Applied Mathematics & Information Sciences
An International Journal

A Rational and Efficient Algorithm for View Revision in Databases

Radhakrishnan Delhibabu[1,2] *and Gerhard Lakemeyer*[1]

[1] Informatik 5, Knowledge-Based Systems Group, RWTH Aachen, Germany.
[2] Department of Computer Science and Engineering, SSN College of Engineering in Chennai, India.

Received: 11 Sep. 2012, Revised: 10Nov. 2012, Accepted: 19 Nov.2012
Published online: 1 May 2013

Abstract: The dynamics of belief and knowledge is one of the major components of any autonomous system that should be able to incorporate new pieces of information. In this paper, we argue that to apply rationality result of belief dynamics theory to various practical problems, it should be generalized in two respects: first of all, it should allow a certain part of belief to be declared as immutable; and second, the belief state need not be deductively closed. Such a generalization of belief dynamics, referred to as base dynamics, is presented, along with the concept of a generalized revision algorithm for Horn knowledge bases. We show that Horn knowledge base dynamics has interesting connection with kernel change and abduction. Finally, we also show that both variants are rational in the sense that they satisfy certain rationality postulates stemming from philosophical works on belief dynamics.

Keywords: AGM, Belief Update, Horn Knowledge Base Dynamics, Kernel Change, Abduction, View update

1 Introduction

Modeling intelligent agents' reasoning requires designing knowledge bases for the purpose of performing symbolic reasoning. Among the different types of knowledge representations in the domain of artificial intelligence, logical representations stem from classical logic. However, this is not suitable for representing or treating items of information containing vagueness, incompleteness or uncertainty, or Horn knowledge base evolution that leads the agent to change his beliefs about the world.

When a new item of information is added to a Horn knowledge base, inconsistency can result. Revision means modifying the Horn knowledge base in order to maintain consistency, while keeping the new information and removing (contraction) or not removing the least possible previous information. In our case, update means revision and contraction, that is insertion and deletion in database perspective. Our previous work [7,8] makes connections with contraction from knowledge base dynamics.

Our Horn knowledge base dynamics, is defined in two parts: an immutable part (Horn formulae) and updatable part (literals) (for definition and properties see works of Nebel [41] and Segerberg [45]). Knowledge bases have a set of integrity constraints (see the definitions in later section). In the case of finite knowledge bases, it is sometimes hard to see how the update relations should be modified to accomplish certain Horn knowledge base updates.

*Example 1.*Consider a database with an (immutable) rule that a staff member is a person who is currently working in the research group under the chair. Additional (updatable) facts are that matthias and gerhard are group chairs, and delhibabu and aravindan are staff members. We restricted that staff and chair names are taken by her/his email id, and our integrity constraint is that each research group has only one chair ie. $\forall x,y,z$ (y=x) $\leftarrow$ group_chair(x,y) $\wedge$ group_chair(x,z).

Immutable part: staff_chair(X,Y)$\leftarrow$
staff_group(X,Z),group_chair(Z,Y).

Updatable part: group_chair(infor1,matthias)$\leftarrow$
group_chair(infor2,gerhard)$\leftarrow$
staff_group(delhibabu,infor1)$\leftarrow$
staff_group(aravindan,infor1)$\leftarrow$

Suppose we want to update this database with the information, staff_chair(delhibabu,aravindan), that is

staff_chair(delhibabu,aravindan)$\leftarrow$
staff_group(delhibabu,Z) $\bigwedge$ group_chair(Z,aravindan)

* Corresponding author e-mail: delhibabu@kbsg.rwth-aachen.de

If we are restricted to definite clauses, there is only one plausible way to do this: delhibabu and aravindan belong to groups infor1, this updating means that we need to delete (remove) matthias from the database and newly add (insert) aravindan to the database (aravindan got promoted to the chair of the research group infor1 and he was removed from research group infor1). This results in an update that is too strong. If we allow disjunctive information into the database, however, we can accomplish the update by minimal adding wrt consistency

staff_group(delhibabu,infor1) ∨
group_chair(infor1,aravindan)

and this option appears intuitively to be correct.

When adding new beliefs to the Horn knowledge base, if the new belief is violating integrity constraints then belief revision needs to be performed, otherwise, it is simply added. As we will see, in these cases abduction can be used in order to compute all the possibilities and it is *not up to user or system* to choose among them.

When dealing with the revision of a Horn knowledge base (both insertions and deletions), there are other ways to change a Horn knowledge base and it has to be performed automatically also. Considering the information, change is precious and must be preserved as much as possible. The *principle of minimal change* [22, 44] can provide a reasonable strategy. On the other hand, practical implementations have to handle contradictory, uncertain, or imprecise information, so several problems can arise: how to define efficient change in the style of AGM [1]; what result has to be chosen [27,32,39]; and finally, according to a practical point of view, what computational model to support for Horn knowledge base revision has to be provided?

The rest of paper is organized as follows: First we start with preliminaries in Section 2. In Section 3, we introduce knowledge base dynamics along with the concept of generalized revision, and revision operator for knowledge base. Section 4 studies the relationship between knowledge base dynamics and abduction. In Section 5, we discuss an important application of knowledge base dynamics in providing an axiomatic characterization for insertion view atoms to databases; and brief summary of the related works nature of view update problem for incomplete to complete information. In Section 6 we give brief overview of related works. In Section 7 we make conclusions with a summary of our contribution as well as a discussion of future directions of investigation. All proofs can be found in the Appendix.

2 Preliminaries

We consider a propositional language $\mathscr{L}_{\mathscr{P}}$ defined from a finite set of propositional variables $\mathscr{P}$ and the standard connectives. We use lower case Roman letters $a,b,x,y,...$ to range over elementary letters and the Greek letters $\varphi,\phi,\psi,...$ for propositional formulae. Sets of formulae are denoted by upper case Roman letters $A,B,F,K,.....$ A literal is an atom (positive literal), or a negation of an atom (negative literal).

For any formula φ, we write $E(\varphi)$ to mean the set of the elementary letters that occur in φ. The same notation also applies to a set of formulae. For any set F of formulae, $L(F)$ represents the sub-language generated by $E(F)$, i.e. the set of all formulae φ with $E(\varphi) \subseteq E(F)$.

Horn formulae are defined [15] as follows:

1. Every $a \in \Phi$, a and $\neg a$ are Horn clauses.
2. $a \leftarrow a_1 \wedge a_2 \wedge ... \wedge a_n$ is a Horn clause, where $n \geq 0$ and $a, a_i \in \Phi$ $(1 \leq i \leq n)$.
3. Every Horn clause is a Horn formula, a is called head and a_i is body of the Horn formula.
4. If φ and ψ are Horn formulae, so is $\varphi \wedge \psi$.

A definite Horn clause is a finite set of literals (atoms) that contains exactly one positive literal which is called the head of the clause. The set of negative literals of this definite Horn clause is called the body of the clause. A Horn clause is non-recursive, if the head literal does not occur in its body. We usually denote a Horn clause as head←body. Let $\mathscr{L}_{\mathscr{H}}$ be the set of all Horn formulae with respect to $\mathscr{L}_{\mathscr{P}}$.

