On Logically Justified Updates [1]

Michael Dekhtyar
Dept. of CS, Tver State Univ.
3 Zheljabova str. Tver, Russia, 170013
Michael.Dekhtyar@tversu.ru

Alexander Dikovsky
Keldysh Institute for Applied Math.
4 Miusskaya sq. Moscow, Russia, 125047
dikovsky@spp.keldysh.ru

Nicolas Spyratos
Université de Paris-Sud, LRI, U.R.A. 410 du CNRS, Bât. 490
F-91405 Orsay Cedex, France
Nicolas.Spyratos@lri.fr

Abstract The Enforced Update Problem we investigate in this paper is as follows: given a theory T which formalizes integrity constraints, an initial state $I \models T$, and an external update Δ which specifies the facts D^+ to be added to I and the facts D^- to be deleted from it, one should introduce minimal changes $\Psi(I)$ in I, sufficient to accomplish Δ and to restore T if and when it is violated, i.e. to guarantee that $D^+ \subseteq \Psi(I), \Psi(I) \cap D^- = \emptyset$, *and* $\Psi(I) \models T$. We give an axiomatic description of such logically justified updates Ψ and describe their reasonable implementation.

1 Introduction

To put it very generally, a database (DB) or a knowledge base (KB) can be seen as a theory T whose models represent states (worlds). The difference between the two is that KBs T are supposed to be deductively closed, whereas this is not necessary of DBs. However in Deductive Data Bases (DDBs), similar to KBs, the *extensional* and *intensional* parts of states are distinguished. The first makes the explicit part of a state, whereas the other is available through its closure. In conventional DBs nothing but the extensional part exists. The difference between KBs and DBs becomes clearer in presence of *actions* changing their states. Speaking of DBs one has in mind a system of conditions on states, *integrity constraints* (IC), which should be preserved after changes. Meanwhile, the IC are optional for KBs. In KBs the actions incorporate some new knowledge into theories representing states, this being accompanied by subsequent logically justified changes. As a result the theories evolve.

[1]This work was sponsored by the Russian Fundamental Studies Foundation (Grants 96-01-00395, 97-01-00973).

Two main approaches to modifying KB and DB states should be distinguished: *revisions* and *updates*. Loosely speaking the revision applies to the whole theory T in some constructive representation (formula, logic program, etc.) and to some new knowledge p and describes the resulting class of states (worlds) in terms of another theory T'. In other words the revision is theory oriented (cf. [23]). In contrast, the update applies simultaneously and independently to individual models of T and transforms them into some models of p. In other words it is pointwise model oriented.

The first effort to formalize the KB modifications was made by Gärdenfors and his colleagues [1, 11] who have proposed some philosophically founded set of postulates. Later Katsuno and Mendelzon [15] have refined these postulates in propositional case. They were first to point out the distinction between the notions of revision and update. Quite nonevidently the axioms of Katsuno and Mendelzon express a very important principle, that of a *minimal change* produced in initial models to obtain the resulting models. This principle was subsequently expressed by many authors through some explicit criterion of minimal difference between two models. The definitions as well as references can be found in the paper [9] where the computational complexity of revisions and updates is explored.

In 1st order case the definitions of DB and KB modifications are complicated by the absence of a satisfactory general constructive interpretation of negation. Various particular semantics were proposed since late seventies (see e.g. [6, 3, 13, 22]). Besides this, different theory interpretations exist: the classical *total* (2-valued) interpretations and the *partial* (3-valued) interpretations. Under total interpretations the truth of $\neg A$ in a model is *equivalent* to the absence of A in this model, whereas this is not the case in general in partial interpretations. Accordingly under total interpretations the models are sets of atoms, and under partial interpretations they are sets of literals. A particular type of interpretations $\mathcal{I}$ being fixed, an update Δ being applied to a KB or a DB specified by a logic program Φ, induces in general a nondeterministic update operator $\Psi_{\Phi,\Delta} : \mathcal{I} \mapsto 2^{\mathcal{I}}$. Sometimes Φ acts as an intensional program, in other cases as IC or even both. Rarely separate logic programs serve for specifying intensional knowledge and IC. There are different techniques of defining the resulting models $\Psi_{\Phi,\Delta}(I)$: sequential, which are based on some notion of a derivation or a computation, now frequently applied to active DBs [7], see e.g. [4, 5, 12, 17, 19, 21, 24], and various nonsequential, such as e.g. revision programming [16], inertia axioms [18, 2], and conflict resolution [14, 10, 8]. There is certain technical dissimilarity between the definitions of KB update operators $\Psi_{\Phi,\Delta}$ from those of DBs. The former generally use some default rule based semantics of negation (CWA [20], stable model semantics [13], etc.), whereas for the latter it is not natural: DB states are determined by actual data. The basic difference between the two is in the way they treat the new data (knowledge) Δ when it is not consistent with Φ. The result of a KB update $\Psi_{\Phi,\Delta}(I)$ in this case is a model of Δ whose deviation from the initial model I is "minimal" and justified by Φ (see [2]). In other words KBs adapt themselves to new knowledge. On the contrary, the result of a DB update $\Psi_{\Phi,\Delta}(I)$ is a model of Φ "minimally deviating" from the initial model I and incorporating a "maximal" part of Δ consistent with Φ. So DBs adapt new data to IC. In both cases the deviation of the resulting model from the initial one is justified by Φ.

This paper substantially extends our previous paper [8]. Here we present an axiomatic definition of DB update operators introducing into DB states minimal necessary changes (we call these operators *conservative*), whereas in [8] a particular algorithm implementing a conservative operator was described and verified. In [8] it is presumed that the external update does not contradict the IC. Here we consider the opposite case too. Besides this, in this parer both, the partial and the total interpretations are considered, whereas in [8] we consider only the partial ones.

