Foundations of Canonical Update Support for Closed Database Views †

Stephen J. Hegner
Department of Computer Science and Electrical Engineering
Votey Building
University of Vermont
Burlington, VT 05405 U.S.A.

Telephone: (802)656-3330
Internet: hegner@uvm.edu
UUCP: ..{!uunet}!uvm-gen!hegner

Abstract

A *closed view* of a database schema is one which is totally encapsulated. Insofar as the user is concerned, the view *is* the database schema. The rest of the database system is not visible through the view, and is is not required for complete use of the view. Similarly, the updates which may be effected through the view have their scope limited entirely to that view. In this paper, we lay the mathematical foundations for the systematic support of such views. The proper context is shown to be that of update translation under constant *meet complement*, a refinement of the constant complement strategy of Bancilhon and Spyratos. The central complexity result for relational schemata is that checking the legality of updates is "infinitely" simpler than blindly checking that the new state is legal for the view schema, and in the particular case that the base schema is constrained by functional dependencies, may always be performed in constant time, even if the view schema is not finitely axiomatizable. We further establish that, under very natural assumptions, update strategies for closed views are unique.

This paper appeared in the *Proceedings of the Third International Conference on Database Theory (ICDT90), Paris, 12–14 December 1990*, Springer-Verlag Lecture Notes in Computer Science, Volume 470, pp. 422–436.

†The research reported herein was performed while the author was visiting the Department of Mathematics of the University of Oslo, Norway. He wishes to thank in particular the members of the Computational Linguistics Group for their kind hospitality during his stay there.

0. Introduction

The capability to perform updates has always been an integral part of database systems, although only recently has the problem begun to be addressed systematically [Abi88]. The ability to perform updates through views — windows on the database which allow only partial access — is also a useful service of a complete database system. It is the stated goal of this research is to begin a development of a *systematic* theory of updates to views. Since there has been a great quantity of published work on the topic of supporting view updates in the past fifteen years, it is appropriate for us to begin by with a justification of why yet another paper on view updates is appropriate, and, more generally, why a new *theory* of view updates is needed.

Views may be (at least roughly) divided into two distinct classes. An *open view* is designed by the user as a "window", primarily for his own convenience. The user of such a view will typically have knowledge of and privileges to the entire database schema, or at least a substantial part of it. Proper use of such a view *requires* knowledge of the larger supporting schema. A *closed view*, on the other hand, is provided by the system administration to the user, the latter having no knowledge of the total system schema beyond that provided through the view. In this case, insofar as possible, the view should appear and behave as just another schema. Use of such a view must require *no* knowledge of the larger schema.

Most of the published work on the topic of supporting view updates is oriented towards the support of open views. The earliest efforts to systematically address the problem included that of Dayal and Bernstein [DB78, DB82] and Furtado *et al* [FSdS79]. Perhaps the most visible proponent of update strategies for open views has been Keller [Kel82, Kel85, Kel84, Kel87]. Other work along these lines include [MT85] and [Mas84]. Generally, these works look at "most plausible" strategies for restricted cases, such as updating projections or join of relations governed by functional dependencies. The resulting updates typically have effects outside of the scope of the view, and indeed, the usual arguments for their appropriateness depends upon the nature of these external effects. Open views are perhaps best supported by a sort of "toolkit" which the user may employ to construct and customize a view, and much of the work identified above could well provide the foundation for such a package. Indeed, Keller [Kel85] has already made a proposal for such a package, and Medieros and Tompa [MT85] have implemented a package to understand various update policies.

In contrast to such a "toolkit" approach for open views, we forward the thesis that for closed views there is a core of systematic principles which is applicable, regardless of the particular application. They must be adhered to in any design, and will hopefully lead to algorithmic procedures (where possible) for the design of closed views with certain specified properties. These principles must protect and support the interests of both the user and the system. From the user's perspective, a closed view should look like a complete schema, in at least the following senses:

(u1) The definition of what constitutes an admissible update to the view must depend only upon aspects of the total schema which are visible through the view. It must not depend in any way upon any part of the state of the overall schema which is not visible through the view.

(u2) The permissible updates should, insofar as possible, be axiomatizable and specifiable in a manner similar to that employed in the base schema.

From the system's perspective, a closed view must behave as an "isolated unit", in at least the following senses.

(s1) The effect of a view update must be limited to the view and its "logical consequences" within the total schema. The user of the view must not be able to change information in the database which is not visible through the view. In other words, the effects of the view, as a modifiable entity, must be suitably encapsulated.

(s2) The way in which updates are reflected back into the total schema must be in a canonical fashion, independent of arbitrary choices.

The central thrust of this paper is to provide a formalization of these four requirements. As a starting point, we observe that in the literature, the sole body of work which addresses update support in closed views, at least to some degree, is that based upon the constant-complement strategy of Bancilhon and Spyratos [BS81b, BS81a]. This includes subsequent work by Cosmadakis and Papadimitriou [CP84], Gottlob, Paolini, and Zicari [GPZ88], and our own earlier work [Heg84]. Because we start by postulating properties of an update strategy rather than by fixing a strategy itself, our approach does not presuppose constant-complement update as a goal unto itself. However, we do conclude that it is a *necessary* component of any good strategy, although not a sufficient one. We shall make more precise the relationship between our work and these references as we proceed to identify the features of our theory.

The paper is organized as follows. In Section 1, we provide more formal background material for the discussion of our ideas, including a precise definition of an acceptable update strategy for a closed view. We then address three key issues, each in its own section. In Section 2, we examine the *realizability* of update strategies which satisfy conditions (u1) and (s1). The key result shows closed strategies to be be equivalent to the "constant-meet" strategy, which is a refinement of the constant-complement strategy of Bancilhon and Spyratos. Our results, in the same spirit as those of Bancilhon and Spyratos, are completely general and require no special structure on database schemata and views. In Section 3, we confine our attention to the relational model in order to adequately address condition (u2) above. That is, we study the *axiomatizability* of admissible updates, or, more precisely, the complexity of the axioms which specify the legal updates to the view. In Section 4 we address requirement (s2) by identifying *uniqueness* conditions under which the update translation depends only upon the view, and not upon the selection of any other parameter. Finally, the work reported here admittedly only scratches the surface. To identify first principles without clouding the issues with more complex details, we work with a very simple model. In Section 5, we identify the most important next directions, relative to the current thrust of the general theory of database updates.

We assume familiarity with the by now traditional notation and terminology of the relational model, as may be found in [Mai83] and [PDGV89]. For Sections 3 and 4, we also assume some basic familiarity with first-order logic, as may be found in [End72] or [Gal86].

Due to space limitations, it has been necessary to omit many details and essentially all proofs. Complete reports containing a more detailed development, including proofs, will be available.

1. Set-Based Schemata and Views

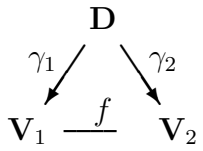
Basic Concepts

1.1 Set-Based schemata and views A *set-based database schema* $\mathbf{D}$ is entirely defined by a set of *legal databases* (or *legal states*), which we denote by $\mathsf{LDB}(\mathbf{D})$.

Let $\mathbf{D}_1$ and $\mathbf{D}_2$ be set-based database schemata. A *morphism* $f : \mathbf{D}_1 \to \mathbf{D}_2$ is just a function $f' : \mathsf{LDB}(\mathbf{D}_1) \to \mathsf{LDB}(\mathbf{D}_2)$. The reason for this apparently redundant notation is for compatibility with the relational framework to be introduced later.

Let $\mathbf{D}$ be a set-based database schema. A *set-based view* of $\mathbf{D}$ is a pair $\Gamma = (\mathbf{V}, \gamma)$ in which $\mathbf{V}$ is a set-based database schema and $\gamma : \mathbf{D} \to \mathbf{V}$ is a set-based morphism with the property that $\gamma' : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$ is surjective. The *congruence* $\mathsf{Congr}(\Gamma)$ of Γ (denoted $\equiv_\Gamma$ in [BS81b]) is the equivalence relation on $\mathsf{LDB}(\mathbf{D})$ defined by $(M_1, M_2) \in \mathsf{Congr}(\Gamma)$ iff $\gamma'(M_1) = \gamma'(M_2)$.

Given set-based views $\Gamma_1 = (\mathbf{V}_1, \gamma_1)$ and $\Gamma_2 = (\mathbf{V}_2, \gamma_2)$, a *view morphism* $f : \Gamma_1 \to \Gamma_2$ is a set-based database morphism $f : \mathbf{V}_1 \to \mathbf{V}_2$ such that the following diagram commutes.



