Enforcement of Integrity Constraints by Means of Minimal Sufficient Changes *

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 ,
and
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, and an external update Δ which specifies the data to be added and the data to be deleted, one should introduce minimal changes in the given initial data base state I, sufficient to accomplish Δ and to restore T if and when it was violated. We give an axiomatic description of its solutions and describe a resolving algorithm.

1 Introduction

In this paper we consider an interesting problem which concerns the new generation of data bases treating external updates in an intelligent way. Such data bases use rule based systems which control the process of online integrity constraints restoration. The problem we investigate (we call it an *Enforced Update Problem*) is as follows: given a theory T which formalizes integrity constraints, and given an external update Δ which specifies the data to be added and the data to be deleted, one should introduce minimal changes in the initial data base state I, sufficient to accomplish Δ and to restore T if and when it was violated. This problem was first investigated in 80-ies in the class of propositional knowledge bases. From the very beginning it was clearly distinguished from a similar problem of knowledge base revision. In 90-ies both problems were attacked for 1st order knowledge bases. However, no clear distinction was made between the Enforced Update Problem for 1st order knowledge bases and data bases. We bridge this gap

*This work was sponsored by the Russian Fundamental Studies Foundation (Grants 95-01-01321, 97-01-00973).

explaining below the deep difference between the two cases and providing an axiomatic definition of solutions of the problem for 1st order data bases.

The paper consists of two parts. In the first part we give a brief survey of the research concerning revision and updates of knowledge bases and data bases. We make clear the difference between them and classify various approaches to 1st order DB updates. In the second part we introduce a class of the so called conservative DB update operators enforcing integrity constraints, which resolve the Enforced Update Problem as applied to data bases, and we describe an algorithm implementing them.

2 Revision vs. Update

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. A very interesting class of KBs is the class of Deductive Data Bases (*DDBs*) with actions. DDBs are described in terms of generalized logic programs of various kinds. Actions can be external or proper (produced by the DDB itself) and they can change DDB states. On the other hand, speaking of DBs one has in mind a system of invariable conditions on states, *integrity constraints* (IC), which manifest some idea of "acceptable data". The IC are optional for KBs. A new data (knowledge) can be seen as some formula p which should be somehow incorporated into the DB or KB. Two main approaches to bringing up to date KBs and DBs can 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 p and describes the resulting class of states (worlds) in terms of another theory T'. In other words the revision is theory oriented. 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 modifications of KBs was made by Gärdenfors and his colleagues [1, 14] who have proposed some philosophically founded set of postulates describing KB modifications with respect to new knowledge. Later Katsuno and Mendelzon [19] have refined these postulates in propositional case. They were first to point out the distinction between the notions of revision and update. In [18] they have proposed the following sets of postulates, formalizing the two. In these postulates T, T_1, T_2 denote propositional theories, p, p_1, p_2 denote propositional formulas which express a new knowledge, $\circ$ denotes revision operator and $\diamond$ denotes update operator.

Revision axioms:
(R1) $T \circ p$ implies p.
(R2) If $T \wedge p$ is satisfiable, then $T \circ p \equiv T \wedge p$.
(R3) If p is satisfiable, then $T \circ p$ is also satisfiable.
(R4) If $T_1 \equiv T_2$ and $p_1 \equiv p_2$, then $T_1 \circ p_1 \equiv T_2 \circ p_2$.
(R5) $(T \circ p_1) \wedge p_2$ implies $T \circ (p_1 \wedge p_2)$.
(R6) If $(T \circ p_1) \wedge p_2$ is satisfiable, then $T \circ (p_1 \wedge p_2)$ implies $(T \circ p_1) \wedge p_2$.

Update axioms:
(U1) $T \diamond p$ implies p.

(U2) If T implies p, then $T \diamond p$ is equivalent to T.
(U3) If both T and p are satisfiable, then $T \diamond p$ is satisfiable.
(U4) If $T_1 \equiv T_2$ and $p_1 \equiv p_2$, then $T_1 \diamond p_1 \equiv T_2 \diamond p_2$.
(U5) $(T \diamond p_1) \wedge p_2$ implies $T \diamond (p_1 \wedge p_2)$.
(U6) If $T \diamond p_1$ implies p_2 and $T \diamond p_2$ implies p_1, then $T \diamond p_1 \equiv T \diamond p_2$.
(U7) If T is complete [1], then $(T \diamond p_1) \wedge (T \diamond p_2)$ implies $T \diamond (p_1 \vee p_2)$.
(U8) $(T_1 \vee T_2) \diamond p \equiv (T_1 \diamond p) \vee (T_2 \diamond p)$.