Formally, a finite Horn knowledge base KB is defined as a finite set of formula from language $\mathscr{L}_{\mathscr{H}}$, and divided into three parts: an immutable theory KB_I is an Horn formulae (head←body), which is the fixed part of the knowledge; updatable theory KB_U is Horn clause (head←); and an integrity constraints KB_{IC} is Horn clause (←body).

Definition 1(Knowledge Base). *Let KB be a finite set of Horn formulae from language $\mathscr{L}_{\mathscr{H}}$ called a Horn knowledge base with, $KB = KB_I \cup KB_U \cup KB_{IC}$, $KB = KB_I \cap KB_U = \varnothing$ and $KB = KB_U \cap KB_{IC} = \varnothing$.*

Working with deductively closed, infinite belief sets is not very attractive from a computational point of view. The AGM approach to belief dynamics is very attractive in its capturing the rationality of change, but it is not always easy to implement either Horn formula based partial meet revision, or model-theoretical revision. In real application from artificial intelligence and database, what is required is to represent the knowledge using a finite Horn knowledge base. Further, a certain part of the knowledge is treated as immutable and should not be changed.

Knowledge base change deals with situations in which an agent has to modify its beliefs about the world, usually due to new or previously unknown incoming information, also represented as formulae of the language. Common operations of interest in Horn knowledge base change are the expansion of an agent's current Horn knowledge base KB by a given Horn clause φ (usually denoted as KB+φ), where the basic idea is to add regardless of the consequences, and the revision of its current beliefs by φ (denoted as KB * φ), where the

intuition is to incorporate φ into the current beliefs in some way while ensuring consistency of the resulting theory at the same time. Perhaps the most basic operation in Horn knowledge base change, like belief change, is that of contraction (AGM [1]), which is intended to represent situations in which an agent has to give up φ from its current stock of beliefs (denoted as KB-φ).

Definition 2(Levi Identity). *Let - be an AGM contraction operator for KB. A way to define a revision is by using Generalized Levi Identity:*

$$KB * \alpha = (KB - \neg\alpha) \cup \alpha$$

Then, the revision can be trivially achieved by expansion, and the axiomatic characterization could be straightforwardly obtained from the corresponding characterizations of the traditional models [17]. The aim of our work is not to define revision from contraction, but rather to construct and axiomatically characterize revision operators in a direct way.

3 Knowledge base dynamics

AGM [1] proposed a formal framework in which revision(contraction) is interpreted as belief change. Focusing on the logical structure of beliefs, they formulate eight postulates which a revision knowledge base (contraction knowledge base was discussed in [8]) has to verify.

Definition 3.*Let KB be a Horn knowledge base with an immutable part KB_I. Let α and β be any two Horn clauses from $\mathscr{L}_{\mathscr{H}}$. Then, α and β are said to be* KB-equivalent *iff the following condition is satisfied: $\forall$ set of Horn clauses $E \subseteq \mathscr{L}_{\mathscr{H}}$: $KB_I \cup E \vdash \alpha$ iff $KB_I \cup E \vdash \beta$.*

These postulates stem from three main principles: the new item of information has to appear in the revised Horn knowledge base, the revised base has to be consistent and revision operation has to change the least possible beliefs. Now we consider the revision of a Horn clause α wrt KB, written as $KB * \alpha$. The rationality postulates for revising α from KB can be formulated.

Definition 4(Rationality postulates for Horn knowledge base revision).

*(KB*1)* Closure: *$KB * \alpha$ is a Horn knowledge base.*
*(KB*2)* Weak Success: *if α is consistent with $KB_I \cup KB_{IC}$ then $\alpha \subseteq KB * \alpha$.*
*(KB*3.1)* Inclusion: *$KB * \alpha \subseteq Cn(KB \cup \alpha)$.*
*(KB*3.2)* Immutable-inclusion: *$KB_I \subseteq Cn(KB * \alpha)$.*
*(KB*4.1)* Vacuity 1: *if α is inconsistent with $KB_I \cup KB_{IC}$ then $KB * \alpha = KB$.*
*(KB*4.2)* Vacuity 2: *if $KB \cup \alpha \nvdash \perp$ then $KB * \alpha = KB \cup \alpha$.*
*(KB*5)* Consistency: *if α is consistent with $KB_I \cup KB_{IC}$ then $KB * \alpha$ consistent with $KB_I \cup KB_{IC}$.*
*(KB*6)* Preservation: *If α and β are KB-equivalent, then $KB * \alpha \leftrightarrow KB * \beta$.*
*(KB*7.1)* Strong relevance: *$KB * \alpha \vdash \alpha$ If $KB_I \nvdash \neg\alpha$*
*(KB*7.2)* Relevance: *If $\beta \in KB \backslash KB * \alpha$, then there is a set KB' such that $KB * \alpha \subseteq KB' \subseteq KB \cup \alpha$, KB' is consistent ($KB_I \cup KB_{IC}$) with α, but $KB' \cup \{\beta\}$ is inconsistent $KB_I \cup KB_{IC}$ with α.*
*(KB*7.3)* Weak relevance: *If $\beta \in KB \backslash KB * \alpha$, then there is a set KB' such that $KB' \subseteq KB \cup \alpha$, KB' is consistent ($KB_I \cup KB_{IC}$) with α, but $KB' \cup \{\beta\}$ is inconsistent $KB_I \cup KB_{IC}$ with α.*

To revise α from KB, only those informations that are relevant to α in some sense can be added (as example in the introduction illustrates). $(KB * 7.1)$ is very strong axiom allowing only minimum changes, and certain rational revision can not be carried out. So, relaxing this condition (example with more details can be found in [8]), this can be weakened to relevance. $(KB * 7.2)$ is relevance policy that still can not permit rational revisions, so we need to go next step. With $(KB * 7.3)$ the relevance axiom is further weakened and it is referred to as "core-retainment".

4 Knowledge base dynamics and abduction

We study the relationship between Horn knowledge base dynamics (discussed in the previous section) and abduction, a well-known from reasoning. This study helps to bring these two fields together, so that abductive logic grammar procedure could be used to implement revision. For this purpose, we use the concepts of generalized kernel change (revision and contraction), an extension of kernel contraction and revision introduced for belief bases. We first observe that generalized kernel change coincides with that of Horn knowledge base change (revision and contraction), and then we process to show its relationship with abduction.

4.1 Kernel revision system

To revise a Horn formula α from a Horn knowledge base KB, the idea of kernel revision is to *keep at least* one element from every inclusion-minimal subset of KB that derives α. Because of the immutable-inclusion postulate, no Horn formula from KB_I can be deleted.