This paper is organized as follows. Section 2 contains the basic notions. The axiomatic definition of the conservative update operators is given and discussed in section 3. In section 4 we describe quite a feasible implementation of conservative update operators in the class of partial interpretations, based on conflict resolution technique. In the last section we discuss their implementation in the class of total interpretations.

2 Basic notions and notation

We consider a first order signature $\mathbf{S}$ consisting of a set $\mathbf{C}$ of constants, a set $\mathbf{P}$ of predicates, and a set of variables $\mathbf{V}$. For a finite set of constants $C \subset \mathbf{C}$ in this signature $\mathbf{A}_C$ denotes the set of all atoms in $\mathbf{S}$ with variables in $\mathbf{V}$ and constants in C. $\mathbf{B}_C \subset \mathbf{A}_C$ denotes the set of all ground instances of atoms in $\mathbf{A}_C$ with respect to C. For an atom $a \in \mathbf{A}_C$ both a and $\neg a$ are *literals.* A pair of the form $(a, \neg a)$ or $(\neg a, a)$ is called a *contrary pair.* For a contrary pair (l_1, l_2) we write $l_1 = \neg . l_2$ and $l_2 = \neg . l_1$. The set of all literals over $\mathbf{V}$ and C is denoted by $\mathbf{L}_C$. For each $W \subseteq \mathbf{L}_C$ we denote by $\neg . W$ the set $\{a | \neg a \in W\} \cup \{\neg a | a \in W\}$. $\mathbf{LB}_C$ denotes the set $\mathbf{B}_C \cup \neg . \mathbf{B}_C$. Let Φ be a finite set of clauses of the form

$$r = (l \ \leftarrow \ l_1, ..., l_n)$$

where $n \geq 0$ and l, l_i are literals in $\mathbf{L}_C$. We call Φ *integrity constraints*, or simply *IC* and the clauses of Φ *constraints.* A constraint is *normal* if it has an atom in its head. Φ is *normal* if all its constraints are normal. We denote by $ground_C(\Phi)$ the set of all ground instances of constraints in Φ using constants in C. Finite subsets $I \subset \mathbf{LB}_C$ are called *partial interpretations.* For a partial interpretation I we set $I^+ = I \cap \mathbf{B}_C$ and $I^- = I \cap \neg . \mathbf{B}_C$. A partial interpretation I is *total* if $I^+ \cup \neg . I^- = \mathbf{B}_C$. So a total interpretation is completely defined by its positive part. Considering partial interpretations we shall regard them as *DB states.* In this context a DB state is *consistent* if it does not contain contrary pairs. Intuitively, in a consistent partial interpretation I the atoms in I^+ are regarded as true, the atoms in $\neg.I^-$ are regarded as false, and all others are regarded as unknown. Considering only total interpretations we shall call DB states their positive parts I^+. When it does not lead to a conflict, we name I^+ a total interpretation. A constraint $r = (l \leftarrow l_1, ..., l_n)$ is *valid in a DB state* I (denoted $I \models r$) if $I \models l$ whenever $I \models l_i$ for each $1 \leq i \leq n$. When partial interpretations are considered $I \models l$ means $l \in I$. When total interpretations are considered, then for a DB state I and for an atom a $I \models a$ means $a \in I$, and for a literal $l = \neg a$ $I \models l$ means $a \notin I$. Total interpretations are consistent by definition. IC Φ *are valid in a DB state* I if every constraint in Φ is valid in I. This is denoted by $I \models \Phi$. I is a *model* of Φ if it is consistent and $I \models \Phi$.

A pair $\Delta = (D^+, D^-)$ where D^+, D^- are finite subsets of $\mathbf{LB}_C$, and $D^+ \cap D^- = \emptyset$, is called an *update*. Intuitively, the literals of D^+ are to be added to I, and those of D^- are to be removed from I. In total interpretations both D^+ and D^- are sets of atoms. We say that $\Delta = (D^+, D^-)$ is *accomplished* in I if $D^+ \subseteq I$ and $D^- \cap I = \emptyset$.

An *implication tree* of a IC Φ for a literal l is a finite tree $T(l)$ whose nodes are labeled by ground literals with no contrary pair among them, the root is labeled by l, and a node v is labeled by a literal l_0 if and only if there is a constraint $r = (l_0 \leftarrow l_1, ..., l_n) \in \Phi$ such that if $n > 0$, then v has sons $v_1, \ldots, v_n$ labeled by $l_1, \ldots, l_n$ respectively. We denote by $cr(T)$ the *crown* of T, i.e. the set of all literals labelling the leaves of T. It is clear that the leaves are labelled by the literals l occurring in facts $l \leftarrow$ of Φ.

For some given Φ, Δ, and I let con be the set of all constants which occur in them. We consider below only the constants in con, i.e. only the sets $\mathbf{A}_{con}$, $\mathbf{B}_{con}$, $\mathbf{L}_{con}$, $\mathbf{LB}_{con}$, $ground_{con}(\Phi)$, etc., therefore the subscript con will be dropped in the sequel.

3 Conservative update operators enforcing IC

We consider the following problem which we call in [8] an *Enforced Update Problem* (*EUP*). Given a DB state I, an update $\Delta = (D^+, D^-)$, and an IC Φ one should find a consistent DB state I_1 such that

- $I_1 \models \Phi$,
- Δ is accomplished in I_1, and
- I_1 is as "*close*" to I as possible.

The first claim says that one needs to restore the IC Φ after the update. The second claim says that the update is enforced in the course of the transformation of the initial DB state. The third claim expresses the conservative nature of the change. It requires that only minimal necessary changes should be performed in the initial state in order to make the resulting DB state satisfy the first two claims.

