A constructive approach to translatability of view updates

Enrico Franconi and Paolo Guagliardo

KRDB Research Centre, Free University of Bozen-Bolzano, Italy

Abstract In this article, we revisit the view update framework by Bancilhon and Spyratos in a setting with constraints expressed in first-order logic. We explicitly consider constraints on both the database and the view schemas, along with inter-schema constraints that implicitly define the view mappings. Using the notion of logical definability we give a constructive characterisation of views, their inverses and complements, and provide an effective method for checking whether an arbitrary first-order expressible view update can be (uniquely) translated under the constant complement principle.

1 Introduction

Views are employed in a number of different areas, such as schema simplification, data integration, query optimisation, data restructuring, just to mention a few. Querying views does not pose particular problems, as it simply reduces to query unfolding. On the other hand, updating views is much harder, because it requires finding suitable ways of propagating an update from the view to the underlying database, so that by refreshing the content of the view from the updated version of the database the changes on the view are reflected exactly.

The task of "translating" a view update into a database update is non-trivial and poses several challenges. The most subtle kind of update anomaly is given by changes not directly wanted nor explicitly made by the user, originating in the view as a "side-effect". An additional difficulty is represented by the fact that there can be more database updates corresponding to a given update on the view. Lastly, yet another complication concerns updates that modify the database even though this is not required in order to reflect the changes made on the view.

Related Work

A general and precise understanding of the view update problem is due to the seminal work [1] by Bancilhon and Spyratos, who devise an abstract framework in which they formalise the problem and provide an elegant solution to it. They introduce the fundamental notion of *view complement*, representing what is missing from a view in order to have the same informative content of the underlying database. Moreover, they introduce the *constant complement* principle, stating that the changes done on a view must not influence the content of its complement during the translation process. Bancilhon and Spyratos provide no constructive characterisation of their approach, stating that "computational algorithms (if they exist) must be sought in specific problems: for example, schemata defined by functional dependencies and views derived by projections". Most of the subsequent work on view updates is centred on the framework [1] and, in particular, its application to the relational model.

Cosmadakis and Papadimitriou [4] consider a restricted setting that consists of a single database relation, in which views are just projections over the attributes of such a "universal" relation. They give necessary and sufficient conditions for the translatability of insertions, deletions and replacements under constant complement and they also study the complexity of finding a suitable complement that makes a given update translatable. Even though an effective method based on the Chase is provided for checking translatability, it is only applicable in the very limited setting consisting of a single database relation with projective views and functional and join dependencies at the database level.

In the context of SQL databases, Lechtenbörger [7,8] gives a characterisation of the constant complement principle in terms of undo operations, showing that view updates are translatable under constant complement precisely if users have the chance to undo all effects of their updates using further view updates. It is then argued that testing whether this holds could be an alternative to checking whether users can observe all effects of their updates in the view schema, but no effective method for doing so is proposed.

Gottlob et al. [5] extend the results of Bancilhon and Spyratos to the class of so-called *consistent views*, which properly contains the class of views translating under constant complement. The main difference between the two is that, during the translation of an update on a consistent view, the complement is not required to remain invariant, but it is allowed to "decrease" according to a suitable partial order. Indeed, in the case in which the partial order is the equality, the framework coincides with the one in [1]. Also in this generalised framework, no constructive characterisation and effective methods for checking the translatability of updates and computing their translations are provided.

Contribution and Outline

In [1], Bancilhon and Spyratos deal with constraints only implicitly and just at the database level. We revisit their framework by considering explicit constraints on both the database and the view schemas, as well as arbitrary inter-schema constraints defining the relationship between the two. All of the constraints are expressed as a theory in first-order logic.

In the present article, we introduce the concept of *view under constraints*, we give a constructive characterisation, based on logical definability, of such views, their inverses and complements, and we provide an effective method for testing whether a first-order expressible view update can be translated under constant complement.

The rest of the paper is organised as follows: in Sec. 2 we introduce the necessary notation and other basic definitions; in Sec. 3 we study the injectivity and surjectivity of views and characterise view updates that are uniquely translatable on views defined by injective mappings; in Sec. 4 we extend the results presented for injective views to the case of translation under constant complement; finally, we conclude in Sec. 5 by pointing out some future research directions.

2 Preliminaries

An *n-ary relation* on a set A, where $n \in \mathbb{N}$ is called the *arity* of the relation, is a subset of the Cartesian product A^n, that is, a set of n-tuples of elements of A. A *signature* consists of a finite set σ of relation symbols and a function $\mathsf{ar}\colon \sigma \to \mathbb{N}$ associating each relation symbol $r \in \sigma$ with a natural number $n = \mathsf{ar}(r)$ called the *arity* of r. For the sake of simplicity we will identify a signature with its set of relation symbols, each of which is considered to have an implicitly associated arity.

A *relational structure* $\mathcal{I}$ over a signature σ is a pair $\langle \varDelta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where $\varDelta^{\mathcal{I}}$ is a (possibly infinite) domain of objects and $\cdot^{\mathcal{I}}$ is a function that associates each symbol $r \in \sigma$ with a relation $r^{\mathcal{I}}$ on $\varDelta^{\mathcal{I}}$, called the *extension* of r, of appropriate arity (that is, if r is an n-placed relation symbol, then $r^{\mathcal{I}}$ is an n-ary relation). For every relational structure $\mathcal{I}$ with domain $\varDelta$, we assume the interpretation function $\cdot^{\mathcal{I}}$ to be from $\varDelta$ to the set $\mathcal{P}(\varDelta) \cup \mathcal{P}(\varDelta^2) \cup \cdots \cup \mathcal{P}(\varDelta^\infty)$, where $\mathcal{P}(A)$ denotes the powerset of A. Given a relational structure $\mathcal{I}$ over σ, its *restriction* to a subset σ' of σ is the relational structure $\mathcal{I}|_{\sigma'}$ obtained from $\mathcal{I}$ by restricting its interpretation function to the relation symbols in σ'. For relational structures $\mathcal{I} = \langle \varDelta, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{J} = \langle \varDelta, \cdot^{\mathcal{J}} \rangle$ over two disjoint signatures σ and σ', respectively, $\mathcal{I} \uplus \mathcal{J} = \langle \varDelta, \cdot^{\mathcal{I}} \cup \cdot^{\mathcal{J}} \rangle$ is a relational structure over $\sigma \cup \sigma'$.