Definition 5(Kernel sets). *Let a Horn knowledge base KB be a set of Horn formulae, where α is Horn clause. The α-inconsistent kernel of KB, noted by $KB\perp_{\perp}\alpha$, is the set of KB' such that:*

1. *$KB' \subseteq KB$ ensuring that $KB_I \subseteq KB'$ and $KB_{IC} \subseteq KB'$.*
2. *$KB' \cup \alpha$ is inconsistent with $KB_I \cup KB_{IC}$.*
3. *For any KB" such that $KB'' \subset KB' \subseteq KB$ then $KB'' \cup \alpha$ is consistent with $KB_I \cup KB_{IC}$.*

That is, given a consistent α, $KB\perp_{\perp}\alpha$ is the set of minimal KB-subsets inconsistent with α.

*Example 2.*Suppose that KB=$\{KB_I : p \leftarrow a \wedge b, p \leftarrow a, q \leftarrow a \wedge b;\ KB_U : a \leftarrow, b \leftarrow;\ KB_{IC} : \emptyset\}$ and $\alpha = \leftarrow p$. Then we have that:

$KB\perp_{\perp}\alpha = \{\{p \leftarrow a \wedge b\}, \{p \leftarrow a\}\}$.

Revision by a Horn clause is based on the concept of a α-inconsistent-kernels. In order to complete the construction, we must define a incision function that cuts in each inconsistent-kernel.

Definition 6(Incision function). *Let KB be a set of Horn formulae. σ is a incision function for KB if and only if, for all consistent Horn clauses α*

1.$\sigma(KB\perp_{\perp}\alpha) \subseteq \bigcup KB\perp_{\perp}\alpha$
2.If $KB' \in KB\perp_{\perp}\alpha$ *then* $KB' \cap (\sigma(KB\perp_{\perp}\alpha)) \neq 0$

Definition 7(Hitting set). *A* hitting set *H for* $KB\perp_{\perp}\alpha$ *is defined as a set s.t. (i)* $H \subseteq \bigcup(KB\perp_{\perp}\alpha)$, *(ii)* $H \cap KB_I$ *is empty and (iii)* $\forall X \in KB\perp_{\perp}\alpha$, $X \neq \emptyset$ *and* $X \cap KB_U$ *is not empty, then* $X \cap H \neq \emptyset$.

A hitting set is said to be *maximal* when H consists of all updatable statements from $\bigcup(KB\perp_{\perp}\alpha)$ and *minimal* if no proper subset of H is a hitting set for $KB\perp_{\perp}\alpha$.

Definition 8(Generalized Kernel revision). *An incision function for KB is a function s.t. for all α, $\sigma(KB\perp_{\perp}\alpha)$ is a hitting set for $KB\perp_{\perp}\alpha$. An operator $*_{\sigma}$ for KB is a generalized kernel revision defined as follows:*

$$KB *_{\sigma} \alpha = \begin{cases} (KB \backslash \sigma(KB\perp_{\perp}\alpha) \cup \alpha, & \text{if } \alpha \text{ is consistent with } KB_I \cup KB_{IC} \\ KB, & \text{otherwise.} \end{cases}$$

An operator $*_{\sigma}$ for KB is a generalized kernel revision iff there is an incision function σ for KB such that $KB * \alpha = KB *_{\sigma} \alpha$ for all beliefs α.

From the definition of hitting set, it is clear that when $KB \vdash \neg\alpha$, α is the hitting set of $KB\perp_{\perp}\alpha$. On the other hand, when $KB_I \vdash \alpha$, the definition ensures that only updatable elements are inserted, and α does follow from the revision. Thus, week success (KB*2), immutable-inclusion(KB*3.2) and vacuity (KB*4.1) are satisfied by generalized kernel revision of α from KB.

*Example 3.*Given KB=$\{KB_I : p \leftarrow a \wedge b, p \leftarrow a, q \leftarrow a \wedge b;\ KB_U : a \leftarrow, b \leftarrow;\ KB_{IC} : \emptyset\}$, $\alpha = \leftarrow p$ and $KB\perp_{\perp}\alpha = \{\{p \leftarrow a \wedge b\}, \{p \leftarrow a\}\}$. We have two possible results for the incision function and its associated kernel revision operator:

$\sigma_1(KB\perp_{\perp}\alpha) = \{p \leftarrow a \wedge b\}$ and $KB *_{\sigma_1} \alpha = \{\{\leftarrow a\}, \{\leftarrow b\}\}$,

$\sigma_2(KB\perp_{\perp}\alpha) = \{p \leftarrow a\}$ and $KB *_{\sigma_2} \alpha = \{\{\leftarrow a\}\}$.

Incision function σ_2 produces minimal hitting set for $KB\perp_{\perp}\alpha$.

Theorem 1.*For every Horn knowledge base KB, $*_{\sigma}$ is a generalized kernel revision function iff it satisfies the postulates (KB*1) to (KB*6) and (KB*7.3).*

4.2 Relationship with abduction

The relationship between Horn knowledge base dynamics and abduction was introduced by the philosopher Pierce (see [2]). We show how abduction grammar could be used to realize revision with immutability condition. A special subset of literal (atoms) of language $\mathscr{L}_{\mathscr{H}}$, *abducibles* Ab, are designated for abductive reasoning. An abductive framework $\langle P, Ab\rangle$ stands for a theory P, which is a set of Horn formulae from $\mathscr{L}_{\mathscr{H}}$, with possible hypotheses *Ab*. An abductive framework for a knowledge base $KB = KB_I \cup KB_U \cup KB_{IC}$ can be given as follows:

$P = KB_I \cup \{\alpha \leftrightarrow \beta | \alpha$ is a Horn clause in KB_U and β is an abducible from Ab that does not appear in $KB\}$.

Definition 9(Minimal abductive explanation). *Let KB be a Horn knowledge base and α an observation to be explained. Then, for a set of abducibles (KB_U), Δ is said to be an abductive explanation wrt KB_I iff $KB_I \cup \Delta \vdash \alpha$. Δ is said to be* minimal *wrt $KB_I \cup KB_{IC}$ iff no proper subset of Δ is an abductive explanation for α, i.e. $\nexists \Delta'$ s.t. $KB_I \cup \Delta' \vdash \alpha$.*

Since an incision function is adding and removing only updatable elements from each member of the kernel set, to compute a generalized revision of α from KB, we need to compute only the abduction in every α-kernel of KB. So, it is now necessary to characterize precisely the abducibles present in every α-kernel of KB. The notion of minimal abductive explanation is not enough to capture this, and we introduce locally minimal and KB-closed abductive explanations.

Definition 10(Local minimal abductive explanations). *Let $(KB_I \cup KB'_U)$ be a smallest subset of KB_U, s.t Δ an minimal abductive explanation of α wrt $(KB_I \cup KB'_U)$ (for some Δ). Then Δ is called local minimal for α wrt KB_U.*

*Note 1.*Let $(KB_I \cup KB_U) \in (\{\Delta^+, \Delta^-\})$. Here Δ^+ refers to admission Horn knowledge base (positive atoms) and Δ^- refers to denial Horn knowledge base(negative atoms) wrt given α. Then problem of abduction is to explain Δ with abducibles (KB_U), s.t. $(KB_I \cup KB_U) \cup \Delta^+ \cup \Delta^- \vdash \alpha$ and $(KB_I \cup KB_U) \cup \Delta^+ \models \alpha \cup \Delta^-$ are both consistent with IC.