1.2 Lemma *Let* $\mathbf{D}$ *be a set-based database schema, and let* $\Gamma_1 = (\mathbf{V}_1, \gamma_1)$ *and* $\Gamma_2 = (\mathbf{V}_2, \gamma_2)$ *be set-based views of* $\mathbf{D}$*. Then there is at most one set-based view morphism* $\Gamma_1 \to \Gamma_2$*. This morphism exists iff* $\mathsf{Congr}(\Gamma_1) \subseteq \mathsf{Congr}(\Gamma_2)$*.* □

1.3 View interpolation and equivalence The morphism guaranteed by the above lemma is of sufficient importance to warrant a special notation. When it exists, we denote the unique f which makes the above diagram commute by $\lambda(\Gamma_1, \Gamma_2)$. This furthermore allows us to regard Γ_2 as a view of $\mathbf{V}_1$. We call this new view the *relativization* of Γ_2 to Γ_1 and denote it by $\Lambda(\Gamma_1, \Gamma_2) = (\mathbf{V}_2, \lambda(\Gamma_1, \Gamma_2))$.

If $\Gamma_1 = (\mathbf{V}_1, \gamma_1)$ and $\Gamma_2 = (\mathbf{V}_2, \gamma_2)$ are set-based views such that there are morphisms $f : \Gamma_1 \to \Gamma_2$ and $g : \Gamma_2 \to \Gamma_1$, then the above uniqueness result guarantees that $g \circ f : \Gamma_1 \to \Gamma_1$ and $f \circ g : \Gamma_2 \to \Gamma_2$ are identity morphisms. Thus, in the standard categorical sense [HS73, 5.13], f and g are isomorphisms. We say that Γ_1 and Γ_2 are *(set-based) isomorphic* in this case. It is trivial to verify that Γ_1 and Γ_2 are isomorphic iff $\mathsf{Congr}(\Gamma_1) = \mathsf{Congr}(\Gamma_2)$. We write $[\Gamma_1]$ to denote the equivalence class of all views which are (set-based) isomorphic to Γ_1. Upon identifying isomorphic views, 1.2 guarantees that view morphism induces a partial order on equivalence classes. As a convenient notation, we write $[\Gamma_2] \leq [\Gamma_1]$ just in case there is a

morphism $f : \Gamma_1 \to \Gamma_2$. In an abstract decomposition theory, we do not distinguish between equivalent views, and, as an abuse of notation, we also write $\Gamma_2 \leq \Gamma_1$.

1.4 Single-relation schemata and projective views Although we do not formally work with the relational model until Section 3, it is nonetheless useful to draw upon simple examples before then. We therefore introduce some notation at this point. We assume familiarity with the notion of a single-relation schema $R[U]$ with attribute set U, as may be found in [Mai83] or [PDGV89]. Let $\mathbf{D}$ denote the schema with single relation $R[U]$, constrained by some set of first-order dependencies Φ. Let $U_1 \subseteq U$. Then $\mathbf{D}_{U_1}$ denotes the single-relation schema whose relation symbol is $R_{U_1}[U_1]$, and whose constraints are those projected from Φ onto the set U_1. (In the examples we consider, these projected constraints will always be first order.) $\pi_{U_1} : \mathbf{D} \to \mathbf{D}_{U_1}$ denotes the database mapping whose underlying function is just the usual projection $\pi_{U_1}{}' : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{D}_{U_1})$, and Π_{U_1} denotes the view $(\mathbf{D}_{U_1}, \pi_{U_1})$.

1.5 Example Let $\mathbf{E}$ denote the single relation schema of three attributes $R[ABC]$, constrained by the join dependency $\bowtie[AB, BC]$ and the functional dependency $B \to C$. $\mathsf{LDB}(\mathbf{E})$ is just the set of all databases which satisfy the join dependency $\bowtie[AB, BC]$. In Π_{AB}, $\mathbf{E}_{AB}$ is defined by the single relation symbol $R_{AB}[AB]$; there are no nontrivial constraints on this schema. The projective views $\Pi_{BC} = (\mathbf{E}_{BC}, \pi_{BC})$ and $\Pi_B = (\mathbf{E}_B, \pi_B)$ are defined similarly, noting that $B \to C$ is a constraint of $\mathbf{E}_{BC}$. We may regard these as set-based schemata and views by "forgetting" the relational structure, and working with the underlying $\mathsf{LDB}(-)$'s. We clearly have that $\Pi_B \leq \Pi_{AB}$ and $\Pi_B \leq \Pi_{AB}$. The morphism $\lambda(\Gamma_{AB}, \Gamma_B)$ is just the projection of R_{AB} onto R_B. In other words, in this instance 1.2 just says that we may factor the projection π_B on R_{ABC} through R_{AB}.

Complements and Their Characterization

1.6 Notational convention Unless otherwise noted, throughout the rest of this section, we let $\mathbf{D}$ denote an arbitrary set-based database schema and $\{\Gamma_1, \Gamma_2\}$ a pair of set-based views of $\mathbf{D}$, with $\Gamma_i = (\mathbf{V}_i, \gamma_i)$.

1.7 The decomposition morphism and types of decompositions The *product schema* $\mathbf{V}_1 \times \mathbf{V}_2$ has $\mathsf{LDB}(\mathbf{V}_1 \times \mathbf{V}_2) = \mathsf{LDB}(\mathbf{V}_1) \times \mathsf{LDB}(\mathbf{V}_2)$. In other words, we just take the cartesian product of the corresponding sets of legal states. The *decomposition morphism* $\Delta\langle\{\Gamma_1, \Gamma_2\}\rangle : \mathbf{D} \to \mathbf{V}_1 \times \mathbf{V}_2$ has as underlying function $\Delta\langle\{\Gamma_1, \Gamma_2\}\rangle' : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V}_1) \times \mathsf{LDB}(\mathbf{V}_2)$, given on elements by $M \mapsto (\gamma_1{}'(M), \gamma_2{}'(M))$.

We say that $\{\Gamma_1, \Gamma_2\}$ forms a *subdirect complementary pair* if the decomposition map $\Delta\langle\{\Gamma_1, \Gamma_2\}\rangle$ is injective, and Γ_1 and Γ_2 are then called *subdirect complements* of one another. If the decomposition map is furthermore bijective, the pair $\{\Gamma_1, \Gamma_2\}$ is called a *direct complementary pair*, and Γ_1 and Γ_2 are called *direct complements* of one another.

In [Heg89], we have investigated extensively the properties of direct complementary pairs, and, more generally, direct decompositions into any number of finite components. Such

decompositions are the natural ones to consider when studying the problem of decomposing a schema for purposes of simplifying the logical and/or physical structure. However, it is subdirect complements which arise naturally in the study of update strategies, and so we focus upon their properties here.

We have borrowed the adjectives *direct* and *subdirect* from the field of universal algebra. The interested reader is invited to compare our definitions with those of *direct product* and *subdirect product* as given in [Gra68]. Note also that in [BS81a, BS81b], the term *complement* is used to define what we call a subdirect complement.

1.8 The constrained product Let $\{\Gamma_1, \Gamma_2\}$ be a subdirect complementary pair. The schema whose set of legal states is $\{(M_1, M_2) \in \mathsf{LDB}(\mathbf{V}_1) \times \mathsf{LDB}(\mathbf{V}_2) \mid (\exists M \in \mathsf{LDB}(\mathbf{D}))\ ((\gamma_1'(M), \gamma_2'(M)) = (M_1, M_2))\}$ is called the *constrained product* of Γ_1 and Γ_2, and is denoted by $\mathbf{V}_1 \otimes \mathbf{V}_2$. It is immediate that $\mathsf{LDB}(\mathbf{V}_1 \otimes \mathbf{V}_2) = \Delta\langle\{\Gamma_1, \Gamma_2\}\rangle(\mathsf{LDB}(\mathbf{D}))$. When we restrict the codomain of $\Delta\langle\{\Gamma_1, \Gamma_2\}\rangle$ to $\mathbf{V}_1 \otimes \mathbf{V}_2$, we get the *relativized decomposition map* $\gamma_1 \otimes \gamma_2 : \mathsf{LDB}(\mathbf{D}) \to \mathbf{V}_1 \otimes \mathbf{V}_2$. Clearly, if $\{\Gamma_1, \Gamma_2\}$ forms a subdirect complementary pair, then $(\gamma_1 \otimes \gamma_2)'$ is a bijection. In this case, the inverse $(\gamma_1 \otimes \gamma_2)^{-1} : \mathbf{V}_1 \otimes \mathbf{V}_2 \to \mathbf{D}$ is called the *reconstruction map* for $\{\Gamma_1, \Gamma_2\}$.

Strictly speaking, the notation $\mathbf{V}_1 \otimes \mathbf{V}_2$ is ambiguous, since the actual definition of this schema depends upon the base schema $\mathbf{D}$ and the view mappings γ_1 and γ_2. However, to keep from becoming buried in overly complex notation, we shall stick to this simpler notation, and let the context clarify the details.