One can see from these axioms that revision and update behave differently with respect to consistency of $T \wedge p$. The axiom (R2) says that when $T \wedge p$ is consistent it is equivalent to the revision (a theory oriented description). Whereas the update axioms admit the resulting models which do not satisfy T. Another difference in behavior is due to the axiom (U2) which implies that contradictive KBs T remain contradictive after updates. As to the revisions, the result in this situation depends on satisfiability of p, which is seen from axiom (R3). The pointwise model based definition of updates is due to the axiom (U8). Quite nonevidently these axioms express a very important principle, that of a *minimal change* produced in initial models to obtain the resulting models. This principle is kept in any specific operator of revision or update. In general it is expressed through some explicit criterion of minimal difference between two models. E.g. Dalal [7] measures the distance between two models by the size of their symmetric difference [2] $|M_1 \nabla M_2|$ and the distance between two theories by

$$|\nabla|(T_1, T_2) = min\{|M_1 \nabla M_2| : M_1 \in Models(T_1), M_2 \in Models(T_2)\}.$$

His revision operator $\circ_D$ meets axioms (R1)-(R6) and is defined for a satisfiable T as

$$Models(T \circ_D p) = \{M \in Models(p) : \exists M' \in Models(T) \; [|M \nabla M'| = |\nabla|(T, p)]\}$$

and for unsatisfiable T as $Models(T \circ_D p) = Models(p)$. Very close to this is the definition in [13] of the update operator $\diamond_F$ which meets the axioms (U1)-(U8). Forbus measures the distance between a model M and a formula p by

$$|\nabla|_M(p) = min\{|M \nabla M'| : M' \in Models(p)\}$$

and defines the update by

$$Models(T \diamond_F p) = \bigcup_{M \in Models(T)} \{M' \in Models(p) : |M' \nabla M| = |\nabla|_M(p)\}.$$

These revision and update operators being applied to the same T and p can give different results even in very simple cases.

Example 1. *Let* $T = a \wedge (b \leftrightarrow c)$ *and* $p = \neg a \wedge b$. *Then* $Models(T)$ *consists of models* $M_1 = \{a\}, M_2 = \{a, b, c\}$, *and* $Models(p)$ *consists of models* $M_1' = \{b\}, M_2' = \{b, c\}$. $T \wedge p$ *is unsatisfiable. So* $|\nabla|(T, p) > 0$. *It is clear that* $|\nabla|_{M_1}(p) = 2, |\nabla|_{M_2}(p) = 1$, *and therefore* $|\nabla|(T, p) = 1$. *Then*

$$T \circ_D p \equiv \neg a \wedge b \wedge c$$

and

$$T \diamond_F p \equiv \neg a \wedge b \wedge c \vee \neg a \wedge b \wedge \neg c \equiv \neg a \wedge b = p.$$

[1] A theory T is *complete* if for any formula p either $T \Rightarrow p$ or $T \Rightarrow \neg p$.
[2] $M_1 \nabla M_2 = (M_1 \setminus M_2) \cup (M_2 \setminus M_1)$

Various other criteria of minimal change of propositional KBs and DBs were introduced since early 80-ies. Their definitions as well as references can be found in the paper [10] where their computational complexity is explored.

In 1st order case the situation with DBs and KBs modifications is more complex. In this case a KB T is seen as a theory, or a logic program, or a system of rules (productions) with distinguished *extensional part*. This part explicitly represents some finite set of given facts. The *intensional part* of knowledge is embodied by non extensional fragments of models of T according to some given semantical definition which is sometimes based on default rules. These fragments can be infinite when functors are available. Optionally in the case of external updates the KBs are supplied by IC expressed by formulas or some other logic programs which filter out the resulting models. As to DBs, they are seen in 1st order case as finite DB states satisfying IC expressed by some 1st order theory or a logic program. One of the problems in 1st order case is the absense of a satisfactory general constructive semantics of negation. Various particular semantics were proposed since late seventies (see e.g. [6, 3, 24, 29]). Moreover, there are two main classes of interpretations of the negation: the classical *total* (2-valued) interpretations and the *partial* (3-valued) interpretations. Under a total interpretation 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 facts in the models of KBs and in the DB states are atoms, whereas under partial interpretations they are literals.