4.3 Generalized revision algorithm

The problem of Horn knowledge base revision is concerned with determining how a request to change can be appropriately translated into one or more atoms or

literals. We give new generalized revision algorithm. It is enough to compute all the KB-locally minimal abduction explanations for α wrt $KB_I \cup KB_U \cup KB_{IC}$. If α is consistent with KB then well-known abductive procedure to compute an abductive explanation for α wrt KB_I could be used to compute kernel revision

Reasoning about Abduction and Deduction

Definition 11([51]). *Let KB=(KB_I, KB_U, KB_{IC}) be a knowledge base, T is updatable part from KB. We define abduction framework $\langle KB^{BG}, KB^{Ab}, IC\rangle$. After Algorithm 1 is executed, u is derived part from KB'. The abduction explanation for u in $\langle KB_I \cup KB_U^*, KB_{IC}\rangle$ is any set T_i, where $T_i \subseteq KB^{Ab}$ such that: $KB_I \cup KB_U^* \cup T \models u$.*

An explanation T_i is minimal if no proper subset of T_i is also an explanation, i.e. if it does not exist any explanation T_j for u such that $T_j \subset T_i$

Definition 12([51]). *Let KB=(KB_I, KB_U, KB_{IC}) be a knowledge base, T is updatable part from KB. After Algorithm 1 is executed, u is derived part from KB'. The deduction consequence on u due to the application of T, $KB_I \cup KB_U^* \cup T \cup u$ is the answer to any question.*

Theorem 2.*Let KB be a Horn knowledge base and α is Horn formula.*

1. *If Algorithm 1 produced KB'as a result of revising α from KB, then KB' satisfies all the rationality postulates (KB*1) to (KB*6) and (KB*7.3).*
2. *Suppose KB'' satisfies all these rationality postulates for revising α from KB, then KB'' can be produced by Algorithm 1.*

5 Application: View updates in database

An important application of knowledge base dynamics, discussed in the previous section, is in providing an axiomatic characterization of view updates in deductive and relational databases. A *definite deductive database* DDB consists of two parts: an *intensional database* IDB (KB_I), a set of definite program clauses; and an *extensional database* EDB (KB_U), a set of ground facts. The intuitive meaning of DDB is provided by the *Least Herbrand model semantics* and all the inferences are carried out through *SLD-derivation. All the predicates that are defined in IDB are referred to as view predicates*and those defined in EDB are referred to as *base predicates.* Extending this notion, an atom(literals) with a view predicate is said to be a *view atom(literals),* and similarly an atom(literals) with base predicate is a *base atom(literals).* Further, we assume that IDB does not contain any unit clauses and that predicates defined in a given DDB are both view and base predicates.

Two kinds of view updates can be carried out on a DDB: An atom(literals), that does not currently follow from DDB, can be *inserted;* or an atom(literals), that currently follows from DDB, can be *deleted [7,8].* In this paper, we consider only insertion an atom(literals) from a DDB. When an atom(literals) *A* is to be inserted, the view update problem is to delete only some relevant EDB facts and then to insert, so that the modified EDB together with IDB will satisfy the insertion of *A* from DDB. As motivated in the introduction, our concern now is to discuss the rationality of view update, and provide an axiomatic characterization for it. This axiomatic characterization can be seen as a declarative semantics for view updates in deductive databases.

Note that DDB can be considered [37,46] as a knowledge base to be revised. The IDB is the immutable part of the knowledge database, while the EDB forms the updatable part. Every base literal is an abducible, but since we deal only with definite databases, we require only positive abducibles. In general, it is assumed that a language underlying a DDB is fixed and the semantics of DDB is the least Herbrand model over this fixed language. Therefore, the DDB is practically a shorthand of its ground instantiation[1], written as IDB_G. Thus, a DDB represent a knowledge base where the immutable part is given by IDB_G and updatable part is EDB. Hence, the rationality postulates (KB*1) to (KB*6) and (KB*7.3) provide an axiomatic characterization for inserting a view atom(literals) *A* to a definite database DDB, and a generalized insertion of *A* to DDB achieves deletion of *A* from DDB.

As observed by Kowalski [26], logic can provide a conceptual level of understanding of relational databases, and hence rationality postulates (KB*1) to (KB*6) and (KB*7.3)can provide an axiomatic characterization for view insertion in relational databases too. A relational database together with its view definitions can be represented by a definite deductive database (EDB representing tuples in the database and IDB representing the view definitions), and so same algorithm can be used to insert view extensions from relational and deductive databases.

But before discussing the rationality postulates and algorithm, we want to make it precise, how a relational database, along with operations on relations, can be represented by definite deductive database. We assume the reader is familiar with relational database concepts. *A relation scheme R* can be thought of as a base predicate whose arguments define the *attributes* **A** of the scheme. Its *relational extension r*, is a finite set of base atoms $R(\mathbb{A})$ containing the predicate *R*. A *database schema*

[1] a ground instantiation of a definite program *P* is the set of clauses obtained by substituting terms in the Herbrand Universe for variables in *P* in all possible ways

Algorithm 1 Generalized revision algorithm

Input : A Horn knowledge base $KB = KB_I \cup KB_U \cup KB_{IC}$ and a Horn clause α to be revised.

Output: A new Horn knowledge base $KB' = KB_I \cup KB_U^* \cup KB_{IC}$, s.t. $KB'$is a generalized revision α to KB.

Procedure $KB(KB, \alpha)$

begin

1. Let V:= $\{c \in KB_{IC} \mid KB_I \cup KB_{IC}$ inconsistent with α wrt $c\}$
   $P := N := 0$ and $KB' = KB$
2. While $(V \neq 0)$
   select a subset $V' \subseteq V$
   For each $v \in V'$, select a literal to be
   remove (add to N) or a literal to be added(add to P)
   Let KB $:= KR(KB, P, N)$
   Let V:= $\{c \in KB_{IC} \mid KB_I$ inconsistent with α wrt $c\}$
   return
3. Produce a new Horn knowledge base KB'

end.

Algorithm 2

Procedure $KR(KB, \Delta^+, \Delta^-)$

begin

1. Let $P := \{e \in \Delta^+ \mid KB_I \not\models e\}$ and $N := \{e \in \Delta^- \mid KB_I \models e\}$
2. While $(P \neq 0)$ or $(N \neq 0)$
   select a subset $P' \subseteq P$ *or* $N' \subseteq N$
   Construct a set $S_1 = \{X \mid X$ is a KB-closed locally minimal abductive wrt P explanation for α wrt $KB_I\}$.
   Construct a set $S_2 = \{X \mid X$ is a KB-closed locally minimal abductive wrt N explanation for α wrt $KB_I\}$.
3. Determine a hitting set $\sigma(S_1)$ and $\sigma(S_2)$
   If $((N = 0)$ *and* $(P \neq 0))$
   Produce $KB' = KB_I \cup \{(KB_U \cup \sigma(S_1)\}$
   else
   Produce $KB' = KB_I \cup \{(KB_U \backslash \sigma(S_2) \cup \sigma(S_1)\}$
   end if
   If $((N \neq 0)$ and $(P = 0))$
   Produce $KB' = KB_I \cup \{(KB_U \backslash \sigma(S_2)\}$
   else
   Produce $KB' = KB_I \cup \{(KB_U \backslash \sigma(S_2) \cup \sigma(S_1)\}$
   end if
4. return KB'

end.

consists of finite collection of relational schemes $< R_1, \ldots, R_n >$, and a *relational database* is a specific extension of database schema, denoted as $< r_1, \ldots, r_n >$. In our context, relational database can be represented by $EDB = \bigcup_{i=1,\ldots,n} R_i(\mathbb{A}_i)$.