The *EUP* was resolved in [14] for constraints of the form $l \leftarrow l_1$. The operators providing solutions to the general *EUP* can be naturally axiomatized. Before we proceed to their axiomatization we should introduce some additional notions and notation.

The claim of minimal changes included into the *EUP* should be formulated in terms of some criterion of "closeness" of DB states. We propose a new and very natural criterion, which is a combination of two independent criteria: that of maximal intersection with the initial DB state, and the other, of minimal symmetric difference with respect to this DB state. We proceed from the assumption that it is more important to keep as much initial facts as possible, than to add possibly fewer new facts. The criterion of minimal symmetric difference as a concept of closeness has been frequently used, see for example [15, 9, 16, 18, 14]. The complex criterion we use here is more precise. First it minimizes deletions of available data and then it minimizes insertions of new data. This is put in a precise form as follows.

Definition 1. *Let I, I_1, I_2 be three DB states. We say that I_1 is to I than I_2 (notation: $I_1 \geq^I_{it} I_2$) if $I \cap I_2 \subseteq I \cap I_1$. We write $I_1 \equiv^I_{it} I_2$ if $I_1 \geq^I_{it} I_2$ and $I_2 \geq^I_{it} I_1$, and we write $I_1 >^I_{it} I_2$ if $I_1 \geq^I_{it} I_2$ and $I_2 \not\geq^I_{it} I_1$. I_1 is to I than I_2 (notation: $I_1 \geq^I_{sd} I_2$) if $I \nabla I_1 \subseteq I \nabla I_2$. We write $I_1 >^I_{sd} I_2$ if $I_1 \geq^I_{sd} I_2$ and $I_2 \not\geq^I_{sd} I_1$.*

As we have already stressed, the DB states resulting from updates should satisfy the IC. Meanwhile, the update can contradict the IC in general. So we are to face two different situations. In the first situation the IC Φ and the update Δ are consistent. This property is easy to formalize.

Definition 2. *An update $\Delta = (D^+, D^-)$ is with IC Φ if there exists a consistent DB state I such that*

- *$I \models \Phi$ and*
- *Δ is accomplished in I.*

In our language the consistency can be checked constructively.

Definition 3. *Let Φ be IC. For a given partial interpretation I*

$$cl(I) = \{l | \exists r = (l \ \leftarrow \ l_1, ..., l_n) \in \Phi \ \ (\bigwedge_{i=1}^{n} I \models l_i \ )\}.$$

A is the total operator

$$T_\Phi(I) = \begin{cases} cl(I) & : \ cl(I) \text{ is consistent} \\ \mathbf{LB} & : \ cl(I) \text{ is inconsistent.} \end{cases}$$

Let us denote this operator by $T^{\in}_{\Phi}$ in the case of partial interpretations and by $T^{\models}_{\Phi}$ in the case of total interpretations.

Several useful remarks can be made concerning $T^{\in}_{\Phi}(I)$.

1. In contrast to the classical positive Horn logic programs, an interpretation I such that $cl(I) \subseteq I$, is not always a model of Φ because it can be inconsistent. In the latter case $T^{\in}_{\Phi}(I) = \mathbf{LB}$. Let us call the interpretations with the property $T^{\in}_{\Phi}(I) \subseteq I$ *closed sets* or *c-sets* of Φ.

2. Evidently, $T^{\in}_{\Phi}$ is continuous, i.e. for any nondecreasing sequence of partial interpretations $I_1 \subseteq I_2 \subseteq \ldots$ $T^{\in}_{\Phi}(\bigcup\limits_{i=1}^{\infty} I_i) = \bigcup\limits_{i=1}^{\infty} T^{\in}_{\Phi}(I_i)$. Therefore the least fixed point of $T^{\in}_{\Phi}$ is defined as $lfp(T^{\in}_{\Phi}) = T^{\in}_{\Phi}\uparrow\omega = M_{min}(\Phi)$, where $M_{min}(\Phi)$ is the minimal c-set of Φ. If $M_{min}(\Phi)$ is consistent, then it is the minimal model of Φ.

3. The definition of consistency of IC Φ with an update $\Delta = (D^+, D^-)$ guarantees the existence of a *model* I of Φ in which Δ is accomplished. Hence, for such Φ and Δ there evidently exists the *minimal model* $M_{min}(\Phi \cup D^+)$ which is in fact the set of all ground consequences of the update Δ with respect to Φ. We denote this model by M_{Δ}. $M_{\Delta} = lfp(T_{\Phi \cup D^+})$, therefore it is constructed in time linear with respect to the summary size of $ground(\Phi)$ and Δ. Hence the consistency of Φ and Δ is recognized in the same time.

In the situation where Δ is consistent with Φ it must be accomplished completely. In the case of their inconsistency one should define some part of Δ, consistent with Φ, which should be accomplished. First we present an axiomatic definition of the operators resolving the *EUP* in the case of consistent IC and update.

Definition 4. *(***Consistency case axioms AC** *) Let the given update Δ be consistent with IC Φ. In this case an operator $\Psi = \Psi[\Phi, \Delta]$ on the set of partial interpretations is a if for each I :*

(AC1) $\Psi(I)$ is a model of Φ,

(AC2) Δ is accomplished in $\Psi(I)$,

(AC3) $\Psi(I)$ is $\geq^{I}_{it}$-maximal with respect to the models of Φ in which Δ is accomplished,

(AC4) in the class of the models of Φ with accomplished Δ and $\equiv^{I}_{it}$-equivalent to $\Psi(I)$, this model is $\geq^{I}_{sd}$-maximal.

Our formalization of DB updates is not in the line of KB updates. The fundamental difference between the KB revisions and updates on the one hand and the DB updates on the other hand is that for the former the update Δ is taken for granted, and the theory Φ is changed when it contradicts Δ, whereas for the latter, just the opposite, the IC cannot be changed (point(AC1)) and the update should be reconciled to it in the case of contradictions. The idea behind this reconciliation is that a maximal in some sense part of Δ, consistent with Φ should be extracted and accomplished. Such a part can be defined in various ways. The simplest one is based on some priority relation on updates.