Basically there are two formal frames of defining the model updates in 1st order DBs and KBs: *sequential* and *non sequential*. The way the updates are treated in active DBs is a good illustration of the sequential framework. The active DBs (see [8, 22]) are supplied by some sort of automatic change mechanism which reacts to updates violating the IC and modifies DB states in order to restore the IC. Consequently this mechanism represents the pointwise DB states updates. In general the sequential change mechanism is based on a system of active rules of the form

$$e_1, ..., e_k, l_1, ..., l_m \longrightarrow a_1, ..., a_n$$

where e_i are some events (mostly updates or update-requests), l_j are tests, and a_t are actions (mostly elementary updates, i.e. insertions or deletions of facts). Sequential change process is initiated by some external events $\bar{u}_0$ which happen in a DB state S_0 satisfying IC Φ. These events activize those rules whose premise events are covered by $\bar{u}_0$ and whose tests are satisfied in S_0. One of the activized rules is chosen and fired. Its actions produce new events $\bar{u}_1$ and change the former DB state to some new state S_1. $\bar{u}_1$ in its turn activize new rules, and so on till we arrive at a DB state S_n satisfying Φ. A particular choice of a fired rule is performed by a strategy which should guarantee the determinicity or the confluence and the successful termination of the process. In active DBs this strategy is implemented by the so called trigger system. Various definitions of sequential semantics can be found in the literature (see [5, 4, 28]). Since the desired properties of confluence and termination are rather hard to achieve, in most papers dealing with the sequential framework some deterministic strategies are considered (see e.g [32, 15, 26]).

Until recently the nonsequential framework was applied to KB updates. This framework presents for each KB P, update Δ and IC Φ a partial operator on the set of interpretations $\mathcal{I}$

$$\Psi_{P,\Delta,\Phi} : \mathcal{I} \mapsto 2^{\mathcal{I}} \; such \; that \; \forall I \in \mathcal{I}. \Psi_{P,\Delta,\Phi}(I) \models \Phi$$

Normally, some kind of the minimal change restriction is imposed on Ψ.

There are three main techniques behind nonsequential operators: program justified revision [21], inertia axioms [25], and conflict resolution. The name of the first technique seems not too much appropriate. The idea is to construct the closest (in terms of the symmetric difference) stable model of the program which is the reduct of the IC Φ with respect to the initial interpretation I. Thus it is rather a pointwise model update [3]. It is worth noting that this operator is partial and is based on some variant of the Reiter Closed World Assumption [27] when applied to partial interpretations. The idea behind the inertia axioms is that some special axioms are added to a KB, which specify intended models of the IC. In [25] this technique is proven to be more general than that of program justified revision. The following simplified example illustrates the difference between the two techniques.

Example 2. *Let P be the KB $\{panic \leftarrow alarm;\ sleep \leftarrow \mathbf{not}\ alarm\}$, $\Phi = \{sleep \leftarrow\}$, the initial interpretation $I = \{alarm, panic\}$. I is a model of P but not of Φ. Suppose we are to update I by deletion of $\{alarm\}$. Then the justified revision gives the interpretation $I' = \{sleep, panic\}$. If we add to P the inertia axiom*

$$\mathbf{not}\ panic \leftarrow \mathbf{not}\ alarm$$

(which together with P says that alarm is the only reason of panic), then the resulting interpretation is $I'' = \{sleep\}$.

The inertia axioms technique was applied in [2] to perform the so called "program updating". This is a transformation of a given KB P with respect to given update rules Δ, which adds to their union a set of inertia axioms. The resulting KB $pu(P, \Delta)$ has the property that Δ-justified revisions of "good" models of P are good models of $pu(P, \Delta)$. In [2] the "good" models are the well-founded models of [30, 29].

Conflict resolution technique provides operators $\Psi_{P,\Delta,\Phi}$ whose specific feature is some strategy of constructing the model $\Psi_{P,\Delta,\Phi}(I)$ through discovery and reconciliation of conflicts in the set of consequences of I via P, Φ, and Δ.

This technique was applied e.g. in [12] to perform view updates in disjunctive DDBs. By a view in a DDB one means a "good" model of the intensional part of a KB. In other words the problem in [12] is how one should change the extensional part of the KB P in order to accomplish the given update Δ of the intensional part and to satisfy the IC Φ.

The common feature of all update and revision techniques as applied to KBs is that they deal with only *"good"* (minimal, well-founded, perfect, stable, etc.) models. This is quite reasonable for systems with intensional knowledge. However this approach does not apply to DBs where *any* model of IC is an acceptable DB state. In DBs the IC Φ should never be violated. So in the case where an update Δ contradicts Φ, the best that could be done is to accomplish a "maximal" part of Δ consistent with Φ. Therefore some specific technique is needed to accomplish DB updates. In the next section we establish a conflict resolution technique for DB updates.