A *constraint* is a closed first-order logic formula φ. We denote by $\mathsf{sig}(\varphi)$ the set of relation symbols occurring in φ and we say that φ is over a signature σ if $\mathsf{sig}(\varphi) \subseteq \sigma$. We extend $\mathsf{sig}(\cdot)$ to sets of constraints in the natural way. Given two disjoint signatures σ and σ' and a finite set $\varSigma$ of constraints over $\sigma \cup \sigma'$, we say that a relation symbol $r \in \sigma$ is *implicitly defined* by the relation symbols in σ' under $\varSigma$ if and only if, for every two $\varSigma$-models $\mathcal{I}$ and $\mathcal{J}$ such that $\varDelta^{\mathcal{I}} = \varDelta^{\mathcal{J}}$, we have that $r^{\mathcal{I}} = r^{\mathcal{J}}$ whenever $\mathcal{I}|_{\sigma'} = \mathcal{J}|_{\sigma'}$. We say that $r \in \sigma$ is *explicitly defined* by the relation symbols in σ' under $\varSigma$ if and only if there exists a first-order logic formula $\phi_r(\overline{x})$ over σ' with as many free variables as the arity of r and such that $\varSigma \models \big(\forall \overline{x} \,.\, r(\overline{x}) \equiv \phi_r(\overline{x})\big)$. The formula $\phi_r(\overline{x})$ is called an *explicit definition* (or simply *definition*) of r with respect to σ' under $\varSigma$.

Clearly, if a relation symbol $r \in \sigma$ has an explicit definition w.r.t. σ' under constraints $\varSigma$, then r is also implicitly defined by σ' under $\varSigma$, because fixing the interpretation of the relation symbols in σ' determines the interpretation of r. In other words, explicit definability always implies implicit definability. In general, the converse does not hold, that is, knowing that a certain symbol is implicitly defined by some other ones does not mean that we can (constructively) find an explicit definition of it in terms of those symbols. For first-order logic (FOL), a fundamental result by Beth [2] establishes that this is actually the case.

Theorem (Beth's Definability). *Let σ and σ' be disjoint signatures, and let $\varSigma$ be a finite set of constraints over $\sigma \cup \sigma'$. If $r \in \sigma$ is implicitly definable from σ' under $\varSigma$, then r has an explicit definition with respect to σ' under $\varSigma$.* $\square$

Note, however, that the above does not hold for all fragments of first-order logic, because even though a FOL explicit definition is always guaranteed to exist, this might not belong to the particular fragment under consideration. Fragments that are known to have the Beth's definability property are, for instance, the Guarded Fragment [6] and the Packed Fragment [9]. Other languages, even beyond first-order logic, are studied in [10]. In this paper, we consider constraints expressed in FOL, for which it is possible to effectively check for implicit definability and for which explicit definitions can be constructively obtained by means of rewriting techniques based on interpolation, e.g., as described in [3]. So, by reducing our problem to checking implicit definability and computing explicit definitions, we have found a constructive solution to it.

A *renaming* over a signature σ is a bijective function $\mathsf{ren}\colon \sigma \to \sigma'$, where σ' is a signature disjoint from σ. We extend $\mathsf{ren}(\cdot)$ to signatures, relational structures and (sets of) constraints in the natural way. For instance, given a constraint φ, $\mathsf{ren}(\varphi)$ is obtained by replacing in φ every occurrence of each relation symbol $r \in \mathsf{sig}(\varphi)$ with $\mathsf{ren}(r)$. Clearly, for a set Σ of constraints over σ and a relational structure $\mathcal{I}$ over $\mathsf{ren}(\sigma)$, we have that $\mathcal{I} \models \mathsf{ren}(\Sigma)$ iff $\mathsf{ren}^{-1}(\mathcal{I}) \models \Sigma$.

A *database schema* is a signature $\mathcal{R} = \{R_1, \dots, R_n\}$ of *database symbols* and a *view schema* is a signature $\mathcal{V} = \{V_1, \dots, V_k\}$ of *view symbols* not occurring in $\mathcal{R}$. A *database state* is a relational structure over $\mathcal{R}$ and a *view state* is a relational structure over $\mathcal{V}$. We denote the sets of all database and view states by S and T, respectively. For a database state $s \in S$ and a view state $t \in T$ having the same domain, the relational structure $s \uplus t$ is called a *global state* over $\mathcal{R} \cup \mathcal{V}$.