Let some linear order $\prec$ be given on pairs of sets of literals (D^+, D^-). For each $\Delta = (D^+, D^-)$ let us denote by $\Delta^{\prec}_{\Phi\Delta}$ the update $max_{\prec}\{(D^+_{\prec}, D^-_{\prec}) | D^+_{\prec} \subseteq D^+, D^-_{\prec} \subseteq D^-, \Phi$ *is consistent with* $(D^+_{\prec}, D^-_{\prec})\}$. In the general case where consistency of Δ and Φ is not presumed the axioms **AC** are revised as follows.

Definition 5. *(***General case axioms** $AG^{\prec}$ *) An operator* $\Psi = \Psi[\Phi, \Delta]$ *on the set of partial interpretations is a*

if for each I *:*

(AG1) $\Psi(I)$ *is a model of* Φ,

(AG2) $\Delta^{\prec}_{\Phi\Delta}$ *is accomplished in* $\Psi(I)$,

(AG3) $\Psi(I)$ *is* $\geq^I_{it}$*-maximal with respect to the models of* Φ *in which* $\Delta^{\prec}_{\Phi\Delta}$ *is accomplished,*

(AG4) in the class of the models of Φ *with accomplished* $\Delta^{\prec}_{\Phi\Delta}$ *and* $\equiv^I_{it}$*-equivalent to* $\Psi(I)$, *this model is* $\geq^I_{sd}$*-maximal.*

These axioms do not guarantee that the update with maximal priority leads to the model closest to the initial one. The axioms only say that the resulting model is closest among those defined by $\Delta^{\prec}_{\Phi\Delta}$. It is clear that the axioms AC are a special case of $AG^{\prec}$.

The definitions 4 and 5 are based on no default rules outlining some class of "good" models. This leads to the following simple but important model completeness property of enforced updates:

Theorem 1. *(of model completeness)*
Let I_1, I_2 *be two different models of a program* Φ. *Then there exists an update* Δ_0 *consistent with* Φ *such that* $I_2 = \Psi[\Phi, \Delta_0](I_1)$.

Proof. We just set $\Delta_0 = (I_2 \setminus I_1, I_1 \setminus I_2)$. This update is accomplished in I_2.
1. Δ_0 is consistent with Φ because $I_2 \models \Phi$.
2. $\Psi(I_1) \cap (I_1 \setminus I_2) = \emptyset$, we have $\Psi(I_1) \cap I_1 \subseteq I_1 \cap I_2$.
3. As I_2 is also a model of Φ with accomplished Δ_0, $\Psi(I_1) \geq^{I_1}_{it} I_2$, i.e. $I_1 \cap I_2 \subseteq \Psi(I_1) \cap I_1$, therefore $\Psi(I_1) \equiv^{I_1}_{it} I_2$.
4. Now, as Δ_0 is accomplished in $\Psi(I_1)$, this implies $I_2 \setminus I_1 \subseteq \Psi(I_1)$, and therefore $I_2 \subseteq \Psi(I_1)$.
5. On the other hand, 3 implies that $\Psi(I_1) \geq^{I_1}_{sd} I_2$. Hence, $\Psi(I_1) \setminus I_1 \subseteq I_2 \setminus I_1$, and so finally, $\Psi(I_1) \subseteq I_2$. $\Box$

This fact implies in particular that the range of a conservative update operator Ψ related to Φ coincides with the set of all models of Φ. This is very natural for the data bases, because they do not propose any criteria for the choice between different models of a user view. Meanwhile, this is just not true for the default rule based operators like the revision in [16, 18], view update in [10], etc. which are more natural for knowledge bases.

Since we consider the *EUP* for finite IC Φ, updates Δ, and initial DB states I, the space of DB states which result from updates is finite as well. So there evidently exists a brute force algorithm which implements some conservative IC enforcing update operator. Given some Φ, Δ, and I it enumerates all DB states in order induced by our relation of closeness to I and checks for every DB state I' whether $I' \models \Phi$ and Δ is accomplished in I'. The first DB state which satisfies both conditions is the result of the update.

4 Principle-oriented algorithms

Conservative IC enforcing update operators can be quite reasonably implemented. We present below an implementing algorithm $\mathcal{A}$ based upon "oracles" who judge which choice is right when a conflict is encountered. We call such an oracle a preference strategy.

Definition 6. *A is any function ρ which maps any contrary pair $(l, \neg.l)$ into one of the literals $l, \neg.l$. A strategy ρ and an update Δ are if for all $l \in M_\Delta$, $\rho(l, \neg.l) = l$ and for all $l \in D^-$, $\rho(l, \neg.l) = \neg.l$. A DB state I and a strategy ρ are if for all $l \in I$, $\rho(l, \neg.l) = l$.*

Another idea behind the algorithm $\mathcal{A}$ is that of a hitting set which cuts out all the choices contradicting a given preference strategy and so reconciles globally all the conflicts caused by the update to be accomplished.

Definition 7. *Let $\mathcal{L} = (L_1, ..., L_k)$ be a collection of finite sets of literals. A set $H \subset$ **LB** is called a of $\mathcal{L}$ if $H \cap L_i \neq \emptyset$ for each $1 \leq i \leq k$. H is a minimal hitting set of $\mathcal{L}$ if it is a hitting set of $\mathcal{L}$ and no proper subset of H is a hitting set of $\mathcal{L}$.*

Algorithm $\mathcal{A}$ resolving the *EUP* for updates consistent with IC.