[3] The true 1st order theory revision is investigated e.g. in [31]

3 Conservative updates

3.1 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.* 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 *DB states.* A DB state is *consistent* if it does not contain contrary pairs. A constraint $r = (l \leftarrow l_1, ..., l_n)$ is *valid in a DB state* I if $l \in I$ whenever $l_i \in I$ for each $1 \leq i \leq n$. IC Φ *are valid in a DB state* I (I satisfies Φ) if every constraint in Φ is valid in I. This is denoted by $I \models \Phi$. Intuitively, a consistent DB-state for which $I \models \Phi$, constitutes a partial model of Φ. The positive literals in I are considered as true, the negative, as false, and those in $\mathbf{LB}_C \setminus I$, as unknown.

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. We say that $\Delta = (D^+, D^-)$ is *accomplished* in I if $D^+ \subseteq I$ and $D^- \cap I = \emptyset$.

An *implication tree* of a ground IC Φ for a literal l is a finite tree $T(l)$ whose nodes are labeled by ground literals with no one 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 $n \geq 0$, 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.

3.2 Enforced update problem

We consider the following problem which we call an *Enforced Update Problem* (*EUP*) and which concerns active data bases [8, 22] and deductive data bases with updates [11]. 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 changes should be performed in the initial state, sufficient to make the resulting DB state satisfy the first two claims.

The *EUP* is resolved in [17, 20] for rules of the form $l \leftarrow l_1$. It differs from the above-mentioned analogous problems concerning KBs in that no intended model claim is imposed on the solutions. Our goal is to give a complete conflict resolution style solution to the *EUP*. The operators providing solutions to the *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 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. This is why we put the symmetric difference criterion on top of the intersection criterion (see also the discussion in [20]). The criterion of maximal intersection as a concept of closeness has been frequently used, see for example [19, 10, 21, 25, 17]. 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* intersection-closer *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* symmetric-difference-closer *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* consistent *with IC* Φ *if there exists a consistent DB state* I *such that*

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

In the situation where Δ is consistent with Φ it must be accomplished completely. The other situation is that where Δ contradicts Φ. In this situation one should define a part of Δ consistent with Φ, which should be accomplished. In this paper we treat the first situation. Therefore, in the sequel we presume that the claim of consistency holds. 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 strong immediate consequence operator *is the total operator*

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

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

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

2.Evidently, $T^{\in}$ is continuous, i.e. for any nondecreasing sequence of partial interpretations $I_1 \subseteq I_2 \subseteq ...$ $T^{\in}(\bigcup_{i=1}^{\infty} I_i) = \bigcup_{i=1}^{\infty} T^{\in}(I_i)$. Therefore the least fixed point of $T^{\in}$ is defined as $lfp(T^{\in}) = T^{\in}{\uparrow}\omega = M_{min}(\Phi)$, 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. Therefore, 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. $M_\Delta = lfp(T^{\in}_{\Phi \cup D^+})$, 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.

Now we arrive at our axiomatic definition of the operators resolving the *EUP*.

Definition 4. *Let the given update Δ be consistent with IC Φ. An operator $\Psi = \Psi[\Phi, \Delta]$ on the set of partial interpretations is called a* conservative IC enforcing update operator *if for each I :*

(1) $\Psi(I)$ is a model of Φ (i.e. $\Psi(I)$ is consistent and $T^{\in}(\Psi(I)) \subseteq \Psi(I)$),

(2) Δ is accomplished in $\Psi(I)$,

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

(4) 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.

The points (1) and (2) together with consistency of Δ and Φ correspond in the propositional case to the fact that the conservative operators meet both axiom sets: for KBs revisions and for KB updates. So 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 is taken for granted and the theory is changed when it contradicts the update, whereas for the latter, just the opposite, the IC cannot be changed (point(1)) and the update should be reconciled to it in the case of contradictions. The point (2) is to be revised in the caseof

Definition 5. *A* preference strategy *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* compatible *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* compatible *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 6. *Let* $\mathcal{L} = (L_1, ..., L_k)$ *be a collection of finite sets of literals. A set* $H \subset \mathbf{LB}$ *is called a* hitting set *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*