We consider a satisfiable finite set Σ of *global constraints* over the signature $\mathcal{R} \cup \mathcal{V}$. Since $\mathcal{R}$ and $\mathcal{V}$ are disjoint, Σ is partitioned into subsets $\Sigma_{\mathcal{R}}$, $\Sigma_{\mathcal{V}}$ and $\Sigma - (\Sigma_{\mathcal{R}} \cup \Sigma_{\mathcal{V}})$, which we call the *database constraints*, the *view constraints* and the *inter-schema constraints*, respectively. A database state s (resp., view state t) is *Σ-consistent* iff there exists a view state t (resp., database state s) with the same domain such that $s \uplus t$ is a model of Σ. We denote the set of Σ-consistent database states (resp., view states) by S_Σ (resp., T_Σ). When a specific Σ is not mentioned explicitly, we refer to Σ-consistent states as *globally consistent* states or states that are *consistent with the global constraints*.

The central notion used throughout the present article is that of *view under constraints*, defined as a function associating each globally consistent database state with a globally consistent view state with the same domain and such that the global state resulting from their union is a model of the constraints.

Definition 1 (View under constraints). A *view* from $\mathcal{R}$ to $\mathcal{V}$ under Σ is a total mapping $f\colon S_\Sigma \to T_\Sigma$ such that $s \uplus f(s) \models \Sigma$ for every $s \in S_\Sigma$. ■

The above naturally extends with explicit constraints the definition of view used by Bancilhon and Spyratos. Indeed, when only database constraints are present, the two notions essentially coincide, although in [1] constraints (at the database level) are considered only implicitly.

We write $\mathcal{R} \twoheadrightarrow_\Sigma \mathcal{V}$ to indicate that every $V \in \mathcal{V}$ is implicitly definable from $\mathcal{R}$ under constraints Σ. In addition, we also use $\mathcal{V} \twoheadrightarrow_\Sigma \mathcal{R}$, $\mathcal{R} \not\twoheadrightarrow_\Sigma \mathcal{V}$ and $\mathcal{V} \not\twoheadrightarrow_\Sigma \mathcal{R}$ with the obvious meaning. Since $\mathcal{R}$ and $\mathcal{V}$ are disjoint, every model of Σ has the form $s \uplus t$, where $s \in S$ and $t \in T$ are (globally consistent) states with the same domain. Therefore, we can give the following characterisation of definability in terms of states.

Definition 2. We say that $\mathcal{R}$ *defines* $\mathcal{V}$ under Σ (written $\mathcal{R} \twoheadrightarrow_{\Sigma} \mathcal{V}$) if and only if, for every $s \in S$ and $t, t' \in T$, whenever $s \uplus t \models \Sigma$ and $s \uplus t' \models \Sigma$, it is the case that $t = t'$. ■

There is an important connection between definability and views under constraints, consisting in the fact that the database symbols can be defined in terms of the view symbols under certain constraints iff a view mapping satisfying such constraints is unique.

Theorem 1. *$\mathcal{R} \twoheadrightarrow_{\Sigma} \mathcal{V}$ iff there is one and only one view from $\mathcal{R}$ to $\mathcal{V}$ under Σ.*

Proof. We will show that there exist two distinct views from $\mathcal{R}$ to $\mathcal{V}$ under Σ if and only if $\mathcal{R} \not\twoheadrightarrow_{\Sigma} \mathcal{V}$.

"if" Assume $\mathcal{R} \not\twoheadrightarrow_{\Sigma} \mathcal{V}$, then for some $s \in S_{\Sigma}$ there are $t', t'' \in T_{\Sigma}$ with $t' \neq t''$ such that $s \uplus t'$ and $s \uplus t''$ are models of Σ. Therefore, we can construct two views f' and f'' such that $f'(s) = t'$ and $f''(s) = t''$.

"only if" Let f and f' be such that $f(s) \neq f'(s)$ for some $s \in S_{\Sigma}$. Then, both $s \uplus f(s)$ and $s \uplus f'(s)$ are models of Σ, hence $\mathcal{R} \not\twoheadrightarrow_{\Sigma} \mathcal{V}$. □

The above theorem gives a characterisation of the views that are expressible by means of constraints in FOL. In what follows, we write $\mathcal{R} \twoheadrightarrow^{f}_{\Sigma} \mathcal{V}$ to indicate that $\mathcal{R} \twoheadrightarrow_{\Sigma} \mathcal{V}$ and f is the (one and only) view *induced by the constraints* Σ from $\mathcal{R}$ to $\mathcal{V}$. Every function f can be made surjective by restricting its codomain to its image: we call the resulting function the *surjection induced by* f or the *surjective restriction of* f. We use concatenation to indicate composition, e.g., fg denotes the composition of f with g.

3 The View Update Problem

In this section, we formally state the problem of view update, by reviewing and adapting some of the "classical" definitions given in [1] to our logic-based setting with constraints.

A *database update* is a function $d \colon S \to S$ that associates each database state with another, possibly the same. Similarly, a *view update* is a function $u \colon T \to T$ associating each view state with another, possibly the same. We denote by $U_{\mathcal{R}}$ and $U_{\mathcal{V}}$ the sets of all database and view updates, respectively. An update $u \in U_{\mathcal{V}}$ is called *strict on* f iff there exists $t \in f(S_{\Sigma})$ such that $u(t) \neq t$. In other words, a strict update does not coincide with the identity mapping on the image of f. The set of all the updates that are strict on f is denoted by U_f.

Given a view under constraints and a view update, we want to find a suitable database update that propagates the changes to the base relations in a consistent way. More specifically, the view update should be translated as a database update that brings the database into a new state from which, by applying the view definition, we reach exactly the updated view state. In addition, we also want to avoid unjustified and unnecessary changes in the database, in the sense that if the view update does not modify the view state, then the database update must not modify the corresponding database state. These requirements are formalised below (cf. Definition 3.1 in [1]).