1.9 Example of subdirect decomposition We continue with the example begun in 1.5. The pair $\{\Pi_{AB}, \Pi_{BC}\}$ forms a subdirect decomposition of $\mathbf{E}$, but not a direct decomposition. $\mathsf{LDB}(\mathbf{E}_{AB} \otimes \mathbf{E}_{BC})$ consists of exactly those pairs $\{(M, N) \in \mathsf{LDB}(\mathbf{E}_{AB}) \times \mathsf{LDB}(\mathbf{E}_{BC}) \mid \lambda(\Pi_{AB}, \Pi_B)'(M) = \lambda(\Pi_{BC}, \Pi_B)'(N)\}$. In other words, the legal states are precisely those pairs whose B projections agree. The reconstruction map $(\pi_{AB} \otimes \pi_{BC})'$ is none other than the join on AB and BC.

1.10 Fully commuting views and meet complements We say that $\{\Gamma_1, \Gamma_2\}$ is a *fully commuting pair* if $\mathsf{Congr}(\Gamma_1) \circ \mathsf{Congr}(\Gamma_2) = \mathsf{Congr}(\Gamma_2) \circ \mathsf{Congr}(\Gamma_1)$, with "$\circ$" denoting ordinary relational composition. A subdirectly complementary pair $\{\Gamma_1, \Gamma_2\}$ which is fully commuting is called a *meet complementary pair*, and Γ_1 and Γ_2 are called *meet complements* of one another.

1.11 Example – not every subdirect complementary pair is fully commuting The views Π_{AB} and Π_{BC} of 1.5 are fully commuting, as is easily verified. To provide an example of noncommuting views, let $\mathbf{F}$ be the same as $\mathbf{E}$ of 1.5, save that we add the functional dependency $A \to C$ to the schema. We still have the subdirect decomposition $\{\Pi_{AB}, \Pi_{BC}\}$, but now the congruences do not commute. Indeed, let $M = \{(a_1, b_1, c_1), (a_2, b_2, c_2)\}$, $N = \{(a_1, b_1, c_1), (a_2, b_2, c_1)\}$, $P = \{(a_1, b_1, c_1), (a_1, b_2, c_1)\}$, with all values distinct. Then it is easy to see that $(M, N) \in \mathsf{Congr}(\Pi_{AB})$ and $(N, P) \in \mathsf{Congr}(\Pi_{BC})$, so that $(M, P) \in \mathsf{Congr}(\Pi_{AB}) \circ \mathsf{Congr}(\Pi_{BC})$. On the other hand, suppose that $(M, P) \in \mathsf{Congr}(\Pi_{BC}) \circ \mathsf{Congr}(\Pi_{AB})$. Then

$(P, M) \in \mathsf{Congr}(\Pi_{AB}) \circ \mathsf{Congr}(\Pi_{BC})$ as well, so there must be a Q such that $(P, Q) \in \mathsf{Congr}(\Pi_{AB})$ and $(Q, M) \in \mathsf{Congr}(\Pi_{BC})$. But then for (P, Q) to be in $\mathsf{Congr}(\Pi_{AB})$ it must be of the form $\{(a_1, b_1, x), (a_1, b_2, x)\}$ for some x. Then, for (Q, M) to be in $\mathsf{Congr}(\Pi_{AB})$, M must be of the form $\{(y_1, b_1, x), (y_2, b_2, x)\}$ for some y_1, y_2, x, which it is not. Thus these views are not fully commuting.

The notion of a meet complement is not a particularly intuitive by itself. However, it does have some rather nice characterizations in terms of certain types of independence-characterizing general dependencies.

1.12 Generalized dependencies Let $\{\Gamma_1, \Gamma_2\}$ be a subdirect complementary pair.

(a) The $\{\Gamma_1, \Gamma_2\}$-*reconstruction dependency* on $\mathbf{V}_1 \otimes \mathbf{V}_2$, denoted $\otimes[\Gamma_1, \Gamma_2]$, is satisfied iff for any $M_1, N_1 \in \mathsf{LDB}(\mathbf{V}_1)$ and $M_2, N_2 \in \mathsf{LDB}(\mathbf{V}_2)$, if any three of the elements of $\{(M_1, M_2), (M_1, N_2), (N_1, M_2), (N_1, N_2)\}$ is in $\mathbf{V}_1 \otimes \mathbf{V}_2$, then so too is the fourth.

(b) Let $\Gamma = (\mathbf{V}, \gamma)$ be a set-based view of $\mathbf{D}$, with $\Gamma \leq \Gamma_1$ and $\Gamma \leq \Gamma_2$. The Γ-*independence dependency* on $\mathbf{V}_1 \otimes \mathbf{V}_2$, denoted $\otimes_\Gamma$, is satisfied iff for any $M_1 \in \mathsf{LDB}(\mathbf{V}_1)$ and $M_2 \in \mathsf{LDB}(\mathbf{V}_2)$,

$$((M_1, M_2) \in \mathsf{LDB}(\mathbf{V}_1 \otimes \mathbf{V}_2)) \Leftrightarrow (\lambda(\Gamma_1, \Gamma)'(M_1) = \lambda(\Gamma_2, \Gamma)'(M_2))$$

1.13 Theorem – characterization of meet complementary pairs *Let $\{\Gamma_1, \Gamma_2\}$ be a subdirect complementary pair. Then the following conditions are equivalent.*

(a) $\{\Gamma_1, \Gamma_2\}$ *is a meet-complementary pair.*

(b) $\mathsf{Congr}(\Gamma_1) \circ \mathsf{Congr}(\Gamma_2)$ *is an equivalence relation.*

(c) $\mathbf{V}_1 \otimes \mathbf{V}_2$ *satisfies* $\otimes[\Gamma_1, \Gamma_2]$.

(d) $\mathbf{V}_1 \otimes \mathbf{V}_2$ *satisfies* $\otimes_\Gamma$, *where* Γ *is the view (unique up to equivalence) whose congruence is the smallest equivalence relation on* $\mathsf{LDB}(\mathbf{D})$ *containing both* $\mathsf{Congr}(\Gamma_1)$ *and* $\mathsf{Congr}(\Gamma_2)$. □

1.14 Meets and Γ-complements In view of (b) above, a meet-complementary pair $\{\Gamma_1, \Gamma_2\}$ uniquely defines (up to equivalence) a view Γ whose congruence is $\mathsf{Congr}(\Gamma_1) \circ \mathsf{Congr}(\Gamma_2)$. We call this view the *meet* of Γ_1 and Γ_2, and say that $\{\Gamma_1, \Gamma_2\}$ is a Γ-*complementary pair*, and Γ_1 and Γ_2 are called Γ-complements of one another. The motivation for this terminology comes from the fact that the views of $\mathbf{D}$ may be endowed with a lattice-like structure, in which Γ is the meet of Γ_1 and Γ_2. See [Heg89, 1.3.15] for a further discussion.

1.15 Example We continue with the examples of 1.5 and 1.11. The dependency $\otimes[\Pi_{AB}, \Pi_{BC}]$ says that if we have three particular models in $\mathbf{E}_{AB} \otimes \mathbf{E}_{BC}$, then we can always build a fourth using a join-like operation. As a concrete example, let $M_1 = \{(a_1, b_1), (a_2, b_2)\}$, $N_1 = \{(a_1, b_1), (a_1, b_2)\}$, $M_2 = \{(b_1, c_1), (b_2, c_1)\}$, and $N_2 = \{(b_1, c_1), (b_2, c_2)\}$. Then, if any three of the pairs in $\{(M_1, M_2), (M_1, N_2), (N_1, M_2), (N_1, N_2)\}$ is in $\mathsf{LDB}(\mathbf{E}_{AB} \otimes \mathbf{E}_{BC})$ the

fourth is also. However, if we consider $\mathbf{F}_{AB} \otimes \mathbf{F}_{BC}$ as arising from the schema $\mathbf{F}$ which enforces the dependency $A \rightarrow C$ as well, then we have $(M_1, M_2), (M_1, N_2), (N_1, M_2) \in \mathsf{LDB}(\mathbf{F}_{AB} \otimes \mathbf{F}_{BC})$, but not (N_1, N_2), since $A \rightarrow C$ will be violated in this case. Thus $\otimes[\Pi_{AB}, \Pi_{BC}]$ does not hold on $\mathbf{F}_{AB} \otimes \mathbf{F}_{BC}$.

It is easy to see that the meet of $\{\Gamma_1, \Gamma_2\}$ for $\mathbf{E}$ is just Π_B. Thus, the dependency $\otimes[\Pi_{AB}, \Pi_{BC}]$ just says that any pair $(M_1, M_2) \in \mathsf{LDB}(\mathbf{E}_{AB}) \times \mathsf{LDB}(\mathbf{E}_{BC})$ which agree on the meet view Π_B is in $\mathsf{LDB}(\mathbf{E}_{AB} \otimes \mathbf{E}_{BC})$.

