture Notes in Computer Science, Springer-Verlag, 2008.

[HS07] Stephen J. Hegner and Peggy Schmidt, "Update support for database views via cooperation," in: Yannis Ioannis, Boris Novikov, and Boris Rachev, eds., *Advances in Databases and Information Systems, 11th East European Conference, ADBIS 2007, Varna, Bulgaria, September 29 - October 3, 2007, Proceedings*, pp. 98–113, Volume 4690 of Lecture Notes in Computer Science, Springer-Verlag, 2007.

[HS73] Horst Herrlich and George. E. Strecker, *Category Theory*, Allyn and Bacon, 1973.

[Hor51] Alfred Horn, "On sentences which are true of direct unions of algebras," *J. Symbolic Logic*, **16**(1951), pp. 14–21.

[Hut97] Alan Hutchinson, "Metrics on terms and clauses," in: Maarten van Someren and Gerhard Widmer, eds., *Machine Learning: ECML-97, 9th European Conference on Machine Learning, Prague, Czech Republic, April 23-25, 1997, Proceedings*, pp. 138–145, Volume 1224 of Lecture Notes in Computer Science, 1997.

[JAK82] Barry E. Jacobs, Alan R. Aronson, and Anthony C. Klug, "On interpretations of relational languages and solutions to the implied constraint problem," *ACM Trans. Database Systems*, **7**(1982), pp. 291–315.

[Kel85] Arthur M. Keller, "Updating relational databases through views," PhD thesis, Stanford University, 1985.

[Lan90] Rom Langerak, "View updates in relational databases with an independent scheme," *ACM Trans. Database Systems*, **15**(1990), pp. 40–66.

[Llo87] John W. Lloyd, *Foundations of Logic Programming, Second Extended Edition*, Springer-Verlag, 1987.

[Mak87] Johann A. Makowsky, "Why Horn formulas matter in computer science: Initial structures and generic examples," *J. Comput. System Sci.*, **34**(1987), pp. 266–292.

[Mon76] J. Donald Monk, *Mathematical Logic*, Springer-Verlag, 1976.

[NC97] Shan-Hwei Nienhuys-Cheng, "Distance between Herbrand interpretations: A measure for approximations to a target concept," in: *Inductive Logic Programming, 7th International Workshop, ILP-97, Prague, Czech Republic, September 17-20, 1997, Proceedings*, pp. 213–226, Volume 1297 of Lecture Notes in Computer Science, Springer, 1997.

[PDGV89] Jan Paredaens, Paul De Bra, Marc Gyssens, and Dirk Van Gucht, *The Structure of the Relational Database Model*, Springer-Verlag, 1989.