Definition 3 (Translation). Let f be a view from $\mathcal{R}$ to $\mathcal{V}$ under Σ, let $d \in U_{\mathcal{R}}$ and $u \in U_{\mathcal{V}}$. We say that d is a *translation* of u (w.r.t. f) iff

(1) $uf = fd$; and (*consistent*)

(2) $\forall s \in S_\Sigma,\ uf(s) = f(s) \implies d(s) = s\,.$ (*acceptable*) ■

A translation of a given update on a view f can only exist if the updated view state lies in the image of f; otherwise, there would be no chance of reaching the new view state through f from some database state, which is what Definition 3 indeed requires. Hence, before we start looking for a translation, we should first make sure that the given view update allows for one.

Definition 4 (Translatability). Let $f\colon S_\Sigma \to T_\Sigma$ be a view from $\mathcal{R}$ to $\mathcal{V}$ under Σ. A view update $u \in U_{\mathcal{V}}$ is *translatable* (w.r.t. f) if and only if for each $s \in S_\Sigma$ there exists $s' \in S_\Sigma$ such that $f(s') = uf(s)$. ■

Note that the condition of translatability given in Definition 4 is equivalent to saying that u is translatable iff $u\big(f(S_\Sigma)\big) \subseteq f(S_\Sigma)$.

Translatability of view updates ensures that there is a translation, but does not rule out the possibility that more than one might exist, which is problematic because we would not know how to choose one. Therefore, we are only interested in view updates that are *uniquely translatable*, that is, for which there exists one and only one translation.

One of the factors contributing to the existence of multiple translations is the loss of information that occurs when two distinct database states collapse into the same view state, that is, when the view mapping is not injective. Intuitively, if a view update results in a view state that is the image of two different database states, we then have two different alternatives for “going back” to the database, and therefore multiple ways of translating the update. However, there might be uniquely translatable updates also on views that are not injective. In fact, even in the presence of view states corresponding to more than one database state, there is only one possible translation (if one does exist at all) as long as the view update does not result in one of these “ambiguous” view states.[1]

When the view mapping is injective, a view update is translatable if and only if it is uniquely translatable and the following theorem gives a characterisation of its unique translation.

Theorem 2. *Let f be an injective view under Σ, let $u \in U_{\mathcal{V}}$ be translatable and let $d \in U_{\mathcal{R}}$. Let $\hat{f}$ denote the surjection induced by f and let $\hat{u}$ be obtained from u by restricting its domain and codomain to $f(S_\Sigma)$.*[2] *Then, d is a translation of u if and only if $d = \hat{f}^{-1}\hat{u}\hat{f}$.*

Proof. Special case of Theorem 5.5 and Theorem 5.6 in [1] when $g \equiv 0$. □

[1] We have a theorem stating this more precisely, but unfortunately we are forced to omit it from the paper for space reasons.

In order to be able to apply the result of Theorem 2, we need to know, in the first place, whether a view is injective. More importantly, once in the presence of an injective view, we also need some way of computing the inverse of its surjective restriction, so as to effectually obtain the unique translation of any translatable view update. Therefore, below we study the injectivity and surjectivity of views under constraints using logical definability, with the aim of giving a constructive characterisation of their inverse.

Lemma 1. *Let f be a view from $\mathcal{R}$ to $\mathcal{V}$ under Σ and let $\mathcal{V} \twoheadrightarrow^{h}_{\Sigma} \mathcal{R}$. Then, 1) f is injective; 2) the restriction of h to the image of f is the inverse of the surjection induced by f.*

Proof.

1) Suppose f is not injective, that is, there are states $s, s' \in S_\Sigma$ such that $s \neq s'$ and $f(s) = f(s')$. Since f is a view under Σ, $s \uplus f(s)$ and $s' \uplus f(s')$ are models of Σ, in contradiction of $\mathcal{V} \twoheadrightarrow_{\Sigma} \mathcal{R}$.
2) We show that $hf(s) = s$ for every $s \in S_\Sigma$. Let $s \in S_\Sigma$ and $t = f(s)$. Since f is a view under Σ, $s \uplus t \models \Sigma$. Now, let $s' = h(t)$ and suppose $s' \neq s$. As h is a view under Σ, also $s' \uplus t \models \Sigma$, in contradiction of $\mathcal{V} \twoheadrightarrow_{\Sigma} \mathcal{R}$. $\square$

Note that in the above lemma, f can be any view under constraints. In the case of a view induced by the constraints, we have the following:

Lemma 2. *Let $\mathcal{R} \twoheadrightarrow_{\Sigma} \mathcal{V}$ and let f be the view from $\mathcal{R}$ to $\mathcal{V}$ induced by Σ. Then, 1) f is surjective; 2) f is injective if and only if $\mathcal{V} \twoheadrightarrow_{\Sigma} \mathcal{R}$.*

Proof.

1) Suppose f is not surjective, that is, there exist states $s \in S_\Sigma$ and $t \in T_\Sigma$ such that $s \uplus t$ is a model of Σ but $t \neq f(s)$. Since f is a view under Σ, also $s \uplus f(s) \models \Sigma$, in contradiction of $\mathcal{R} \twoheadrightarrow_{\Sigma} \mathcal{V}$.
2) By Lemma 1, f is injective whenever $\mathcal{V} \twoheadrightarrow_{\Sigma} \mathcal{R}$, thus we just need to show the "only if" part. The proof is by contraposition. Assume $\mathcal{V} \not\twoheadrightarrow_{\Sigma} \mathcal{R}$, hence there are Σ-models $s \uplus t$ and $s' \uplus t$ with $s \neq s'$. Since $s, s' \in S_\Sigma$ and f is a view under Σ, $s \uplus f(s)$ and $s' \uplus f(s')$ are also models of Σ. But then, as $\mathcal{R} \twoheadrightarrow_{\Sigma} \mathcal{V}$, we have $f(s) = t = f(s')$. Therefore, f is not injective. $\square$

Assuming the view to be induced by the constraints is a restriction almost always satisfied in practise. In fact, a view is usually specified by providing an explicit definition for each view symbol, in terms of the database symbols. For instance, each (virtual) view table in SQL is defined by means of a named `SELECT`-query over the database tables.