Input: a DB state I, an update $\Delta = (D^+, D^-)$, and IC Φ consistent with Δ. Let **lit** be the set of literals present either in I, or in Δ, or in $ground(\Phi)$.

Step 1.

Construct $ground(\Phi)$.

Construct M_Δ.

Eliminate from $ground(\Phi)$ every clause r such that for some l in its body $l \in D^-$ or $\neg.l \in M_\Delta$.

From the body of each remaining clause eliminate every $l \in M_\Delta$.

Let the resulting program be denoted by Φ_Δ.

Step 2.

Construct $\tilde{I} = ((I \cup M_\Delta) \setminus \neg.M_\Delta) \setminus D^-$.

Let $next(H)$ be the function which enumerates all subsets of $\tilde{I} \setminus M_\Delta$ in the lexicographic order.

Step 3.

Construct the set $\pi = \{(l, \mathcal{L}_l) \mid l \in \mathbf{LB},\ \mathcal{L}_l = \{c \mid c \subseteq \tilde{I}$ *and* $c = cr(T(l))$ *for some implication tree* $T(l)$ *of* $\Phi_\Delta \cup \tilde{I}\} \neq \emptyset\}$.

Step 4.
$H := \tilde{I} \setminus M_\Delta$
FOR_ALL ρ *compatible with* M_Δ **DO**
  $H_\rho := \emptyset \; ; b := false$
  WHILE $\neg b$ **DO**
    b := true
    FOR_ALL $l \in$ **lit DO**
      IF $(\rho(l, \neg.l) = l$ & **FOR_SOME** $c \in \mathcal{L}_{\neg.l}(c \cap H_\rho = \emptyset))$
      THEN $H_\rho := next(H_\rho); b := false; BREAK$
      END_IF
    END_FOR_ALL
  END_WHILE
  IF$H_\rho \subsetneq H$
  THEN $H := H_\rho$
  END_IF
END_FOR_ALL
Step 5.
  $I_1 := (\tilde{I} \setminus H) \cup \{l \mid (l, \mathcal{L}_l) \in \pi \text{ and } \exists\, c \in \mathcal{L}_l(c \cap H = \emptyset)\}.$
Output: I_1.

Algorithm $\mathcal{A}$ finds in the course of the outermost loop **FOR_ALL** at the step 4 a strategy ρ and a minimal hitting set H of

$$R(\rho) = \bigcup\{\mathcal{L}_{\neg.l} \mid (\neg.l, \mathcal{L}_{\neg.l}) \in \pi \;\&\; \rho(l, \neg.l) = l\}$$

which is optimal in the sense that for no subset $H' \subset H$ and for no preference strategy ρ' compatible with Δ, H' is a hitting set of $R(\rho')$. This optimal H is then used at the step 5 to construct the resulting DB state I_1.

Example 1. *Let* Φ *be defined as*

$$\Phi = \begin{cases} c & \leftarrow a, b \\ \neg c & \leftarrow d, f \\ e & \leftarrow d, b \\ \neg e & \leftarrow \end{cases}$$

$\Delta = (\{a, d\}, \emptyset)$, *and* $I = \{b.f, \neg e\}$. *Then* $M_\Delta = \{a, d, \neg e\}$, $\tilde{I} = \{b, f, \neg e, a, d\}$, $\pi = \{(c, \{a, b\}),\ (\neg c, \{d, f\}),\ (e, \{b, d\}),\ (\neg e, \{\neg e\})\}$, $\rho(c, \neg c) = \neg c$ *and* $\rho(e, \neg e) = \neg e$, $H = \{b\}$, *and the algorithm delivers the model* $I_1 = \{f, \neg e, a, d, \neg c\}$.

The following theorem shows that the algorithm $\mathcal{A}$ is correct.

Theorem 2. *Let* I_1 *be the DB state constructed by the algorithm* $\mathcal{A}$ *for IC* Φ, *a given DB state* I *and an update* $\Delta = (D^+, D^-)$. *Then*
  1) Δ *is accomplished in* I_1;

2) I_1 is consistent;

3) $I_1 \models \Phi$;

4) For no DB state $I' \models \Phi$ with accomplished Δ, $I' >^I_{it} I_1$;

5) For no DB state $I' \models \Phi$ with accomplished Δ, and such that $I_1 \equiv^I_{it} I'$, $I' >^I_{sd} I_1$.

The proof of this theorem is close to the proof of correctness of the *EUP*-algorithm $\mathcal{A}$ in [8]. The algorithm we present here is much more optimal.

Corollary 1. *(of nondegeneracy and implementation)*

(1) For any IC Φ and update Δ consistent with Φ there exists a conservative IC-enforcing update operator $\Psi = \Psi[\Phi, \Delta]$.

(2) The algorithm $\mathcal{A}$ implements some conservative IC-enforcing update operator.

From the axioms $AG^{\prec}$ it follows that the *EUP* in general case is reduced to this problem in the case of consistent updates and IC, at the cost of an additional loop on subsets of Δ. Indeed, the maximal subset of Δ consistent with Φ always exists and is constructed by the evident algorithm. After $\Delta^{\prec}_{\Phi\Delta}$ is found the algorithm $\mathcal{A}$ for consistent update and IC is applied.

5 The Case of Total Interpretations

Now we consider the *EUP* in the case of total interpretations. We presume that update $\Delta = (D^+, D^-)$ and IC Φ are consistent. This means that the program $\Phi_1 = \Phi \cup D^+ \cup \neg.D^-$ has a total model. Moreover, the set of literals $M_\Delta = lfp(T^{\in}_{\Phi_1})$ is consistent (though its positive part should not necessarily be a model of Φ_1). Let $M_\Delta = M^+_\Delta \cup \neg.M^-_\Delta$, where $M^+_\Delta = M_\Delta \cap \mathbf{B}$ and $M^-_\Delta = M_\Delta \cap \neg.\mathbf{B}$. It is clear that if I is a total model of Φ such that Δ is accomplished in I, then $M^+_\Delta \subseteq I$ and $I \cap \neg.M^-_\Delta = \emptyset$.