2. Updates in the Set-Based Case

Update Families and Update Strategies

2.1 Notational convention We continue to let $\mathbf{D}$ be an arbitrary set-based database schema, with $\Gamma_1 = (\mathbf{V}_1, \gamma_1)$ and $\Gamma_2 = (\mathbf{V}_2, \gamma_2)$ arbitrary set-based views of $\mathbf{D}$.

2.2 Simple update families A *simple update family* for $\mathbf{D}$ is a subset $\mathbf{U}_\mathbf{D} \subseteq \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D})$ which is reflexive and transitive. $(M_1, M_2) \in \mathbf{U}_\mathbf{D}$ just means that the update which changes the state of $\mathbf{D}$ from M_1 to M_2 is allowed. Reflexivity assures that all identity updates are allowed, and transitivity ensures that updates may be composed. Note that a simple update family need not be *complete* in the sense of [BS81b], since we do not postulate that updates be reversible (symmetry of $\mathbf{U}_\mathbf{D}$.)

Of course, in current research in the study of database transactions, the set $\mathbf{U}_\mathbf{D}$ is specified by some sort of transaction language, such as *TL*, *detTL*, and their relatives [AV90]. However, at this point, it would not serve our purpose to incorporate such a language into our model, since we are interested in general admissibility of updates, irrespective of the characteristics of specific transaction languages. Once these principles have been established, further structure may be imposed to study the impact of these differences.

2.3 Update strategies for views Let $\mathbf{U}_\mathbf{D}$ and $\mathbf{U}_{\mathbf{V}_1}$ be simple update families for $\mathbf{D}$ and $\mathbf{V}_1$, respectively. An *update strategy* for $\mathbf{U}_{\mathbf{V}_1}$ with respect to $\mathbf{U}_\mathbf{D}$ is a rule which tells us how to translate each update of $\mathbf{V}_1$ given in $\mathbf{U}_{\mathbf{V}_1}$ into an update of $\mathbf{D}$. The analog of such a rule is called a *translator* in [BS81b]; we employ an equivalent representation which is a variant of that introduced in [Heg84]. Formally, an *update strategy* for $\mathbf{U}_{\mathbf{V}_1}$ with respect to $\mathbf{U}_\mathbf{D}$ is a partial function $\rho : \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{V}_1) \rightarrow \mathsf{LDB}(\mathbf{D})$ satisfying conditions (upt1) – (upt4) below. The semantics are described as follows. Let M be the current state of the base schema $\mathbf{D}$; the current state of $\mathbf{V}_1$ must then be $\gamma_1'(M)$. Suppose that the desired update is $(\gamma_1'(M), N) \in \mathbf{U}_{\mathbf{V}_1}$. $\rho(M, N)$ gives us the new state of $\mathbf{D}$, so that $(M, \rho(M, N)) \in \mathbf{U}_\mathbf{D}$ is the *translation* of $(\gamma'(M), N) \in \mathbf{U}_{\mathbf{V}_1}$.

(upt1) For any $M \in \mathsf{LDB}(\mathbf{D})$ and $N \in \mathsf{LDB}(\mathbf{V}_1)$, whenever $\rho(M, N)$ is defined, $\gamma'(\rho(M, N)) = N$. In other words, the translation must be to a state in $\mathsf{LDB}(\mathbf{D})$ which maps to N.

(upt2) For any $M, P \in \mathsf{LDB}(\mathbf{D})$ and $N \in \mathsf{LDB}(\mathbf{V}_1)$, $\rho(M, N) = P$ implies $(M, P) \in \mathbf{U}_\mathbf{D}$. In other words, the translation must be a legal update of $\mathbf{D}$.

(upt3) For each $(M, N) \in \mathbf{U}_{\mathbf{V}_1}$, there are $P, Q \in \mathsf{LDB}(\mathbf{D})$ such that ${\gamma_1}'(P) = M$, ${\gamma_1}'(Q) = N$, and $\rho(P, N) = Q$. In other words, each update in $\mathbf{U}_{\mathbf{V}_1}$ must be realizable relative to *some* state of the base schema.

(upt4) For any $M \in \mathsf{LDB}(\mathbf{D})$ and $N \in \mathsf{LDB}(\mathbf{V}_1)$, whenever $\rho(M, N)$ is defined, $({\gamma_1}'(M), N) \in \mathbf{U}_{\mathbf{V}_1}$. In other words, ρ may support only updates which are in $\mathbf{U}_{\mathbf{V}_1}$. If the other three conditions are satisfied, we can always force this condition by suitably restricting the domain of definition of ρ.

2.4 Simplifying convention To keep the notation within reasonable bounds, from here on we shall assume that the base schema $\mathbf{D}$ always has enough updates; that is, we shall assume that $\mathbf{U}_{\mathbf{D}} = \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D})$. In this case, condition (upt2) will automatically be satisfied. This is not a serious limitation, because in the more general case, we can simply proceed as though $\mathbf{U}_{\mathbf{D}} = \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D})$ were true, and then check to see if in fact the true $\mathbf{U}_{\mathbf{D}}$ in fact has the update transition to support the required view update.

2.5 The constant complement strategy Let us assume that $\{\Gamma_1, \Gamma_2\}$ forms a subdirect complementary pair. Define the partial function $\mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle : \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{V}_1) \to \mathsf{LDB}(\mathbf{D})$ by

$$\mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle(M, N) = \begin{cases} \Delta\langle\{\Gamma_1, \Gamma_2\}\rangle'^{-1}(N, {\gamma_2}'(M)) & \text{if } (N, {\gamma_2}'(M)) \in \mathsf{LDB}(\mathbf{V_1} \otimes \mathbf{V_2}); \\ \text{undefined} & \text{otherwise.} \end{cases}$$

We call $\mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle$ the *full strategy with constant subdirect complement* Γ_2. Of course, this is nothing more than constant complement translation in the sense of Bancilhon and Spyratos [BS81b]. We define $\mathbf{U}\langle\Gamma_1, \Gamma_2\rangle$ to be precisely the domain of definition of $\mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle$.

2.6 Functoriality and implicit reversibility Let us now introduce two "niceness" properties on update strategies. First of all, if we compose two updates to the view, the translation should be the composition of the individual translations. Additionally, the identity update to the view should translate to the identity update on the base schema. Formally, an update strategy ρ for a simple update family $\mathbf{U}_{\mathbf{V}_1}$ for $\mathbf{V}_1$ is said to be *functorial* if it satisfies the following two conditions.

(f1) For any $M \in \mathsf{LDB}(\mathbf{D})$, $\rho(M, \gamma'(M))$ is defined and equals M.

(f2) For any $M_1 \in \mathsf{LDB}(\mathbf{D})$, $N_1, N_2 \in \mathsf{LDB}(\mathbf{V}_1)$, if both $\rho(M_1, N_1)$ and $\rho(\rho(M_1, N_1), N_2))$ are defined, then $\rho(M_1, N_2)$ is defined and equals $\rho(\rho(M_1, N_1), N_2))$.

We note that functoriality is implicit in the definition of a translation of Bancilhon and Spyratos [BS81b, Def. 3.3].

The condition of *implicit reversibility* stipulates that, after an update, if we know the *new* state of the base schema $\mathbf{D}$ and the *old* state of the view schema $\mathbf{V}_1$ (as well as the update strategy), that should be enough to recover the old state of $\mathbf{D}$. In other words, no update to $\mathbf{V}_1$ should change the state of $\mathbf{D}$ in a way which is not totally encoded in the state of $\mathbf{V}_1$. Formally, ρ is *implicitly reversible* if

(ir) For any $M_1, M_2 \in \mathsf{LDB}(\mathbf{D})$, $N \in \mathsf{LDB}(\mathbf{V}_1)$, if both $\rho(M_1, N)$ and $\rho(M_2, N)$ are defined with $\rho(M_1, N) = \rho(M_2, N)$ and $\gamma'(M_1) = \gamma'(M_2)$, then $M_1 = M_2$.

Our motivation for requiring implicit reversibility is to recapture condition (s1). Intuitively, if an update strategy is not implicitly reversible, then some update can alter the state of $\mathbf{D}$ in such a way that we cannot observe through knowledge of the change in view state alone. Conversely, if it is implicitly reversible, then nothing which is not determined completely by the view state can change through a view update.

Because a constant complement strategy in the sense of Bancilhon and Spyratos is defined to be *complete* [BS81b, Def. 3.2], it is not only implicitly reversible, but in fact explicitly so.

We may now state the classic result of Bancilhon and Spyratos [BS81b, Sec. 7], recast within our notational framework.