The following is an important consequence of Lemma 1 and Lemma 2, stating that it is possible to invert a view induced by a set of constraints iff the database symbols are implicitly defined by the view symbols under the same constraints, in which case the inverse is also effectively computable. In such a situation, the constraints induce two views that are indeed one the inverse of the other.

Theorem 3. *Let $\mathcal{R} \twoheadrightarrow^{f}_{\Sigma} \mathcal{V}$. Then, f is invertible if and only if $\mathcal{V} \twoheadrightarrow^{h}_{\Sigma} \mathcal{R}$, and in such a case $h = f^{-1}$.* $\square$

[2] As u is assumed to be translatable, $u\big(f(S_\Sigma)\big) \subseteq f(S_\Sigma)$.

The next step towards the application of Theorem 2 is the translatability of view updates, of which we give an interesting characterisation in what follows. The general idea consists in imposing additional constraints on the view schema so that every *legal* view update is translatable.

A consistent set of constraints over $\mathcal{R} \cup \mathcal{V}$ is $\mathcal{R}$*-defining* iff it contains only formulas, one for each $R \in \mathcal{R}$, of the form $\forall \overline{x}\big(R(\overline{x}) \equiv \phi_R(\overline{x})\big)$, with $\mathsf{sig}\big(\phi_R(\overline{x})\big) \subseteq \mathcal{V}$. Clearly, an $\mathcal{R}$-defining set Θ is such that $\mathcal{V} \twoheadrightarrow_{\Theta} \mathcal{R}$ and induces a function θ from $\mathcal{V}$ to $\mathcal{R}$. Since Θ does not contain nor entail any database or view constraints, every view state $t \in T$ is Θ-consistent and therefore in the domain of θ. We know by Beth's theorem that whenever $\mathcal{V} \twoheadrightarrow_{\Sigma} \mathcal{R}$ there is an explicit definition for each of the database symbols in terms of the view symbols, that is, the constraints entail an $\mathcal{R}$-defining set Θ. In such a case, we call the $\mathcal{V}$*-embedding* of Σ the set $\widetilde{\Sigma}_{\mathcal{V}}$ of view constraints obtained by replacing, for each $R \in \mathcal{R}$, every occurrence of $R(\overline{x})$ in Σ with the definition $\phi_R(\overline{x})$ given in Θ. The $\mathcal{V}$-embedding of a set Σ of global constraints is a set of view constraints having the same "restrictiveness" of the whole Σ, but with the advantage that they can be checked locally on the view schema. Indeed, it turns out that a view state is Σ-consistent iff it is a model of $\widetilde{\Sigma}_{\mathcal{V}}$ and this is of particular importance in the case of surjective views.

THEOREM 4. *Let f be a surjective view from $\mathcal{R}$ to $\mathcal{V}$ under Σ, let $\mathcal{V} \twoheadrightarrow_{\Sigma} \mathcal{R}$ and let $u \in U_{\mathcal{V}}$. Then, u is translatable if and only if $u(t) \models \widetilde{\Sigma}_{\mathcal{V}}$ for every $t \in T_{\Sigma}$.*

Proof (sketch). Since $\mathcal{V} \twoheadrightarrow_{\Sigma} \mathcal{R}$, let Θ be the $\mathcal{R}$-defining set entailed by Σ and let θ denote the function induced by Θ. Then, we have that $\theta(t) \uplus t \models \Theta$ for every $t \in T$. Hence, for every $\psi \in \Sigma$, we have that $\theta(t) \uplus t \models \forall \overline{x}\big(\psi(\overline{x}) \equiv \psi'(\overline{x})\big)$, where ψ' is obtained by replacing every occurrence in ψ of each database predicate with the corresponding definition given in Θ. As $\widetilde{\Sigma}_{\mathcal{V}}$ is obtained through the same substitution, we get that $\theta(t) \uplus t \models \Sigma$ iff $t \models \widetilde{\Sigma}_{\mathcal{V}}$, that is, $t \in T$ is Σ-consistent iff it is a model of the $\mathcal{V}$-embedding of Σ. Since f is surjective, $f(S_{\Sigma}) = T_{\Sigma}$. Therefore, u is translatable iff $u(T_{\Sigma}) \subseteq T_{\Sigma}$, that is, iff $u(t) \in T_{\Sigma}$ for every $t \in T_{\Sigma}$, and $u(t) \in T_{\Sigma}$ iff $u(t) \models \widetilde{\Sigma}_{\mathcal{V}}$. $\square$

Note that, under the assumptions of Theorem 4, every globally consistent view state is in the image of the view and, moreover, satisfies the $\mathcal{V}$-embedding of the global constraints. Thus, the above result essentially says that we have to make sure that, by updating a view state that is legal w.r.t. the embedded constraints, we always end up in another legal view state.

Let ren be a renaming over $\mathcal{R} \cup \mathcal{V}$ and let $\mathsf{ren}(\mathcal{V}) = \mathcal{V}'$. Then, a $\mathcal{V}'$-defining set Ξ of constraints over $\mathcal{V} \cup \mathcal{V}'$ represents a view update. Indeed, the function ξ induced by Ξ takes a view state t over $\mathcal{V}$ and returns an updated view state $\xi(t)$ over $\mathcal{V}'$. The view update *expressed* by Ξ is the function associating each $t \in T$ with $\mathsf{ren}^{-1}\big(\xi(t)\big)$. From Theorem 4, we then get the following characterisation of the translatability of those view updates that are expressible, as described, by means of a logical theory.