In the algorithm to follow we use a particular order function *next*. Let $X \subseteq \mathbf{B}, Y \subseteq \mathbf{B}$. We consider the following partial order $\prec_Y$ on subsets of X: $X_1 \prec_{X,Y} X_2$ holds for some $X_1, X_2 \subseteq X$ iff $X_1 \cap Y \subsetneq X_2 \cap Y$ or $X_1 \cap Y = X_2 \cap Y$ and $X_2 \setminus Y \subsetneq X_1 \setminus Y$. Let $next^X_Y(Z)$ be a function which enumerates all subsets of X in any linear order compatible with $\prec_{X,Y}$. For example one can use the lexicographic enumeration on $Z \cap Y$ combined with antilexicographic enumeration on $Z \setminus Y$.

The algorithm $\mathcal{B}$ below implements a conservative update operator which resolves the *EUP* for total DB states. It resembles the algorithm $\mathcal{A}$ but in fact it differs from it substantially in the steps 3 and 4.

Algorithm $\mathcal{B}$ resolving the *EUP* for total interpretations

Input: a total DB state I, and some consistent update $\Delta = (D^+, D^-)$ and IC Φ.

Step 1.

Construct $ground(\Phi)$ and $M_\Delta = M_\Delta^+ \cup M_\Delta^-$.

Eliminate from $ground(\Phi)$ every clause r such that for some l in its body either $\neg.l \in M_\Delta^+$, or $l \in M_\Delta^-$.

Then eliminate all clauses with literals in M_Δ in their heads.

From the body of each remaining clause eliminate every $l \in M_\Delta$.

The resulting program is denoted by Φ_Δ.

Step 2.

Construct $\tilde{I} = ((I \cup M_\Delta^+) \setminus \neg.M_\Delta^-$.

Step 3.

Set $\bar{\mathbf{L}}_\Delta = \mathbf{lit} \setminus (M_\Delta \cup \neg.M_\Delta)$

and

$$\pi = \{(l, \mathcal{L}_l) \mid l \in \mathbf{lit},\ \mathcal{L}_l = \{c \mid c \subseteq \bar{\mathbf{L}}_\Delta \ and\ c = cr(T(l)) \\ for\ some\ implication\ tree\ T(l)\ of\ \Phi_\Delta \cup \bar{\mathbf{L}}_\Delta\} \neq \emptyset\}.$$

Construct the set of literals which contribute to producing conflicts:

$$K := \{a \in \mathbf{B} | \exists(\rho\ compatible\ with\ M_\Delta)\exists l(\rho(l, \neg l) = l \& (\neg.l, \mathcal{L}_{\neg.l}) \in \pi\ \& \\ \exists(c \in \mathcal{L}_{\neg.l})(a \in c \vee \neg a \in c))\}.$$

Step 4.

$H := K \cap I$

FOR_ALL ρ *compatible with* M_Δ **DO**

    $H_\rho := K \setminus I;\ b := false$

    WHILE $\neg b$ & $(H_\rho \prec_{K,I} H)$ **DO**

        $b := true$

        FOR_ALL $l \in \mathbf{lit} \setminus M_\Delta$**DO**

        IF $(\rho(l, \neg.l) = l$ & **FOR_SOME** $c \in \mathcal{L}_{\neg.l}(c \cap (H_\rho \cup \neg.(K \setminus H_\rho)) = \emptyset))$

        THEN $H_\rho := next_I^K(H_\rho);\ b := false;\ BREAK$

        END_IF

        END_FOR_ALL

    END_WHILE

    IF $H_\rho \prec_{K,I} H$

    THEN $H := H_\rho$

    END_IF

END_FOR_ALL

Step 5.

$I' := (\tilde{I} \setminus H) \cup (K \setminus H)$

$I_1^+ := I' \cup \{a \in \mathbf{B} \mid (a, \mathcal{L}_a) \in \pi\ \&\ (H \cup \neg.(K \setminus H))\ is\ not\ a\ hitting\ set\ of\ \mathcal{L}_a\}$.

Output: $I_1 = I_1^+ \cup \neg.(\mathbf{B} \setminus I_1^+)$.

It is easy to check that algorithm $\mathcal{B}$ has the following properties.

Lemma 1. *(i) After the step 3* $\{a \in \mathbf{B} \mid (a, \mathcal{L}_a), (\neg a, \mathcal{L}_{\neg a}) \in \pi\} = \mathbf{B} \setminus (M_\Delta^+ \cup \neg.M_\Delta^-)$. *Moreover, for any* $a \in \mathbf{B} \setminus (M_\Delta^+ \cup \neg.M_\Delta^-)$ $\{a\} \in \mathcal{L}_a$ *and* $\{\neg a\} \in \mathcal{L}_{\neg a}$.

(ii) After the step 4 $H \cap M_\Delta = \emptyset$ *and for any* $a \in \mathbf{B} \setminus (M_\Delta^+ \cup \neg.M_\Delta^-)$ $a \in H$ *or* $\neg a \in \neg.(K \setminus H)$.

(iii) After the step 5 $M^+_\Delta \subseteq I^+_1$ *and* $\neg.M^-_\Delta \cap I^+_1 = \emptyset$*, hence the update* Δ *is accomplished in* I_1*.*

Lemma 2. $I_1 \models \Phi$.