Join is a binary operator for combining two relations. Let r and s be two relational extensions of schema R (with attributes $\mathbb{R}$) and S (with attributes $\mathbb{S}$), respectively. Let $\mathbb{T} = \mathbb{R} \cup \mathbb{S}$. The join of r and s, written as $r \otimes s$, is the relational extension $q(\mathbb{T})$ of all tuples t over $\mathbb{T}$ such that there are $t_r \in r$ and $t_s \in s$, with $t_r = t(\mathbb{R})$ and $t_s = t(\mathbb{S})$. Join can be captured by a constraint clause $Q(\mathbb{T}) \leftarrow R(\mathbb{R}), S(\mathbb{S})$. Our integrity constraint (IC) is that each research group has only one chair i.e. $\forall x, y, z$ (y=x) $\leftarrow$ group_chair(x,y) $\wedge$ group_chair(x,z) (see definition and properties of similarity in works of Christiansen [11] and Godfrey [19]).

*Example 4.*Let us consider two relational schemes R and S from Example 1, with attributes $R = \{Group, Chair\}$ and $S = \{Staff, Group\}$.Consider the following extensions r and s:

s	Staff	Group
	delhibabu	infor1
	aravindan	infor2

r	Group	Chair
	infor1	matthias
	infor2	gerhard

Tab. 1. Base table for s and r

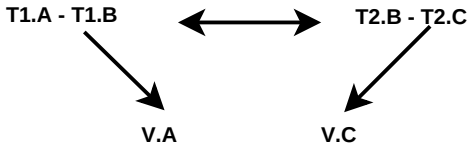

Fig 1. View Atom

The following rule, $T(Staff, Group, Chair) \leftarrow S(Staff, Group), R(Group, Chair)$

represents the join of s and r, which is given as:

$s \otimes r$	$Staff$	$Group$	$Chair$
	delhibabu	infor1	matthias
	aravindan	infor2	gerhard

Tab. 2. $s \otimes r$

To sum up, we showed how relational database and operators on relations can be conceptually captured by definite deductive databases. All solutions translate [38] a view update request into a **transaction combining insertions and deletions of base relations** for satisfying the request. Further, a definite deductive database can be considered as a knowledge base, and thus rationality postulates and insertion algorithm of the previous section can be applied for view updates in database.

5.1 View insertion algorithm

Since relational and definite deductive databases can be considered as knowledge bases, and inserting a view atom(literals) (tuple) A can be considered as revision of A, a specific instance of Algorithm 1 can be used to compute insertion of a view atom(literals) to a database. In fact, we have to discuss how to compute all DDB-closed locally minimal abductive explanations for A wrt IDB_G. As expected, these abductive explanations can be computed using deduction trees, and the process is discussed in the sequel.

An update request U = B, where B is a set of base facts, is not true in KB. Then, we need to find a transaction $T = T_{ins} \cup T_{del}$, where $T_{ins}(\Delta_i)$ (resp. $T_{del}(\Delta_j)$) is the set of facts, such that U is true in $DDB' = ((EDB - T_{del} \cup T_{ins}) \cup IDB \cup IC)$. Since we consider definite deductive databases, SLD-tree can be used to compute the required abductive explanations. The idea is to get all EDB facts used in a SLD-derivation of A wrt DDB, and construct that as an abductive explanation for A wrt IDB_G.

There are two ways to find minimal elements (insertion and deletion) with integrity constraints. Algorithm 3 first checks consistency with integrity constraints and then reduces steps with abductive explanation for A . Algorithm 4 is doing *vice versa*, but both algorithm outputs are similar.

Unfortunately, this algorithm does not work as intended for any deductive database, and a counter example is produced below. Thus, general algorithms 3 and 4 produced some unexpected sets in addition to locally minimal abductive explanations.

*Example 5.*Consider a deductive database DDB as follows:

$$IDB: \begin{array}{l} p \leftarrow a \wedge e \\ q \leftarrow a \wedge f \\ p \leftarrow b \wedge f \\ q \leftarrow b \wedge e \\ p \leftarrow q \\ q \leftarrow a \end{array} \qquad EDB: \begin{array}{l} a \leftarrow \\ e \leftarrow \\ f \leftarrow \end{array} \qquad IC: \leftarrow b$$

We need to insert p. First, we check consistency with IC and after we find Δ_i and Δ_j via tree deduction.

$[\leftarrow a, e$
■ $[.\leftarrow q \leftarrow a, f$
■ $\leftarrow a$
■ $\leftarrow b, e$
□ $].\leftarrow q \leftarrow b, f$
□ $].\leftarrow p$

From Algorithm 3 it is easy to conclude which branches are consistent wrt IC (shown on tree by ■). For the next step, we need to find minimal accommodate and denial literal with wrt to p. The subgoals of the tree are $\leftarrow a, e$ and $\leftarrow a, f$, which are minimal tree deductions of only facts. Clearly, $\Delta_i = \{a, e, f\}$ and $\Delta_j = \{b\}$ with respect to IC, are the only locally minimal abductive explanations for p wrt IDB_G, but they are not locally minimal explanations.

From Algorithm 4, the subgoals of the tree are $\leftarrow a, e$, $\leftarrow a, f$, $\leftarrow b, f$ and $\leftarrow b, e$. Clearly, $\Delta_i = \{a, b, e, f\}$ and $\Delta_j = \{a, e, f\}$. In the next step, we check consistency with IC. Δ_i and Δ_j are only locally minimal abductive explanations for p wrt IDB_G, but they are not locally minimal explanations (more explanations can be found in [33]).

The program is clear due to the unwanted recursion $p \leftarrow a \wedge b, p \leftarrow a$. Will the algorithm work as intended if we restrict ourselves to acyclic program [8] that excludes such loop? One would expect a positive answer, but

Algorithm 3	Algorithm to compute all DDB-closed locally minimal abductive explanation of an atom(literals)
Input :	A definite deductive database $DDB = IDB \cup EDB \cup IC$ an literals $\mathscr{A}$
Output :	Set of all DDB-closed locally minimal abductive explanations for $\mathscr{A}$ wrt IDB_G
begin	
1.	Let $V := \{c \in IC \mid IDB \cup IC$ inconsistent with $\mathscr{A}$ wrt $c\}$ While $(V \neq 0)$ Construct a complete SLD-tree for $\leftarrow \mathscr{A}$ wrt DDB. For every successful branch i: construct $\Delta_i = \{D \mid D \in EDB$ and D is used as an input clause in branch $i\}$ For every unsuccessful branch j: construct $\Delta_j = \{D \mid D \in EDB$ and D is used as an input clause in branch $j\}$ Produce set of all Δ_i and Δ_j computed in the previous step as the result. return
2.	Produce all DDB-closed locally minimal abductive explanations in Δ_i and Δ_j
end.	