2.7 Theorem – realizability of update strategies via constant complement *Let $\mathbf{U}_{\mathbf{V}_1}$ be a simple update family for $\mathbf{V}_1$, and let ρ be an update strategy on Γ_1 for $\mathbf{U}_{\mathbf{V}_1}$. Then there exists a subdirect complement Γ_2 of Γ_1 such that $\rho \subseteq \mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle$ if and only if ρ is functorial and implicitly reversible. If $\mathbf{U}_{\mathbf{V}_1}$ is furthermore symmetric, then we may choose Γ_2 such that $\rho = \mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle$.* □

2.8 Uniform updatability and closed update strategies A major shortcoming of condition (upt3) is that it is existential – whether or not we can actually realize an update $(M, N) \in \mathbf{U}_{\mathbf{V}_1}$ depends upon the particular state of $\mathbf{D}$. In a closed view, this state is not known to the user, and so, in general, a constant-complement update strategy will violate condition (u1). To rectify this, we postulate the stronger condition of *uniformity*, which stipulates that we can always realize the update $(M, N) \in \mathbf{U}_{\mathbf{V}_1}$. Formally, we say that ρ is *uniform* if the following condition is satisfied.

(un) For any $(M, N) \in \mathbf{U}_{\mathbf{V}_1}$ and $P \in \mathsf{LDB}(\mathbf{D})$ with $\gamma_1{}'(P) = M$, $\rho(P, N)$ is defined. In other words, all of the updates in $\mathbf{U}_{\mathbf{V}_1}$ are supported, regardless of the state of $\mathbf{D}$.

We call an update strategy ρ which is uniform a *closed update strategy*. Before establishing the formal properties of such a strategy, we show by example that not all update strategies are uniform.

2.9 Example – non-uniform updatability We return to the context established in 1.11. Specifically, the base schema is $\mathbf{F}$. Let $M = \{(a_1, b_1, c_1), (a_2, b_2, c_2)\}$ and $P = \{(a_1, b_1, c_1), (a_2, b_2, c_1)\}$, as in 1.11. Then, in either case, the state of the view schema $\mathbf{F}_{AB}$ is $\{(a_1, b_1), (a_2, b_2)\}$. Suppose that we want to update that state to $\{(a_1, b_1), (a_1, b_2)\}$. If the state of the schema is is M, then this update is not possible with constant complement Π_{BC}, as the functional dependency $A \to C$ will be violated. On the other hand, if the state of $\mathbf{F}$ is P, then the update is quite possible with this constant complement; the new state is $N = \{(a_1, b_1, c_1), (a_1, b_2, c_1)\}$. Thus, whether or not we can effect this update with constant complement Π_{BC} depends upon the specific state of $\mathbf{F}_{BC}$, so the corresponding strategy

cannot be uniform. Note that if the base schema were instead $\mathbf{E}$, so that we are not required to enforce the dependency $A \to C$, then updating with constant Π_{BC} is uniform.

It turns out that adding to the constant complement strategy the stipulation that the complement be a meet complement is exactly what is needed to ensure a closed update strategy. More precisely, we have the following refinement of 2.7.

2.10 Theorem – characterization of uniform updatability *Let $\mathbf{U}_{\mathbf{V}_1}$ be a simple update family for $\mathbf{V}_1$, and let ρ be an update strategy on Γ_1 for $\mathbf{U}_{\mathbf{V}_1}$. Then there exists a meet complement Γ_2 of Γ_1 such that $\rho \subseteq \mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle$ iff ρ is functorial, implicitly reversible, and uniform, i.e., iff ρ is a closed update strategy. Furthermore, as in 2.7, if $\mathbf{U}_{\mathbf{V}_1}$ is symmetric, then we may choose Γ_2 such that $\rho = \mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle$, and, in this case, $\mathbf{U}_{\mathbf{V}_1} = \{(M, N) \in \mathsf{LDB}(\mathbf{V}_1) \mid \lambda(\Gamma_1, \Gamma)'(M) = \lambda(\Gamma_1, \Gamma)'(N)\}$, where Γ is the meet of Γ_1 and Γ_2. In other words, the allowed updates are precisely those which keep Γ constant.* □

2.11 Remarks on the literature Work on the problem of characterizing meet-like conditions in terms of commuting congruences in the setting of universal algebra goes back to at least [Fle55]. In the database context, Bancilhon and Spyratos addressed the issue of determining the conditions under which update admissibility is independent of the complement state in their VLDB paper on independence [BS81a]. While their presentation is sketchy, they do recapture some of the ideas we have presented here with their notion of *weak independence*. However, they seem to have been unaware of the commuting congruences characterization, and to have missed the key point that the meet is uniquely defined by the complementary views, if it exists. In our own earlier paper [Heg84], we worked with the special case of Γ-complements in which Γ is the trivial view whose schema has exactly one legal state, so that the complements are direct. This made for interesting theory, but it is far too constraining to support realistic updates. Finally, Gottlob, Paolini, and Zicari [GPZ88] generalize the notion of constant-complement update to *decreasing-complement* update. While this generalization does not preserve the notion of implicit reversibility, the relationship between our work and theirs nonetheless warrants further study.

3. The Logical Structure of Subdirect Decomposition

We now turn our attention ensuring condition (s2). To obtain meaningful results, we must look beyond the simple set-based context and work with a data model which admits abstract axiomatization. The natural choice is the relational model.

The General Relational Case

3.1 Relational schemata and views We have already used simple relational schemata and views in several examples. However, for a more systematic investigation using the special properties of the relational model, we must establish some additional notation. A relational schema $\mathbf{D}$ consists of a finite set of relation symbols $\mathsf{Rel}(\mathbf{D})$, each such $R \in \mathsf{Rel}(\mathbf{D})$ with

a unique positive arity (number of columns) $\mathsf{Ar}(R)$, as well as a set of constraints $\mathsf{Con}(\mathbf{D})$. Unless further stipulations are made, constraints are taken to be arbitrary sentences in the first-order language defined by $\mathsf{Rel}(\mathbf{D})$. $\mathsf{LDB}(\mathbf{D})$ denotes the set of all legal databases of $\mathbf{D}$; that is, the models of $\mathsf{Con}(\mathbf{D})$, while $\mathsf{DB}(\mathbf{D})$ denotes the set of all databases (structures) in the language of $\mathbf{D}$, whether or not they are models of $\mathsf{Con}(\mathbf{D})$. Given a set of constraints Φ with the property that $\mathsf{Con}(\mathbf{D}) \models \varphi$ for each $\varphi \in \Phi$, we write $\mathsf{Mod}(\Phi)$ to denote the set of all elements of $\mathsf{DB}(\mathbf{D})$ which satisfy each element of Φ (*i.e.*, the models of Φ). A set of constraints Φ with the property that $\mathsf{Mod}(\Phi) = \mathsf{Con}(\mathbf{D})$ is called a *basis* for $\mathsf{Con}(\mathbf{D})$.

One point which does require some clarification is the definition of domains for the columns of the relations. We have advocated elsewhere [Heg89] the use of Boolean algebra of types, in lieu of the more traditional disjoint domains. In general, our results apply in this more general setting. However, to avoid introducing the rather copious additional necessary to support this extended framework, we shall simply work with the more traditional and familiar one here. In any case, the extensions necessary are largely notational, and do not provide any essential new insights within this context.

A relational view *is* a set-based view, but with additional structure. Precisely, a *relational view* of the relational schema $\mathbf{D}$ is a pair $\Gamma = (\mathbf{V}, \gamma)$ in which $\mathbf{V}$ is also a relational schema and $\gamma : \mathbf{D} \to \mathbf{V}$ is a relational database mapping whose underlying mapping $\gamma' : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$ is surjective. This differs from the set-based definition of 1.1 in that we now work with relational morphisms. Specifically, the relational morphism $\gamma : \mathbf{D} \to \mathbf{V}$ associates with each $R \in \mathsf{Rel}(\mathbf{V})$ a first-order formula $\mathsf{Def}(\gamma, R)$ in the language of the schema $\mathbf{D}$ with (by convention) exactly the variables in $\{v_1, \ldots v_{\mathsf{Ar}(R)}\}$ free. The instance of R defined by the formula (or query) $R(v_1, \ldots, v_{\mathsf{Ar}(R)}) \Leftrightarrow \mathsf{Def}(\gamma, R)$, gives rise to γ' in the natural fashion. (See [JAK82] for more on this idea.) In general, given a function $\gamma' : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$, there are many different choices of $\{\mathsf{Def}(\gamma, R) \mid R \in \mathsf{Rel}(\mathbf{D})\}$ which define it. We call such representations *equivalent*.

Because we are dealing with the subject of update checking, we must be able to work with states which do not satisfy all of the integrity constraints. Specifically, the set of formulas $\{\mathsf{Def}(\gamma, R) \mid R \in \mathsf{Rel}(\mathbf{D})\}$ also gives rise to a function $\gamma^* : \mathsf{DB}(\mathbf{D}) \to \mathsf{DB}(\mathbf{V})$. We use the same formulas in $\{\mathsf{Def}(\gamma, R) \mid R \in \mathsf{Rel}(\mathbf{D})\}$ to define the function on the larger domain.