Proof. Let $r = (l \leftarrow l_1, ..., l_k) \in ground(\Phi)$, $k \geq 0$ be such that $l_i \in I_1$ for all $1 \leq i \leq k$. If $l \in M_\Delta$ then by point (iii) of Lemma 1 $I_1 \models l$. Now suppose that $l \in \mathbf{LB} \setminus M_\Delta$ and let $r' = (l \leftarrow l'_1, ..., l'_{k_1})$ be the residue of rule r included into Φ_Δ. Then for every $1 \leq j \leq k_1$ $l'_j \notin M_\Delta$ and at the step 3 of $\mathcal{B}$ a set $\{l'_1, ..., l'_{k_1}\}$ is included in $\mathcal{L}_l$ and $(l, \mathcal{L}_l)$ is included in π. Since all $l'_j \in I_1$, then $\{l'_1, ..., l'_{k_1}\} \cap H = \emptyset$ and therefore $\rho(l, \neg.l) = l$ for that preference strategy ρ whose minimal H was found at the step 4. Then $l \notin H$ and $l \in I_1$. □

Using these lemmas it is easy to establish correctness of algorithm $\mathcal{B}$.

Theorem 3. *Let* I_1 *be the total DB state constructed by the algorithm* $\mathcal{B}$ *for some IC* Φ*, total DB state* I *and update* $\Delta = (D^+, D^-)$*. Then*

1) Δ *is accomplished in* I_1*;*

2) I_1 *is consistent;*

3) $I_1 \models \Phi$*;*

4) For no DB state $I' \models \Phi$ *with accomplished* Δ*,* $I' >^I_{it} I_1$*;*

5) For no DB state $I' \models \Phi$ *with accomplished* Δ*, and such that* $I_1 \equiv^I_{it} I'$*,* $I' >^I_{sd} I_1$*.*

The main problem with algorithm $\mathcal{B}$ is its complexity. Point (i) of lemma 1 shows that the conflict set at the step 3 includes contrary pairs for all literals (except those in M_Δ) independent of the initial DB state I. Therefore at the step 4 the number of considered different preference strategies is the same as the number of different models which should be considered by the straightforward brute force algorithm we mention in the previous section. We don't know whether the algorithm $\mathcal{B}$ can be so modified for general IC as to deal only with the conflicts produced by I or $\tilde{I}$. However it does so in a very special situation which is rather common in practice. In this situation there is an efficient algorithm resolving the *EUP*.

For a constraint $r = p(\bar{u}) \leftarrow \beta$ let us denote the predicate p by $def(r)$ and say that p is *defined* by r. Let $def(\Phi)$ denote the set of all predicates defined by constraints in Φ. For a literal $l = p(\bar{u})$ or $l = \neg p(\bar{u})$ $def(l) = p$. For an update Δ we denote by $def(\Delta)$ the set of predicates p such that $p = def(l)$ for some $l \in \Delta$.

Definition 8. *A normal IC* Φ *is* superstratified *if there exists a stratification* $def(\Phi) = \bigcup_{i=1}^{n} P_i, P_i \cap P_j = \emptyset, i < j \leq n$ *such that the following condition holds:*

- *Let* $r = h \leftarrow \beta_1, l, \beta_2$ *be a constraint in* Φ*,* $def(r) \in P_i$ *and* $def(l) \in P_j$*. Then* $j < i$*.*

The partition on $def(\Phi)$ induces the partition of Φ : $\Phi = \bigcup_{i=1}^{n} \Phi_i$, where Φ_i $(1 \leq i \leq n)$ is the set of constraints defining predicates in P_i. It is clear that superstratified IC are not recursive and that the constraints in Φ_1 are positive facts. The situation we consider is where Φ is superstratified and D^- does not affect the facts in Φ_1 and predicates in $P_i, i > 1$.

Algorithm $\mathcal{C}$
Input: a total DB state I, an update $\Delta = (D^+, D^-)$, a superstratified normal IC Φ with stratification $def(\Phi) = \bigcup_{i=1}^{n} P_i$, consistent with an update $\Delta = (D^+, D^-)$ such that $def(D^-) \cap P_i = \emptyset, 1 < i \leq n$.
Step 1.
Construct $ground(\Phi)$.
Eliminate from $ground(\Phi)$ every clause $r \leftarrow \beta_1, l, \beta_2$ with $l \in (D^- \cup \neg.D^+)$.
From the body of each remaining clause eliminate every $l_1 \in D^+$ and every $\neg.l_2$, where $l_2 \in D^-$. Let $\Phi_\Delta = \bigcup_{i=1}^{t} \Phi_{\Delta,i}$ be the resulting superstratified program.
Step 2.
Construct $J_1 = (I \cup \Phi_{\Delta,1} \cup D^+) \setminus D^-$.
Step 3.
Output: $I_1 = T^{\models}_{\Phi_{\Delta,t}} \uparrow \omega(...T^{\models}_{\Phi_{\Delta,2}} \uparrow \omega(T^{\models}_{J_1}(\emptyset))...)$.
Since all conflicts are resolved at the step 2 the following fact holds.

Theorem 4.
(1) The algorithm $\mathcal{C}$ *implements a conservative IC-enforcing update operator on total DB states, normal IC* Φ *superstratified by* $def(\Phi) = \bigcup_{i=1}^{n} P_i$, *and consistent with* Φ *updates* $\Delta = (D^+, D^-)$ *such that* $def(D^-) \cap P_i = \emptyset, 1 < i \leq n$.
(2) $\mathcal{C}$ *runs in polynomial time with respect to the summary size of* I, $ground(\Phi)$, *and* Δ.