Algorithm 4	Algorithm to compute all DDB-closed locally minimal abductive explanation of an atom(literals)
Input :	A definite deductive database $DDB = IDB \cup EDB \cup IC$ an literals $\mathscr{A}$
Output :	Set of all DDB-closed locally minimal abductive explanations for $\mathscr{A}$ wrt IDB_G
begin	
1.	Construct a complete SLD-tree for $\leftarrow \mathscr{A}$ wrt DDB. For every successful branch i: construct $\Delta_i = \{D \mid D \in EDB$ and D is used as an input clause in branch $i\}$ For every unsuccessful branch j: construct $\Delta_j = \{D \mid D \in EDB$ and D is used as an input clause in branch $j\}$
2.	Let $V := \{c \in IC \mid IDB \cup IC$ inconsistent with $\mathscr{A}$ wrt $c\}$ While $(V \neq 0)$ Produce set of all Δ_i and Δ_j is consistent with IC as the result. return Produce all DDB-closed locally minimal abductive explanations in Δ_i and Δ_j
end.	

unfortunately still some unwanted sets may be produced as the following example highlights.

So, even for acyclic program, algorithms 3 and 4 do not work as intended (that is to generate all and only the DDB-close locally minimal abductive explanations). Does this mean that generalized revision can not be carried out for database in general?. Probably we should approach the problem from different perspective. We have seen that algorithms 3 and 4 may compute some unwanted sets in addition to the required ones. What exactly are those sets? Is it possible to characterize them? The following lemma answers these questions.

Lemma 1.*Let $DDB = IDB \cup EDB \cup IC$ be a definite deductive database and A an atom(literals). Let S be the set of all DDB-closed locally minimal abductive explanations for A wrt IDB_G. Let S' be the set of explanations returned by algorithms 3 and 4 given DDB and A as inputs. Then, the following propositions hold:*

1.$S \subseteq S'$.
2.$\forall \Delta'(\Delta' \in \Delta_i \cup \Delta_j) \in S'$: $\exists \Delta \in S$ s.t. $\Delta \subset \Delta'$.
3.Suppose DDB is resticted to be acyclic then: $\forall \Delta' \in S'$: $\Delta' \subset \bigcup S$.

Having characterized what exactly is computed by algorithms 3 and 4, we now proceed to show that algorithms 5 and 6 are useful for view insertion. The key to the solution is the following lemma, which established the preservable of hitting set computation among two sets.

Lemma 2. *1.Let S be a set of sets, and S' another set s.t. $S \subseteq S'$ and every member of $S' \backslash S$ contains an element*

of S. Then, a set H is minimal hitting set for S iff it is a minimal hitting set for S′.

2. *Let S be a set of sets, and S′ another set s.t. S ⊆ S′ and for every member X of S′\S: X contains a member of S and X is contained in ⋃S. Then, a set H is a hitting set for S iff it is a hitting set for S′.*

Thus algorithms 3 and 4 in conjunction with an algorithm to compute minimal hitting set can be used to compute partial meet revision (defined in section 4.1) of A from DDB.

When DDB is acyclic, generalized revision of A from DDB can be obtained by Algorithm 6. Observe that the first two steps of Algorithm 5 are same as those of algorithms 3 and 4, and we have already established what exactly are computed by them. Steps 3 and 4 clearly compute a minimal hitting set and as established by lemma 1 and lemma 2, this algorithm produces a partial meet contraction of A from DDB. This result is formalized below.

Theorem 3.*Let DDB be a definite deductive database and A an atom(literals) to be inserted. Then DDB' is a result of algorithm 5 given DDB and A as inputs, iff DDB' is a partial meet revision of A from DDB, satisfying the postulates (KB*1) to (KB*6) and (KB*7.1).*

We proceed to present Algorithm 6 to compute generalized revision for definite deductive database. As observed before, this is not possible in general, but for a restricted case of acyclic program.

Theorem 4.*Let DDB be a definite deductive database and A an atom(literals) to be inserted. Then DDB' is a result of algorithm 6 given DDB and A as inputs, iff DDB' is a generalized revision of A from DDB, satisfying the postulates (KB*1) to (KB*6) and (KB*7.3).*

Algorithms 5 and 6 are inefficient, as they need to build a complete SLD-tree. Unfortunately, any rational algorithm for insertion can not avoid constructing complete SLD-trees. If these algorithms are changed to extract input clauses from incomplete SLD-derivation, then the new algorithm should check the derivability of an atom(literals) from a deductive database, before any insertion is carried out(otherwise, success can not be satisfied). Checking derivability is also computationally expensive and more then that, weak relevance policy (KB*7.3) will not be satisfied in general. Finally, any rational algorithm must construct a complete SLD-tree.

5.2 Incomplete to Complete Information

Many of the proposals in the literature on incomplete databases have focussed on the extension of the relational model by the introduction of null values. In this section, we show how view update provides completion of incomplete information. More detailed surveys of this area can be found in [36].

The earliest extension of the relational model to incomplete information was that of Codd [13] who suggested that missing values should be represented in tables by placing a special *null value* symbol $'*'$ at any table location for which the value is unknown. Table 3, shows an example of a database using this convention. Codd proposed an extension to the relational algebra for tables containing such nulls, based on three valued logic and a null substitution principle.

In terms of our general semantic scheme, the intended semantics of a database D consisting of Codd tables can be described by defining $Mod(D)$ to be the set of structures $M_{D'}$, where D' ranges over the relational databases obtained by replacing each occurrence of $'*'$ in the database D by some domain value. Different values may be substituted for different occurrences.

A plausible integrity constraint on the meaning of a relational operator on tables in $\mathscr{T}$ is that the result should be a table that represents the set of relations obtained by pointwise application of the operator on the models of these tables. For example, if R and S are tables in $\mathscr{T}$ then the result of the join $R \bowtie S$ should be equal to a table T in $\mathscr{T}$ such that

$$Mod(T) = \{r \bowtie t \mid r \in Mod(R),\ s \in Mod(S)\}$$

In case the definitions of the operators satisfy this integrity constraint (with respect to the definition of the semantics *Mod* on $\mathscr{T}$).

Let us consider what above equation requires if we take R and S to be the Codd Tables 3. First of all, note that in each model, if we take the value of the null in the tuple (delhibabu,*) to be v, then the join will contain one tuples (delhibabu, v), which include the value v. If T is to be a Codd table, it will need to contain tuples (delhibabu,X) to generate each of these tuples, where X are either constants or '*'. We now face a problem. First, X cannot be a constant c, for whatever the choice of c we can find an instance $r \in Mod(R)$ and $s \in Mod(S)$ for which the tuple (delhibabu, c) does not occur in $r \bowtie s$. If they were, X would have their values in models of T assigned independently.