3.2 Example Just to make sure that there is no confusion, let us solidify these definitions with the example of 1.5. We have $\mathsf{Rel}(\mathbf{E}) = \{R\}$, $\mathsf{Con}(\mathbf{E}) = \{\bowtie [AB, BC], B \to C\}$ $\mathsf{Rel}(\mathbf{E}_{BC}) = \{R_{AB}\}$, $\mathsf{Con}(\mathbf{E}_{BC}) = \{B \to C\}$, and $\mathsf{Def}(\pi_{BC}, R)$ is the formula $(\exists z)(R(z, v_1, v_2))$. An equivalent morphism is given by f, with $\mathsf{Def}(f, R_{BC})$ the formula $(\exists z)(\forall x, y, z, u, w)$ $(R(x, y, z) \wedge R(u, y, w) \Rightarrow (z = w)) \wedge R(z, v_1, v_2))$. It just embeds $B \to C$ into the view definition; this makes no difference on legal databases, so $f' = \pi_{BC}{}'$. However, $f^* \neq \pi^*_{BC}$. Indeed, if $M \in \mathsf{DB}(\mathbf{E})$ is any database which does not satisfy $B \to C$, then $f^*(M)$ is the empty relation, while $\pi^*_{BC}(M)$ is the usual BC projection.

3.3 Notational convention Unless otherwise noted, for the rest of this subsection, we assume that $\mathbf{D}$ is an arbitrary relational schema, and that $\Gamma_1 = (\mathbf{V}_1, \gamma_1)$, $\Gamma_2 = (\mathbf{V}_2, \gamma_2)$, and $\Gamma = (\mathbf{V}, \gamma)$ are relational views of $\mathbf{D}$.

3.4 Lemma – lifting of 1.2 to the relational case *The statement of 1.2 continues to hold in the relational setting. Specifically, we may replace each occurrence of "set-based" with "relational", and retain a valid statement. In particular, if* $\mathsf{Congr}(\Gamma_1) \subseteq \mathsf{Congr}(\Gamma_2)$*, the view* $\Lambda(\Gamma_1, \Gamma_2) = (\mathbf{V}_2, \lambda(\Gamma_1, \Gamma_2))$ *is well defined as a relational view of* $\mathbf{V}_1$*.*

PROOF OUTLINE: This is relatively straightforward application of Beth's definability theorem for first-order logic [Gal86, 6.6.2]. □

3.5 The update admissibility problem and relative axiomatization Let $\{\Gamma_1, \Gamma_2\}$ be a Γ-complementary pair. The $\langle\Gamma_1, \Gamma\rangle$-*update-admissibility problem* is that of deciding, for a given $(M, N) \in \mathsf{LDB}(\mathbf{V}_1) \times \mathsf{DB}(\mathbf{V}_1)$ with $\lambda(\Gamma_1, \Gamma)^*(M) = \lambda(\Gamma_1, \Gamma)^*(N)$, whether or not $(M, N) \in \mathbf{U}\langle\Gamma_1, \Gamma_2\rangle$. Clearly, this amounts to deciding whether or not $N \in \mathsf{LDB}(\mathbf{V}_1)$. But since we must verify that $\lambda(\Gamma_1, \Gamma)^*(M) = \lambda(\Gamma_1, \Gamma)^*(N)$ anyway, it behoves us to utilize this information when checking whether $N \in \mathsf{LDB}(\mathbf{V}_1)$. In terms of constraints, we say that a set Φ of constraints on $\mathbf{V}_1$ with $\mathsf{Con}(\mathbf{V}_1) \models \Phi$ is a Γ-*relative axiomatization* of $\mathbf{V}_1$ if every $M \in \mathsf{DB}(\mathbf{V}_1)$ which is in $\mathsf{Mod}(\Phi)$ with $\lambda(\Gamma_1, \Gamma)^*(M) \in \mathsf{LDB}(\mathbf{V})$ is in fact in $\mathsf{LDB}(\mathbf{V}_1)$. In other words, if we know that M maps to a legal state of $\mathbf{V}$, then to see if it is in $\mathsf{LDB}(\mathbf{V}_1)$, it suffices to verify that it is a model of Φ. (Note that equivalent representations of the same view may have different relative axiomatizations, as this definition depends upon $\lambda(\Gamma_1, \Gamma)^*$ and not just $\lambda(\Gamma_1, \Gamma)'$.)

3.6 Theorem – checking update admissibility *Assume that* $\mathbf{D}$ *is finitely axiomatizable, and that* $\{\Gamma_1, \Gamma_2\}$ *is a* Γ*-complementary pair. Then both* $\mathbf{V}_1$ *and* $\mathbf{V}_2$ *have finite* Γ*-relative axiomatizations. Thus, in particular, let* $M \in \mathsf{LDB}(\mathbf{V}_1)$ *and let* $N \in \mathsf{DB}(\mathbf{V}_2)$ *with the property that* $\lambda(\Gamma_1, \Gamma)^*(M) = \lambda(\Gamma_2, \Gamma)^*(N)$*. Then there is a finite set of constraints* Φ_1 *such that* $(M, N) \in \mathbf{U}\langle\Gamma_1, \Gamma_2\rangle$ *is true iff* $N \in \mathsf{Mod}(\Phi_1)$*.* □

3.7 Example We illustrate just how much this can simplify things via a specific example. Let $\mathbf{G}$ denote the single-relation schema $R[ABCDE]$, constrained by the dependencies $\mathsf{Con}(\mathbf{G}) = \{A \to D, BC \to D, CD \to A, A \to E, \bowtie [ABCD, ABCE]\}$. It can be shown that the projective view Π_{ABCE} is not finitely axiomatizable. Yet $\{\Pi_{ABCE}, \Pi_{ABCD}\}$ forms a meet complementary pair with meet Π_{ABC}. The Π_{ABC}-relative axiomatization of Π_{ABCE} is just $\{A \to C\}$. In other words, to solve the $\langle\Gamma_1, \Gamma\rangle$-update-admissibility problem, instead of having to check an infinite set of axioms, we need check only a single functional dependency.

Updates on Projections of a Single Relation

We now focus our attention on the special case that the base schema $\mathbf{D}$ is a single-relation schema constrained by rather simple dependencies, and all views are projections. Within this context, we are able to establish some very strong results.

3.8 Notation and the context We now let $\mathbf{D}$ be the schema whose single relation symbol is $R[U]$. Recall that an *typed equality generating dependency (TEGD)* [Fag82] (also

called a *generalized functional dependency* [Mai83, 14.7], [FV86]) is a sentence of the form

$$(\bigwedge_{i=1}^{m} R(x_{i1}, \ldots, x_{in})) \Rightarrow (x_{j_\ell} = x_{k_\ell})$$

On the left-hand side, some of the x_α's may be the same, provided they are in the same column of R. A functional dependency (FD) is just a TEGD with $m = 2$. The *order* of a TEGD φ is the number of tuples we must check at a time to verify satisfaction. More precisely, the order of φ is the least number p such that for any $M \in \mathsf{DB}(\mathbf{D})$, if each $N \subseteq M$ containing at most p tuples is in $\mathsf{LDB}(\mathbf{D})$, then so too is M. An FD has order two.

Now let $W \subseteq U$. We say that the TEGD φ (on $R[U]$) *embeds* in W if there is a TEGD ψ on $R[W]$ such that for any $M \in \mathsf{DB}(\mathbf{D})$, $M \in \mathsf{Mod}(\{\varphi\})$ iff $\pi^*_W(M) \in \mathsf{Mod}(\{\psi\})$. If $U_1, U_2 \subseteq U$ and Ψ is a set of TEGD's on $R[U]$, then an *embedded cover* of Ψ into $\{U_1, U_2\}$ is a set Φ of TEGD's on $R[U]$ such that $\mathsf{Mod}(\Psi) = \mathsf{Mod}(\Phi)$ and each $\varphi \in \Phi$ embeds in either U_1 or U_2. Clearly, this generalizes the notion of an embedded cover for a set of FD's [BH81].

3.9 Lemma *Let* $\mathbf{D}$ *be the single relation schema* $R[U]$*, constrained by a set of TEGD's plus a single join dependency* $\bowtie[U_1, U_2]$*. Let* Z *be a subset of either* U_1 *or else of* U_2*. Then there is a (possibly infinite) set of TEGD's* Φ *such that* $((R_Z[Z], \Phi), \pi_Z)$ *is a view of* $\mathbf{D}$*.*

PROOF OUTLINE: Follows directly from [Fag82, 6.1]. □

Our main characterization theorem for meet complementary pairs is the following.