THEOREM 5. *Let f be a surjective view from $\mathcal{R}$ to $\mathcal{V}$ under Σ, let $\mathcal{V} \twoheadrightarrow_{\Sigma} \mathcal{R}$ and let $u \in U_{\mathcal{V}}$ be expressed by Ξ. Then, u is translatable iff $\widetilde{\Sigma}_{\mathcal{V}} \cup \Xi \models \mathsf{ren}\big(\widetilde{\Sigma}_{\mathcal{V}}\big)$.*

Proof (sketch). Let ξ denote the function induced by Ξ. We have that $t \uplus \xi(t) \models \widetilde{\Sigma}_{\mathcal{V}} \cup \Xi$ for every $t \in T_\Sigma$ and that, vice versa, every model $\mathcal{I}$ of $\widetilde{\Sigma}_{\mathcal{V}} \cup \Xi$ is such that $\mathcal{I} = t \uplus \xi(t)$ for some $t \in T_\Sigma$. Then, $\widetilde{\Sigma}_{\mathcal{V}} \cup \Xi \models \mathsf{ren}(\widetilde{\Sigma}_{\mathcal{V}})$ iff $\mathcal{I} \models \widetilde{\Sigma}_{\mathcal{V}} \cup \Xi$ implies $\mathcal{I} \models \mathsf{ren}(\widetilde{\Sigma}_{\mathcal{V}})$ for every $\mathcal{I}$, which in turn is the case iff $t \uplus \xi(t) \models \mathsf{ren}(\widetilde{\Sigma}_{\mathcal{V}})$ for every $t \in T_\Sigma$. Moreover, we have that $t \uplus \xi(t) \models \mathsf{ren}(\widetilde{\Sigma}_{\mathcal{V}})$ iff $\xi(t) \models \mathsf{ren}(\widetilde{\Sigma}_{\mathcal{V}})$ iff $\mathsf{ren}^{-1}\big(\xi(t)\big) \models \widetilde{\Sigma}_{\mathcal{V}}$ iff $u(t) \models \widetilde{\Sigma}_{\mathcal{V}}$. Therefore, since (by assumption) f is surjective and $\mathcal{V} \twoheadrightarrow_\Sigma \mathcal{R}$, our claim follows from Theorem 4. □

Under the assumptions of the above theorem, the view f is injective by Lemma 1. Hence, by Theorem 2 every translatable view update u has the unique translation $f^{-1}uf$. However, we might not be able to actually compute f^{-1} unless $\mathcal{R} \twoheadrightarrow_\Sigma \mathcal{V}$, in which case Theorem 3 ensures that the inverse of f is the view from $\mathcal{V}$ to $\mathcal{R}$ induced by Σ. When $\mathcal{R} \twoheadrightarrow_\Sigma \mathcal{V}$, $\mathcal{V} \twoheadrightarrow_\Sigma \mathcal{R}$ and Ξ expresses a translatable view update u, we have that $\mathcal{V} \twoheadrightarrow_\Xi \mathsf{ren}(\mathcal{V})$ and $\mathsf{ren}(\mathcal{V}) \twoheadrightarrow_{\mathsf{ren}(\Sigma)} \mathsf{ren}(\mathcal{R})$, therefore the unique translation of u is the database update expressed by the set Υ such that $\mathcal{R} \twoheadrightarrow_\Upsilon \mathsf{ren}(\mathcal{R})$, obtained by replacing in $\mathsf{ren}(\Sigma)$ every occurrence of $\mathsf{ren}(V)$ with its definition in terms of $\mathcal{V}$ and, in turn, every occurrence of V with its definition in terms of $\mathcal{R}$.

4 The View Complement

In the previous section, we presented a scenario in which injective views ensure that for each translatable view update there is only one possible and acceptable way of consistently propagating the changes towards the base relations through a suitable database update. Since an injective view is lossless, in such a case the update is essentially performed on a "restructured" copy of the whole database, in the sense that the full informative content of the database is available also in the view, though by means of a different schema. It is easy to imagine situations in which this is not case, and sometimes even undesirable. For instance, consider a (most common) scenario where user views are created in order to allow access to specific portions of the database, keeping untouched the rest of the data that is beyond the scope of the view. Since such views are lossy by design, the results achieved in Sec. 3 are not directly applicable.

The lack of injectivity in a view causes loss of information due to the fact that distinct database states are mapped to the same view state. In order to be able to distinguish between distinct database states, we need some extra "hints" that, combined with what is already known from the view itself, give a full account of the database content. This additional information is provided by another view, that takes the name of *view complement*, because it "complements" the partial information of a lossy view.

For the rest of this section, let $\mathcal{R}$, $\mathcal{V}$ and $\mathcal{W}$ be pairwise disjoint signatures, and let Σ and Γ be finite sets of constraints over $\mathcal{R} \cup \mathcal{V}$ and $\mathcal{R} \cup \mathcal{W}$, respectively, such that their union is consistent.

Definition 5 (View complement). Let f be a view from $\mathcal{R}$ to $\mathcal{V}$ under Σ and let g be a view from $\mathcal{R}$ to $\mathcal{W}$ under Γ. We say that g is a *complement* of f iff (1) $S_\Sigma = S_\Gamma$ and (2) $\forall s, s' \in S_\Sigma,\ s \neq s' \wedge f(s) = f(s') \implies g(s) \neq g(s')$. ■

In other words, a complement of f is a view g operating on the same domain of f and capable of distinguishing between distinct database states which f maps to the same view state. Note that there exists at least one complement for every view, namely the "identity" mapping over the whole database.