Here the repetition of $*$ indicates that the *same* value is to be occurrence of the null in constructing a model of the table. Unfortunately, this extension does not suffice to satisfy the integrity constraint $(\forall x,y,z$ (y=x) $\leftarrow$ group_chair(x,y) $\wedge$ group_chair(x,z)).

Staff	Group
delhibabu	infor1
delhibabu	*

Group	Chair
infor1	mattias
*	aravindan

Tab. 3. Base Table after Transaction

Algorithm 5	Partial meet revision for definite deductive database
Input :	A definite deductive database $DDB = IDB \cup EDB \cup IC$ an literals $\mathscr{A}$
Output:	A Partial meet revision of $\mathscr{A}$ from DDB.
begin	
1.	Let $V := \{c \in IC \mid IDB \cup IC$ inconsistent with $\mathscr{A}$ wrt $c\}$
	While $(V \neq 0)$
2.	Construct a complete SLD-tree for $\leftarrow \mathscr{A}$ wrt DDB.
3.	For every successful branch i:construct $\Delta_i = \{D \mid D \in EDB\}$
	and D is used as an input clause in branch i.
	Let there be m such sets.
	Let $E^* = \{\{D_1, \ldots, D_m\} \mid D_i \in \Delta_i\}$
	Let E be a inclusion-minimal set among $E*$, i.e. $\nexists E' \in E^*$
	s.t. $E' \subset E$.
4.	For every unsuccessful branch j:construct $\Delta_j = \{D \mid D \in EDB\}$
	and D is used as an input clause in branch j.
	Let there be m such sets.
	Let $F^* = \{\{D_1, \ldots, D_m\} \mid D_j \in \Delta_j\}$
	Let F be a inclusion-maximum set among $F*$, i.e. $\nexists F' \in F^*$
	s.t. $F' \subseteq F$.
	Let $V := \{c \in IC \mid IDB \cup IC$ inconsistent with $\mathscr{A}$ wrt $c\}$
	return
5.	Produce $DDB \backslash F \cup E$ as the result.
end.	

Algorithm 6	Generalized revision for acyclic definite deductive database
Input :	An acyclic definite deductive database $DDB = IDB \cup EDB \cup IC$ an literals $\mathscr{A}$
Output:	A generalized revision of $\mathscr{A}$ from DDB.
begin	
1.	Let $V := \{c \in IC \mid IDB \cup IC$ inconsistent with $\mathscr{A}$ wrt $c\}$
	While $(V \neq 0)$
2.	Construct a complete SLD-tree for $\leftarrow \mathscr{A}$ wrt DDB.
3.	For every successful branch i:construct $\Delta_i = \{D \mid D \in EDB\}$
	and D is used as an input clause in branch i.
	Construct a hitting set D for all Δ_i's computed in the previous step.
4.	For every unsuccessful branch j:construct $\Delta_j = \{D \mid D \in EDB\}$
	and D is used as an input clause in branch j.
	Construct a hitting set D for all Δ_j's computed in the previous step.
	Let $V := \{c \in IC \mid IDB \cup IC$ inconsistent with $\mathscr{A}$ wrt $c\}$
	return
5.	Produce $DDB \backslash F \cup E$ as the result.
end.	

In the model of these tables in which $* = infor1$, the join contains the tuple (delhibabu, infor1) and (infor1, aravindan).

If $*_1 =$ infor1 then (delhibabu, infor1) $\in R \bowtie S$

If $*_2 =$ infor1 then (infor1, aravindan) $\in R \bowtie S$

The following table shows when transaction is made to base table:

Staff	Group	Chair
delhibabu	infor1	mattias
delhibabu	*	aravindan

Tab. 4. $s \otimes r$ after Transaction

The following table shows completion of incomplete information with application of integrity constraint and redundancy:

Staff	Group	Chair
delhibabu	infor1	aravindan

Tab. 5. Redundant Table

6 Related Works

We begin by recalling previous work on view deletion. Chandrabose [7,8], defines a contraction operator in view deletion with respect to a set of formulae or sentences using Hansson's [21] belief change. Similar to our approach, he focused on set of formulae or sentences in knowledge base revision for view update wrt. insertion and deletion and formulae are considered at the same level. Chandrabose proposed different ways to change knowledge base via only database deletion, devising particular postulate which is shown to be necessary and sufficient for such an update process.

Our Horn knowledge base consists of two parts, immutable part and updatable part , but focus is on principle of minimal change. There are more related works on that topic. Eiter [16], Langlois[28], and Delgrande [15] are focusing on Horn revision with different perspectives like prime implication, logical closure and belief level. Segerberg [45] defined new modeling for belief revision in terms of irrevocability on prioritized revision. Hansson [21], constructed five types of non-prioritized belief revision. Makinson [34] developed dialogue form of revision AGM. Papini[42] defined a new version of knowledge base revision. Here, we consider immutable part as a Horn clause and updatable part as an atom(literals).

We are bridging gap between philosophical work, paying little attention to computational aspects of database work. In such a case, Hansson's[21] kernel change is related with abductive method. Aliseda's [2] book on abductive reasoning is one of the motivation keys. Christiansen's [12] work on dynamics of abductive logic grammars exactly fits our minimal change (insertion and deletion). Wrobel's [48] definition of first order theory revision was helpful to frame our algorithm.

On other hand, we are dealing with view update problem. Keller's [23] thesis is motivation for view update problem. There is a lot of papers on view update problem (for example, recent survey paper on view update by Chen and Liao[10] and survey paper on view algorithm by Mayol and Teniente [35]. More similar to our work is paper presented by Bessant et al. [4] , local search-based heuristic technique that empirically proves to be often viable, even in the context of very large propositional applications. Laurent et al.[29], parented updating deductive databases in which every insertion or deletion of a fact can be performed in a deterministic way.

Furthermore, and at a first sight more related to our work, some work has been done on ontology systems and description logics (Qi and Yang [43], and Kogalovsky [24]). Finally, when we presented connection between belief update versus database update, we did not talk about complexity (see the works of Liberatore [30,31], Caroprese [6], Calvanese's [9], and Cong [14]).

The significance of our work can be summarized in the following:

-We have defined new kind of revision operator on knowledge base and obtained axiomatic characterization for it. This operator of change is based on α consistent-remainder set. Thus, we have presented a way to construct revision operator without need to make use of the generalized Levi's identity nor of a previously defined contraction operator.
-We have defined new way of insertion and deletion of an atom(literals) as per norm of principle of minimal change.
-We have proposed new generalized revision algorithm for knowledge base dynamics, interesting connections with kernel change and abduction procedure.
-We have written new view insertion algorithm for DDB, and we provided Horn knowledge base revision, using our axiomatic method.
-Finally, we shown connection between belief update versus database update.