3.10 Theorem *Suppose that* $\mathbf{D}$ *is the single-relation schema* $R[U]$*, and is constrained by a (finite) set* Ψ *of TEGD's of some finite order* m*, plus the join dependency* $\bowtie[U_1, U_2]$*. Then* $\{\Pi_{U_1}, \Pi_{U_2}\}$ *forms a meet complementary pair with meet* $\Pi_{U_1 \cap U_2}$ *iff there is an embedded cover of* Ψ *into* $\{U_1, U_2\}$*. In case this embedded cover does exist, it may be taken to be composed entirely of TEGD's of order* m*. In particular, if* $m = 2$*, all of the TEGD's involved reduce to FD's, so the test becomes one of deciding the existence of an embedded FD cover.* □

3.11 Corollary *Suppose that* $\mathbf{D}$ *is the single-relation schema* $R[U]$*, and is constrained by a (finite) set* Ψ *of FD's plus the join dependency* $\bowtie[U_1, U_2]$*. Then*

(a) *There is a polynomial-time algorithm (polynomial in the size of* Ψ *plus the size of* U*) to decide whether or not* $\{\Pi_{U_1}, \Pi_{U_2}\}$ *is a meet complementary pair with meet* $\Pi_{U_1 \cap U_2}$*.*

(b) *Let* $\{\Pi_{U_1}, \Pi_{U_2}\}$ *be a meet complementary pair with meet* $\Pi_{U_1 \cap U_2}$*, and let* Ω *be an embedded cover for* Ψ*. Define the following quantities:*

  (i) K *= the number of FD's in* Ω *which embed in* U_1 *but not in* $U_1 \cap U_2$*.*

  (ii) $T(n)$ *= the time required to retrieve a single tuple satisfying a partial match specification (with fields either fixed or totally unspecified) in a relation of* n *tuples.*

  (iii) *For* $(M, N) \in \mathsf{DB}(\mathbf{V}_1) \times \mathsf{DB}(\mathbf{V}_1)$*, define* $\mathsf{InsertSize}(M, N)$ *to be the number of tuples in* $N \setminus M$*.*

Then there is an algorithm for solving the $\langle \Pi_{U_1}, \Pi_{U_1 \cap U_2} \rangle$-update-admissibility problem which has worst-case time complexity of $O(K \cdot T(\mathsf{InsertSize}(M, N))$ for any given $(M, N) \in \mathsf{LDB}(\mathbf{V}_1) \times \mathsf{DB}(\mathbf{V}_1)$ (with $\lambda(\Pi_{U_1}, \Pi_{U_1 \cap U_2})^*(M) = \lambda(\Pi_{U_1}, \Pi_{U_1 \cap U_2})^*(N)$).

PROOF OUTLINE: Beeri and Honeyman [BH81] have given a polynomial time algorithm to decide whether or not a set of projections has an embedded cover, which, in view of the above theorem, gives us the algorithm for (a). For (b), we note that it suffices to check the validity of the updates against a set of FD's. This can be performed in time proportional to the size of the update, as argued in [GW86]. □

Let us remark in particular that in the case that we have an associative memory for the database which can do partial match retrieval of a single tuple in constant time (in the sense of [GW86]), then the time complexity of a single insertion or replacement is $O(K)$, which is independent of the database size. Deletions are free in this context in any case.

3.12 Example – the meet need not be defined by column intersection In the above results, we have been careful to stipulate that the meet be defined by the intersection of the columns. This is not a vacuous stipulation. Indeed, let $\mathbf{D}$ have the single relation symbol $R[ABC]$ and be constrained by $\{A \to C, C \to A, B \to AC\}$. Consider the views Π_{AB} and Π_{BC}, which clearly form a subdirect complementary pair. Their meet may be characterized by the view $\Gamma = (\mathbf{V}, \gamma)$ with $\mathbf{V}$ the single-relational schema $S[BB]$, and with $\mathsf{Def}(\gamma, S) = (\exists x, y)(R(x, v_1, y) \wedge R(x, v_2, y))$. There are no constraints on $\mathbf{V}$. Clearly $\Gamma \not\cong \Pi_B$, so the meet is not defined by column intersection.

3.13 Remarks on the literature Cosmadakis and Papadimitriou [CP84] have conducted an extensive investigation into updating projections of single-relation schemata using the constant-complement strategy, focusing on insertions. The algorithms which they present show substantially higher complexity than ours. The differences are principally due to the fact that they did not confine their attention to meet complementary pairs, as we have. Indeed, the marked complexity differences provide further support for the practical aspects of closed update strategies. A more careful comparison of these approaches is warranted, but must be deferred to another paper.

4. Uniqueness and Canonicity of Update Strategies

In the introduction, we proposed in condition (s2) that an update reflection strategy should be independent of arbitrary choices. In the general case, examples show that this is not possible. However, upon restricting our attention to the $\exists{+}\wedge$-views, which include all expressions built up from the basic operations of projection, restriction, and join, we are able to establish strong uniqueness and optimality results.

Optimality with Respect to a Fixed Meet

4.1 $\exists{+}\wedge$-views We say that a first-order formula φ is an $\exists{+}\wedge$-formula if it is of the form $(Q)(\alpha_1 \wedge \ldots \wedge \alpha_m)$, with Q a (possibly empty) sequence of existential quantifiers, and each α_i a positive literal. The relational view $\Gamma = (\mathbf{V}, \gamma)$ of $\mathbf{D}$ is called an $\exists{+}\wedge$-view if, for each $R \in \mathsf{Rel}(\mathbf{V})$, the interpretation formula $\mathsf{Def}(\gamma, R)$ is an $\exists{+}\wedge$-formula in which every v_i, $1 \leq i \leq \mathsf{Ar}(R)$, occurs in some α_i. (Recall that the full definition of R is $R(v_1, \ldots, v_{\mathsf{Ar}(R)}) \Leftrightarrow \mathsf{Def}(\gamma, R)$.) Our $\exists{+}\wedge$-views are essentially the so-called *conjunctive queries* of Chandra and Merlin [CM76]. Note that any composition of the projection, restriction, and join operators yields an $\exists{+}\wedge$-view.

We say that two relational views Γ_1 and Γ_2 are $\exists{+}\wedge$-isomorphic, and write $\Gamma_1 \cong_{[\exists+\wedge]} \Gamma_2$, if they are isomorphic in the ordinary relational sense, and, in addition, we may represent both $\Lambda(\Gamma_1, \Gamma_2)$ and $\Lambda(\Gamma_2, \Gamma_1)$ as $\exists{+}\wedge$ views. This notion of isomorphism is strictly stronger than usual first-order logical isomorphism, as illustrated in the next example.

4.2 Example Let $\mathbf{D}$ be the relational schema with two unary relational symbols $R[A]$ and $S[A]$, with no constraints other than that the two relations share the same domain. Let Σ_R denote the view which preserves $R[A]$ identically but discards $S[A]$, and define Σ_S similarly. Define $\Sigma_T = (\mathbf{S}, \sigma)$ to be the view whose schema $\mathbf{S}$ contains the single relation symbol $T[A]$ with defining formula $\mathsf{Def}(\sigma, T) = (R(v_1) \vee S(v_1)) \wedge (\neg(R(v_1) \wedge S(v_1)))$. In other words, T is the symmetric difference of R and S. Intuitively, Σ_R and Σ_S, which are $\exists{+}\wedge$-views, provide "direct" views of the schema $\mathbf{D}$, while Σ_T, which is not a $\exists{+}\wedge$-view, provides a "convoluted" one.

Now it is not difficult to see that both $\{\Sigma_R, \Sigma_S\}$ and $\{\Sigma_R, \Sigma_T\}$ are direct complementary pairs (and so, *a fortiori*, a meet complementary pair with the meet the trivial one-state view $\Gamma_\perp(\mathbf{D})$ [Heg89, 1.1.1]). Yet Σ_S and Σ_T are clearly not isomorphic. We seek to identify the *formal* way in which Σ_S a "better" complement than Σ_T. If we form the view $\Sigma_R \otimes \Sigma_T$ (use the obvious extension of 1.8 to the relational case) of $\mathbf{D}$ with relation symbols R and T, then $\Sigma_R \otimes \Sigma_T$ is isomorphic to the identity view $\Gamma_\top(\mathbf{D})$ in the usual sense, but it is not $\exists{+}\wedge$-isomorphic to it, as there is no way to compute the state of T without using negation and disjunction. Indeed, upon restricting our attention to $\exists{+}\wedge$-isomorphisms, we can conclude that subdirect complements with respect to a given meet are unique. This stands in stark contrast to the fact that in the more general case, subdirect complements are never unique, except in trivial cases [BS81b, 4.4].

4.3 Proposition – uniqueness of $\exists{+}\wedge$-complements *Let $\{\Gamma_1, \Gamma_2\}$ and $\{\Gamma_1, \Gamma_3\}$ be Γ-complementary pairs of $\exists{+}\wedge$-views. Then $\Gamma_2 \cong_{[\exists+\wedge]} \Gamma_3$. In other words, Γ-complements are unique when we restrict our attention to $\exists{+}\wedge$-views.* □

Even if we do not fix the meet, $\exists{+}\wedge$-complements must provide the same update translation when they overlap.