Concluding Remarks

In this paper we present a formal framework for updating data bases with automatic restoration of integrity constraints. We distinguish such updates from the updates of knowledge bases, taking into account all models of integrity constraints, not only intended ones. We describe update operators axiomatically, and the only requirements we impose on them are: (1) accomplishing the maximal part of update consistent with IC, (2) restoring the IC, and (3) introducing minimal changes into the initial DB state, sufficient to achieve (1) and (2). The minimal change criterion combines two principles: that of maximal intersection and that of minimal symmetric difference. The so formalized *conservative* updates are complete in the sense that any correct DB state can be derived from any other correct DB state by some appropriate update. The conservative updates are quite reasonably implementable in the case of partial interpretations. We propose a principle-oriented algorithm computing the conservative updates, which is quite feasible in practical situations where a local change to a DB state does not introduce too many conflicts. On the other hand, in the case of total interpretations we can propose some principle-oriented algorithm only in a special class of normal stratified IC.

References

[1] Alchourón, C.E., Gärdenfors, P., and Makinson, D.: On the logic of theory change: partial meet contraction and revision functions. *J. Symbolic Logic.* **50** (1985) 510-530.

[2] Alferes, J.J.,Pereira, L.M.: Update-Programs Can Update Programs. In J.Dix, L.M. Pereira, T.C. Przymusinski, editors: *Second International Workshop, NMELP'96. Selected Papers.* LNCS **1216** (1996) 110-131.

[3] Apt, K.R., Bol, R.: Logic programming and negation: A survey. *Journal of Logic Programming.* **19-20** (1994) 9-71.

[4] Baral, C., Lobo, J.: Formal Characterization of Active Databases. In Pedreschi, D., Zaniolo, C., editors: *Logic in Databases. Intern. Workshop LID'96. Proceedings.* San Milano, Italy. (1996), 175-195.

[5] Baralis E., Ceri S., Paraboschi S.: Improved rule analysis by means of triggering and activation graphs. In T.Sellis, editor: *Rules in Database Systems,* LNCS **985** (1995) 165-181.

[6] Bidoit, N.: Negation in rule-based database languages: a survey. *Theoretical Computer Science.* **78** (1991) 3-83

[7] Dayal, U., Hanson,E., and Widom, J.: Active database systems. In: W. Kim, editor, *Modern Database Systems.* Addison Wesley (1995) 436-456.

[8] Dekhtyar, M., Dikovsky, A., Spyratos, N.: On Conservative Enforced Updates. In: Dix, J., Furbach, U., Nerode, A., editors: *Proceedings of 4th International Conference, LPNMR'97.* Dagstuhl Castle, Germany, LNCS **1265** (1997) 244-257.

[9] Eiter, T., Gottlob, G.: On the complexity of propositional knowledge base revision, updates, and counterfactuals. Artificial Intelligence **57** (1992) 227-270.

[10] Fernandez, J.A., Grant, J., Minker, J.: Model-Theoretic Approach to View Updates in Deductive Databases. *Journ. of Automated Reasoning,* **17**(1996) 171-197.

[11] Gärdenfors, P., Makinson, D.: Revisions of knowledge systems using epistemic entrenchment. In: *Proceedings of the Second Conference on Theoretical Aspects of Reasoning about Knowledge* , Pacific Grove, CA (1988) 83-95.

[12] Gottlob, G., Moerkotte, G., Subrahmanian, V.S.: The PARK semantics for active databases. In: *Proceedings of EDBT'96.* Avignon, France (1996).

[13] Gelfond, M., Lfschitz, V.: The stable semantics for logic programs. In: R.Kovalsky and K.Bowen, editors, *Proc. of the 5th Intern. Symp. on Logic Programming.* Cambridge, MA, MIT Press (1988) 1070-1080.

[14] Halfeld Ferrari Alves, M., Laurent, D., Spyratos, N., Stamate, D.: Update rules and revision programs. Rapport de Recherche Université de Paris-Sud, Centre d'Orsay, LRI **1010** (12 / 1995).

[15] Katsuno, H., Mendelzon, A. O.: Propositional knowledge base revision and minimal change. Artificial Intelligence **52** (1991) 253-294.

[16] Marek, V.W., Truszcińsky, M.: Revision programming, database updates and integrity constraints. In: *International Conference on Data Base theory, ICDT.* LNCS **893** (1995) 368-382.

[17] Picouet, Ph., Vianu, V.: Expressiveness and Complexity of Active Databases. In: Afrati, F., Kolaitis, Ph., editors, *6th Int. Conf. on Database Theory, ICDT'97.* LNCS **1186** (1997) 155-172.

[18] Przymusinski, T.C., Turner, H.: Update by Means of Inference Rules. In: V.W.Marek, A.Nerode, M.Truszczyński, editors, *Logic Programming and Non-monotonic Reasoning.* Proc. of the Third Int. Conf. LPNMR'95, Lexington, KY, USA (1995) 166-174.

[19] Raschid, L., Lobo, J.: Semantics for Update Rule Programs and Implementation in a Relational Database Management System. ACM Trans. on Database Systems **21** (December 1996) 526-571.

[20] Reiter, R.,: A logic for default reasoning. *Artificial Intelligence.* **13** (1980) 81-132.

[21] Schewe, K.-D., Thalheim, B.: Consistency Enforcement in Active Databases. In: Chakravarty, S., Widom, L., editors: *Research Issues in Data Engineering - Active Databases.* Proceedings. Houston, (1994).

[22] Van Gelder, A., Ross, K.A., and Schlipf, J.S.: The Well-Founded Semantics for General Logic Programs. Journal of the ACM **38** (1991) 620-650.

[23] Witteveen, C., van der Hoek, W.: A General Framework for Revising Non-monotonic Theories. In Dix, J., Furbach, U., Nerode, A., editors: *Proceedings of 4th International Conference, LPNMR'97.* Dagstuhl Castle, Germany, LNCS **1265** (1997) 258-272.

[24] Zaniolo, C.: Active database rules with transaction-consious stable-model semantics. In:*Proceedings of ourth International Conference, DOOD'95.* LNCS **1013** (1995) 55-72