The idea of view complement was first introduced by Bancilhon and Spyratos in [1]. Our definition is indeed based on their work (cf. Theorem 4.2 in [1]) with the additional requirement that f and g must have the same domain, which has to be explicitly enforced here as we are in a setting with views under constraints. Since the notion of view complement is symmetric, in that g is a complement of f iff f is a complement of g, we will sometimes simply say that two views f and g are "complementary".

Given two views f and g under constraints Σ and Γ, respectively, their *union* is the function $f \uplus g$ associating each $s \in S_\Sigma \cap S_\Gamma$ with the state $f(s) \uplus g(s)$. The union of f and g turns out to be a view under $\Sigma \cup \Gamma$ and, when f and g are induced by their associated constraints, $f \uplus g$ is indeed induced by $\Sigma \cup \Gamma$. The connection between complementarity and injectivity of views is given by the fact that two views under constraints and with the same domain are complementary if and only if their union is injective.

LEMMA 3. *Let f be a view from $\mathcal{R}$ to $\mathcal{V}$ under Σ, let g be a view from $\mathcal{R}$ to $\mathcal{W}$ under Γ and let $S_\Sigma = S_\Gamma$. Then, $f \uplus g$ is injective iff f and g are complementary.*

Proof. Assuming $S_\Sigma = S_\Gamma$, f and g are not complementary iff there exist $s, s' \in S_\Sigma$ with $s \neq s'$ and such that $f(s) = f(s')$ and $g(s) = g(s')$. As f and g are views under constraints, $f(s)$ and $g(s)$ are states with the same domain over disjoint signatures, and so are $f(s')$ and $g(s')$. Therefore, we have that $f(s) = f(s')$ and $g(s) = g(s')$ iff $f(s) \uplus g(s) = f(s') \uplus g(s')$, that is, iff $f \uplus g$ is not injective. $\square$

Below we give a characterisation of complementarity between two views induced by constraints in terms of logical equivalence and definability.

THEOREM 6. *Let $\mathcal{R} \twoheadrightarrow^{f}_{\Sigma} \mathcal{V}$ and $\mathcal{R} \twoheadrightarrow^{g}_{\Gamma} \mathcal{W}$. Then, f and g are complementary if and only if $\widetilde{\Sigma}_\mathcal{R} \equiv \widetilde{\Gamma}_\mathcal{R}$ and $\mathcal{V} \cup \mathcal{W} \twoheadrightarrow_{\Sigma \cup \Gamma} \mathcal{R}$.*[3]

Proof (sketch). Since $s \in S$ is Σ-consistent iff $s \models \widetilde{\Sigma}_\mathcal{R}$ and it is Γ-consistent iff $s \models \widetilde{\Gamma}_\mathcal{R}$, we have $S_\Sigma = S_\Gamma$ iff $\widetilde{\Sigma}_\mathcal{R} \equiv \widetilde{\Gamma}_\mathcal{R}$. Moreover, it can be easily shown that $\mathcal{R} \twoheadrightarrow^{f}_{\Sigma} \mathcal{V}$ and $\mathcal{R} \twoheadrightarrow^{g}_{\Gamma} \mathcal{W}$ together imply $\mathcal{R} \twoheadrightarrow^{f \uplus g}_{\Sigma \cup \Gamma} \mathcal{V} \cup \mathcal{W}$. Then, from Lemma 3, f and g are complementary iff $f \uplus g$ is injective, which by Lemma 2 is the case if and only if $\mathcal{V} \cup \mathcal{W} \twoheadrightarrow_{\Sigma \cup \Gamma} \mathcal{R}$. $\square$

Therefore, one way of finding a complement of a given view f induced by a set of constraints Σ consists in abducing another set of constraints Γ consistent with Σ and satisfying the conditions of Theorem 6, which guarantees that the view g induced by such a Γ is indeed a complement of f.

Following the rationale that the only purpose for which a view complement is made available is that of allowing for a lossy view to be updatable, we demand that the information it provides be invariant during the update process. In other

[3] $\widetilde{\Sigma}_\mathcal{R}$ and $\widetilde{\Gamma}_\mathcal{R}$ denote the $\mathcal{R}$-embeddings of Σ and Γ, respectively.

words, view updates must never modify, neither directly nor indirectly, any data that belongs to the view complement. Putting together translatability of updates and invariance of the complement results in the formal notion given below (cf. Definition 5.1 in [1]).

Definition 6 (g-translatability). Let f be a view under Σ and let g be a complement of f. A view update $u \in U_{\mathcal{V}}$ is called *g-translatable* iff for each $s \in S_{\Sigma}$ there exists $s' \in S_{\Sigma}$ such that:

(1) $f(s') = uf(s)$; and (translatability)
(2) $g(s') = g(s)$. (constant complement) ■

That is, a view update is g-translatable if it is translatable (according to Definition 4) and, in addition, leaves the complement g unchanged. For this reason, we say that such an update is *translatable under constant complement.* In general, there might be more than one complement of a given view, and an update is g-translatable or not depending on the particular complement g we consider. Thus, the choice of a complement defines an "update policy" by assigning unambiguous semantics to the view updates.

The following theorem establishes an important relationship between translatability w.r.t. a view under constant complement and translatability w.r.t. the union of a view and its complement.