4.4 Corollary *Let $\{\Gamma_1, \Gamma_2\}$ and $\{\Gamma_1, \Gamma_3\}$ be meet complementary pairs of $\exists{+}\wedge$-views (with possibly different meets). Then, for any $(M, N) \in \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{V}_1)$, if both*

$\mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle(M, N)$ *and* $\mathsf{UpdStr}\langle\Gamma_1, \Gamma_3\rangle(M, N)$ *are defined, they are equal. In other words, there is only one way to translate a given update with respect to a meet* $\exists{+}\wedge$*-complement; the specific choice of complement does not affect the translation.* □

Save though updates translated through $\exists{+}\wedge$-complements are unique, one may still ask if there are "better" ways. If we measure the goodness of an update strategy by the amount of changes that it makes to the base schema, then 4.6 below establishes that the canonical update strategy is optimal.

4.5 Canonical triples and update optimality Let Γ_1 be an $\exists{+}\wedge$-view of $\mathbf{D}$, and let $(M, N, P) \in \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{V}_1) \times \mathsf{LDB}(\mathbf{D})$. We say that (M, N, P) is a *legal triple* for Γ_1 if $\gamma_1'(P) = N$, and that it is a *canonical triple* for Γ_1 if $P = \mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle$ for some Γ_2 which is a meet complement of Γ_1, as well as a $\exists{+}\wedge$-view. In view of the above corollary, if (M, N, P) and (M, N, Q) are each canonical triples, then $P = Q$.

Let us call a legal triple (M, N, P) for Γ_1 *optimal* if for any other legal triple (M, N, Q) and any $\exists{+}\wedge$-view $\Omega = (\mathbf{W}, \omega)$ of $\mathbf{D}$, if $\omega'(M) = \omega'(Q)$, then $\omega'(M) = \omega'(P)$ as well. In other words, through the eyes of arbitrary $\exists{+}\wedge$-views, P is "closer" to M than any other element of $\mathsf{LDB}(\mathbf{D})$ which maps to N.

4.6 Theorem – optimality and canonicity are equivalent *Let* Γ_1 *be a* $\exists{+}\wedge$*-view and let* (M, N, P) *be a legal triple for* Γ_1*. Then* (M, N, P) *is optimal iff it is canonical.* □

Global optimality

As a final step to the optimal closed update strategy, it is natural to ask, given a $\exists{+}\wedge$-view Γ_1, if there is a "best" meet $\exists{+}\wedge$-complement Γ_2 for which $\mathsf{UpdStr}\langle\Gamma_1, \Gamma_2\rangle$ recaptures all canonical updates. The answer is unfortunately negative, as illustrated by the following example.

4.7 Example Let $\mathbf{D}$ denote the single-relation schema of three attributes $R[ABC]$, constrained by the FD's $A \to B$, $A \to C$, $B \to C$, and $C \to B$. In other words, A is a key, and B and C determine each other. We consider the three views Π_{AB}, Π_{BC}, and Π_{AC}, each with their embedded dependencies. It is easy to see that both Π_{AB} and Π_{AC} are meet $\exists{+}\wedge$-complements of Π_{BC}, yet there is no meet complement which contains both. Therefore, we cannot recapture all canonical updates within a single setting.

5. The Next Step

In this work, we have laid down firm mathematical principles for the support of updates in closed user views. To avoid unnecessary complications in formulating first principles, we have assumed a completely abstract formulation of potential updates, in the form of simple update families. However, in actual database systems, potential updates are typically

expressed in terms of a transaction language. The obvious next step of our research will add structure to the simple update families by explicitly assuming that the possible updates are expressed by a transaction language, such as members of the *TL* or *detTL* families [AV90]. An update strategy then becomes a *program transformer*, rather than just an abstract mapping. We seek to determine the degree to which the simplicity of updatability which we have established in the abstract setting lifts to this more structured context. In particular, we are interested in establishing *algorithmically* specifiable connections between view updates and their translations. A parallel step will be to work directly with schemata and views which are specified transactionally, in the spirit of [AV89].

References

[Abi88] Abiteboul, S., "Updates, a new frontier," in: *ICDT'88, 2nd International Conference on Database Theory*, pp. 1–18, 1988.

[AV89] Abiteboul, S. and Vianu, V., "A transaction-based approach to relational database specification," *J. Assoc. Comp. Mach.*, **36**(1989), pp. 758–789.

[AV90] Abiteboul, S. and Vianu, V., "Procedural languages for database queries and updates," 1990, to appear in *J. Comput. System Sci.*

[BS81a] Bancilhon, F. and Spyratos, N., "Independent components of databases," in: *Proceedings of the Seventh International Conference on Very Large Data Bases*, pp. 398–408, 1981.

[BS81b] Bancilhon, F. and Spyratos, N., "Update semantics of relational views," *ACM Trans. Database Systems*, **6**(1981), pp. 557–575.

[BH81] Beeri, C. and Honeyman, P., "Preserving functional dependencies," *SIAM J. Computing*, **10**(1981), pp. 647–656.

[CM76] Chandra, A. K. and Merlin, P. M., "Optimal implementation of conjunctive queries in relational databases," in: *Proceedings of the 1976 ACM Symposium on the Theory of Computing*, pp. 77–90, 1976.

[CP84] Cosmadakis, S. and Papadimitriou, C., "Updates of relational views," *J. Assoc. Comp. Mach.*, **31**(1984), pp. 742–760.

[DB78] Dayal, U. and Bernstein, P. A., "On the updatability of relational views," in: *Proceedings of the Fourth International Conference on Very Large Data Bases*, pp. 368–474, 1978.

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

[End72] Enderton, H. B., *A Mathematical Introduction to Logic*, Academic Press, 1972.

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

[FV86] Fagin, R. and Vardi, M. Y., "The theory of data dependencies – a survey," in: Anshel, M. and Gewirtz, W., eds., *Mathematics of Information Processing*, pp. 19–71, American Mathematical Society, 1986.

[Fle55] Fleischer, I., "A note on subdirect products," *Acta Math. Acad. Sci. Hungar.*, **6**(1955), pp. 463–465.

[FSdS79] Furtado, A. L., Sevcik, K. C., and dos Santos, C. S., "Permitting updates through views of databases," *Information Systems*, **4**(1979), pp. 269–283.

[Gal86] Gallier, J. H., *Logic for Computer Science*, John Wiley and Sons, 1986.

[GPZ88] Gottlob, G., Paolini, P., and Zicari, R., "Properties and update semantics of consistent views," *ACM Trans. Database Systems*, **13**(1988), pp. 486–524.

[GW86] Graham, M. H. and Wang, K., "Constant time maintenance or the triumph of the fd," in: *Proceedings of the Fifth ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, pp. 202–216, 1986.

[Gra68] Grätzer, G., *Universal Algebra*, D. Van Nostrand, 1968.

[Heg84] Hegner, S. J., "Canonical view update support through Boolean algebras of components," in: *Proceedings of the Third ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, pp. 163–172, 1984.

[Heg89] Hegner, S. J., "Unique complements and decompositions of database schemata," Technical Report PC 12/ 12.89, Centro di Ricerche in Fisica e Matematica (CERFIM), Locarno, Switzerland, 1989, Submitted for publication.

[HS73] Herrlich, H. and Strecker, G. E., *Category Theory*, Allyn and Bacon, 1973.

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

[Kel82] Keller, A., "Updates to relational databases through views involving joins," in: Schueuermann, P., ed., *Improving Database Usability and Responsiveness*, pp. 363–384, Academic Press, 1982.

[Kel84] Keller, A., "Choosing a view update translator by dialog at view definition time," in: *Proceedings of the Twelfth International Conference on Very Large Data Bases*, pp. 467–474, 1984.

[Kel85] Keller, A., "Algorithms for translating view updates to database updates for views involving selections, projections, and joins," in: *Proceedings of the Fourth ACM SIGACT-SIGMOD Conference on Principles of Database Systems*, pp. 154–163, 1985.

[Kel87] Keller, A., "Comments on Bancilhon and Spyratos' "Update semantics of relational views"," *ACM Trans. Database Systems*, **12**(1987), pp. 521–523.

[Mai83] Maier, D., *The Theory of Relational Databases*, Computer Science Press, 1983.

[Mas84] Masunaga, Y., "A relational database view update translation mechanism," in: *Proceedings of the Tenth International Conference on Very Large Data Bases*, pp. 309–320, 1984.

[MT85] Medeiras, C. B. and Tompa, F. W., "Understanding the implications of view update policies," in: *Proceedings of the Eleventh International Conference on Very Large Data Bases*, pp. 316–323, 1985.

[PDGV89] Paredaens, J., De Bra, P., Gyssens, M., and Van Gucht, D., *The Structure of the Relational Database Model*, Springer-Verlag, 1989.