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

Step 1.
Construct $ground(\Phi)$.
Construct M_Δ.
Eliminate from $ground(\Phi)$ every clause r such that for some $l \in body(r)$, $l \in D^-$ or $\neg.l \in M_\Delta$.
From the body of each remaining clause eliminate every $l \in M_\Delta$.
The resulting program is denoted by Φ_Δ.
Step 2.
Construct $\tilde{I} = ((I \cup M_\Delta) \setminus \neg.M_\Delta) \setminus D^-$.
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.
Guess a preference strategy ρ compatible with Δ.
$R(\rho) := \{\neg.l \mid (\neg.l, \mathcal{L}_{\neg.l}) \in \pi \ \&\ \rho(l, \neg.l) = l\}$;
Step 5.
$\mathcal{L}(\rho) := \bigcup\{\mathcal{L}_l \mid l \in R(\rho)\}$;
Step 6.
Find the first minimal hitting set H of $\mathcal{L}(\rho)$ such that for no subset $H' \subset H$ and for no preference strategy ρ' compatible with Δ, H' is a hitting set of $\mathcal{L}(\rho')$.
Step 7.
$I_1 := (\tilde{I} \setminus H) \cup \{l \mid (l, \mathcal{L}_l) \in \pi$ *and* $\exists\ c \in \mathcal{L}_l (c \cap H = \emptyset)\}$.
Output: I_1.

Algorithm $\mathcal{A}$ is nondeterministic. At the step 4 it guesses some preference strategy and then checks at the step 6 that the found ρ is optimal, i.e. there exists a hitting set H of $\mathcal{L}(\rho)$, whose proper subsets are hitting sets for no strategies compatible with Δ. This optimal H is then used at the step 7 to construct the resulting DB state I_1. Let us observe that both steps of

choice: 4 and 6 are correct. Indeed, at the step 4 one really can guess some preference strategy ρ compatible with Δ, because Φ is consistent with Δ, and so $M_\Delta \cap D^- = \emptyset$. On the other hand, if there exists a minimal hitting set for some preference strategy compatible with Δ, then there evidently exists some hitting set H satisfying the condition at the step 6 (if there is a subset $H' \subset H$ which is a hitting set of $\mathcal{L}(\rho')$ for some preference strategy ρ', then one could guess ρ' in place of ρ at the step 4).

Example 3. *Let Φ be defined as*

$$\Phi = \left\{ \begin{array}{rl} c & \leftarrow a, b \\ \neg c & \leftarrow d, f \\ e & \leftarrow d, b \\ \neg e & \leftarrow \end{array} \right.$$

$\Delta = (\{a, d\}, \emptyset)$, $I = \{b.f, \neg e\}$, and $I_0 = \{f, \neg e, a, d\}$. 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 $I_1 \equiv^I_{it} I'$, $I' >^I_{sd} I_1$.

This theorem is in fact stronger than it says:

Theorem 3. *(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.

The algorithm $\mathcal{A}$ resolving the *EUP* is rather complex. Theoretically it is in *PSPACE*. However, as it is shown in [9], $\mathcal{A}$ can be optimized in many ways. It can be reorganized so as to become incremental (which it is not in the present setting). The optimized algorithm works in polynomial time for quite a broad class of IC. Moreover, due to a standard technique of intelligent ground closure construction and specialization it can be made quite practical.

Concluding Remarks

In this paper we have presented a formal framework for updating data bases with automatic restoration of integrity constraints. We distinguish such updates from revisions 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 update, (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. We sketch an algorithm computing the conservative updates and describe elsewhere various possibilities of practical optimizations ([9]). We consider elsewhere the case where updates contradict integrity constraints and are accomplished partially.

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] Dalal M.: Investigations into a theory of knowledge base revision: preliminary report. In: *Proceedings AAAI-88*, St. Paul, MN (1988) 475-479.

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

[9] 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.

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

[11] Fagin, R., Kuper, G., Ullman, J., and Vardi, M.Y.: Updating Logical Databases. In: P. Kanellakis, editor, *Advances in Computing Research*, JAI Press **3** (1986) 1-18.

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

[13] Forbus K.D.:Introducing actions into qualitative simulation. In: *Proceedings of IJCAI-89* (1989) 1273-1278.

[14] 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.

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

[16] 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.

[17] 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).

[18] Katsuno, H., Mendelzon, A. O.: On the difference between updating a knowledge base and revising it. In: J.A.Allen, R.Fikes, and E.Sandewell,eds., *Principles of Knowledge Representation and Reasoning: Proceedings of the Second Int. Conference* San Mateo, CA (1991) 387-395.

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

[20] Laurent, D., Spyratos, N., Stamate, D.: Deterministic enforcement of constraints. *This special issue.*

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

[22] Mueck T.A. Active Databases: Concepts and Design Support: *Advances in Computers.* **39** (1994), 107-185.

[23] 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.

[24] Przymusinski, T.C.: Reasoning with Knowledge and Belief. *This special issue.*

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

[26] 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.

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

[28] 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).

[29] Van Gelder, A.: A Tutorial on the Well-Founded Semantics. *This special issue.*

[30] 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.

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

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