Theorem 7. *Let f and g be complementary, let $u \in U_{\mathcal{V}}$ and let $v \in U_{\mathcal{W}} \setminus U_g$. Then, u is g-translatable w.r.t. f if and only if $u \uplus v$ is translatable w.r.t. $f \uplus g$.*

Proof. The update $u \uplus v$ is translatable w.r.t. $f \uplus g$ iff, for every $s \in S_{\Sigma}$, there exists $s' \in S_{\Sigma}$ such that $\big(f \uplus g\big)(s') = \big(u \uplus v\big)\big((f \uplus g)(s)\big)$. Since f and g have disjoint codomains, this is the case iff $f(s') \uplus g(s') = uf(s) \uplus vg(s)$ and, in turn, iff $f(s') = uf(s)$ and $g(s') = vg(s)$. As v is a non-strict update, it is the identity on $g(S_{\Sigma})$, therefore $vg(s) = g(s)$. □

This allows us to extend the result obtained for the translatability of updates on injective views to the case of g-translatability.

Theorem 8. *Let $\mathcal{R} \twoheadrightarrow^{f}_{\Sigma} \mathcal{V}$, let $\mathcal{R} \twoheadrightarrow^{g}_{\Gamma} \mathcal{W}$ and let g be a complement of f. Let $\widetilde{\Pi}$ be the $(\mathcal{V} \cup \mathcal{W})$-embedding of $\Sigma \cup \Gamma$ and let* ren *be a renaming over $\mathcal{R} \cup \mathcal{V} \cup \mathcal{W}$. Let $u \in U_{\mathcal{V}}$ be expressed by Ξ and let Ω be the $\mathcal{W}$-defining set such that $\forall \overline{x} \,.\, W(\overline{x}) \equiv \mathsf{ren}\big(W(\overline{x})\big)$ for each $W \in \mathcal{W}$. Then, u is g-translatable if and only if $\widetilde{\Pi} \cup \Xi \cup \Omega \models \mathsf{ren}(\widetilde{\Pi})$.*

Proof. Let ω denote the function induced by Ω. Clearly, $z = \mathsf{ren}^{-1}\big(\omega(z)\big)$ for every $z \in g(S_{\Sigma})$, therefore Ω expresses an update $v \in U_{\mathcal{W}} \setminus U_g$. By Theorem 7, u is g-translatable w.r.t. f iff the update $u \uplus v$ expressed by $\Xi \cup \Omega$ is translatable w.r.t. $f \uplus g$ and, by Theorem 5, this is the case iff $\widetilde{\Pi} \cup \Xi \cup \Omega \models \mathsf{ren}\big(\widetilde{\Pi}\big)$. □

5 Conclusion and Future Work

We presented a framework for view update based on the notion of "view under constraints". On the one hand, such a framework "extends"—so to say—the one of Bancilhon and Spyratos by adding explicit constraints also at the view level.

Indeed, when there are no inter-schema constraints nor constraints on the view schema, the notion of view under constraints coincides with the notion of view used in [1]. On the other hand, our framework is an instance of Bancilhon and Spyratos' abstract one, in that we essentially consider only view mappings that are expressible by means of first-order logic constraints.

Using logical definability, we gave a constructive characterisation of when and whether a view induced by a set of constraints is invertible, so as to being able to effectively compute the (unique) translation of a view update that is translatable and expressible in FOL. Indeed, we also provided an applicable method, based on the Beth's definability property and the idea of local "embedding" of the constraints, for testing whether a FO-expressible view update is translatable (under constant complement). We have an experimental tool, based on a FOL theorem prover, that checks for implicit definability and derives explicit definitions, and so it can be used for testing the criterion of translatability we presented in this paper and for computing the corresponding translation.

For what concerns future work, first we would like to formally show that the setting considered in [4] by Cosmadakis and Papadimitriou is a special case of our general framework and that their results on the translatability of insertions, deletions and replacements can be derived with an instantiation of the method we presented. Then, we want to study in some detail the connection with logical abduction, that could be used for obtaining view complements and translatable updates.

References

1. F. Bancilhon and N. Spyratos. Update semantics of relational views. *ACM Transactions on Database Systems*, 6(4):557–575, December 1981.
2. E. W. Beth. On Padoa's method in the theory of definition. *Indagationes Mathematicae*, 15:330–339, 1953.
3. A. Borgida, J. de Bruijn, E. Franconi, I. Seylan, U. Straccia, D. Toman, and G. Weddell. On finding query rewritings under expressive constraints. In *Proceedings of SEBD-2010*, Rimini, Italy, June 2010.
4. S. S. Cosmadakis and C. H. Papadimitriou. Updates of relational views. *Journal of the Association for Computing Machinery*, 31(4):742–760, October 1984.
5. G. Gottlob, P. Paolini, and R. Zicari. Properties and update semantics of consistent views. *ACM Transactions on Database Systems*, 13(4):486–524, December 1988.
6. E. Hoogland, M. Marx, and M. Otto. Beth definability for the guarded fragment. In *Logic for Programming and Automated Reasoning*, volume 1705 of *Lecture Notes in Computer Science*, pages 273–285. Springer Berlin / Heidelberg, 1999.
7. J. Lechtenbörger. The impact of the constant complement approach towards view updating. In *Proceedings of PODS 2003*, pages 49–55, San Diego, CA, June 2003.
8. J. Lechtenbörger. A note on "the impact of the constant complement approach towards view updating", October 2003.
9. M. Marx. Queries determined by views: Pack your views. In *Proceedings of PODS 2007*, pages 23–30, Beijing, China, June 2007.
10. A. Nash, L. Segoufin, and V. Vianu. View and queries: Determinacy and rewriting. *ACM Transactions on Database Systems*, 35(3), July 2010.