7 Conclusion and remarks

The main contribution of this research is to provide a link between theory of belief dynamics and concrete applications such as view updates in databases. We argued for generalization of belief dynamics theory in two respects: to handle certain part of knowledge as immutable; and dropping the requirement that belief state be deductively closed. The intended generalization was achieved by introducing the concept of knowledge base dynamics and generalized contraction for the same. Further, we also studied the relationship between knowledge base dynamics and abduction resulting in a generalized algorithm for revision based on abductive procedures. We also successfully demonstrated how knowledge base dynamics can provide an axiomatic characterization for insertion an atom(literals) to a definite deductive database. Finally, we give a quick overview of the main operators for belief change, in particular, belief update versus database update.

In bridging the gap between belief dynamics and view updates, we have observed that a balance has to be achieved between computational efficiency and rationality. While rationally attractive notions of generalized revision prove to be computationally inefficient, the rationality behind efficient algorithms based on incomplete trees is not clear at all. From the belief dynamics point of view, we may have to sacrifice some postulates, vacuity for example, to gain computational efficiency. Further weakening of relevance has to be explored, to provide declarative semantics for algorithms based on incomplete trees.

On the other hand, from the database side, we should explore various ways of optimizing the algorithms that would comply with the proposed declarative semantics. We believe that partial deduction and loop detection techniques, will play an important role in optimizing algorithms of the previous section. Note that, loop

detection could be carried out during partial deduction, and complete SLD-trees can be effectively constructed wrt a partial deduction (with loop check) of a database, rather than wrt database itself. Moreover, we would anyway need a partial deduction for optimization of query evaluation.

Though we have discussed only about view updates, we believe that knowledge base dynamics can also be applied to other applications such as view maintenance, diagnosis, and we plan to explore it further (see works [6] and [5]). It would also be interesting to study how results using soft stratification [3] with belief dynamics, especially the relational approach, could be applied in real world problems. Still, a lot of developments are possible, for improving existing operators or for defining new classes of change operators. As immediate extension, question raises: is there any *real life application for AGM in 25 year theory?* [18]. The revision and update are more challenging in logical view update problem(database theory), so we can extend the theory to combine results similar to Hansson's [20], Konieczny's [25]and Nayak, [40].

Appendix

Proof of Theorem 1. (**If part**) * satisfies (KB*1) to (KB*6) and (KB*7.3). We must show that $*$ is a generalized kernel revision. Let σ be a incision function such that for α. When $KB_I \vdash \alpha$, (KB*1) to (KB*6) and (KB*7.3) imply that $KB * \alpha = KB$ coincides with generalized revision.

When $KB_I \vdash \neg\alpha$, the required result follows from the two observations:

1. $\exists KB' \in KB\perp_\perp \alpha$ s.t. $KB * \alpha \subseteq KB'$ (when $KB_I \vdash \alpha$)
   Let σ be an incision function for KB and $*_\sigma$ be the generalized revision on KB that is generated by σ. Since * satisfies closure (KB*1), $KB *_\sigma \alpha$ is KB contained in α. Also, satisfaction of weak success postulate (KB*2) ensures that $\alpha \subseteq KB *_\sigma \alpha$. Every element of $KB\perp_\perp \alpha$ is a inclusion minimal subset that does derive α, and so any subset of KB that does derive α must be contained in a member of $KB\perp_\perp \alpha$.
2. $\bigcap(KB\perp_\perp \alpha) \subseteq KB *_\sigma \alpha$ (when $KB_I \vdash \alpha$)
   Consider any $\beta \in \bigcap(KB\perp_\perp \alpha)$. Assume that $\beta \notin KB * \alpha$. Since * satisfies weak relevance postulate (KB*7.3), it follows that there exists a set KB' s.t. $KB' \subseteq KB \cup \alpha$; KB' is a consistent with α; and $KB' \cup \{\beta\}$ is inconsistent with α. But this contradicts that β is present in every minimal subset of KB that does derive α. Hence β must not be in $KB *_\sigma \alpha$.

(**Only if part**) Let $KB * \alpha$ be a generalized revision of α for KB. We have to show that $KB * \alpha$ satisfies the postulate (KB*1) to (KB*6) and (KB*7.3).

Let σ be an incision function for KB and $*_\sigma$ be the generalized revision on KB that is generated by σ.

Closure. Since $KB *_\sigma \alpha$ is a Horn knowledge base, this postulate is trivially shown.

Weak Success. Suppose that α is consistent. Then it is trivial by definition that $\alpha \subseteq KB *_\sigma \alpha$.

Inclusion. Trivial by definition.

Immutable-inclusion. Since every $X \in KB\perp_\perp \alpha$ is such that $X \subseteq KB_I$ then this postulate is trivially shown.

Vacuity 1. Trivial by definition.

Vacuity 2. If $KB \cup \{\alpha\}$ is consistent then $KB\perp_\perp \alpha = \{\{\text{KB}\}\}$. Hence $KB *_\sigma \alpha = KB \cup \{\alpha\}$.

Consistency. Suppose that α is consistent. Then $KB\perp_\perp \alpha \neq= \emptyset$ and by definition, every $X \in KB\perp_\perp \alpha$ is consistent with α. Therefore, the intersection of any subset of $KB\perp_\perp \alpha$ is consistent with α. Finally, $KB *_\sigma \alpha$ is consistent.

Uniformity. If α and β are KB-equivalent, then $KB\perp_\perp \alpha = KB\perp_\perp \beta$

Weak relevance. Let $\beta \in KB$ and $\beta \notin KB *_\sigma \alpha$. Then $KB *_\sigma \alpha \neq KB$ and, from the definition of $*_\sigma$,it follows that:

$$KB *_\sigma \alpha = (KB \backslash \sigma(KB\perp_\perp \alpha)) \cup \alpha$$

Therefore, from $\beta \notin (KB \backslash \sigma(KB\perp_\perp \alpha)) \cup \alpha$ and $\beta \in KB$, we can conclude that $\beta \in \sigma(KB\perp_\perp \alpha)$. By definition $\sigma(KB\perp_\perp \alpha) \subseteq \bigcup KB\perp_\perp \alpha$, and it follows that there is some $X \in KB\perp_\perp \alpha$ such that $\beta \in X$. X is a minimal KB-subset inconsistent with α. Let $Y = X \backslash \{\beta\}$. Then Y is such that $Y \subset X \subseteq KB \subseteq KB \cup \alpha$. Y is consistent with α but $Y \cup \{\beta\}$ is consistent with α. ■

Proof of Theorem 2. Follows from Theorem 1 and Definition 10. ■

Proof of Lemma 1.

1. Consider a $\Delta(\Delta \in \Delta_i \cup \Delta_j) \in S$. We need to show that Δ is generated by algorithm 3 at step 2. From lemma 1, it is clear that there exists a A-kernel X of DDB_G s.t. $X \cap EDB = \Delta_j$ and $X \cup EDB = \Delta_i$. Since $X \vdash A$, there must exist a successful derivation for A using only the elements of X as input clauses and similarly $X \nvdash A$. Consequently Δ must have been constructed at step 2.
2. Consider a $\Delta'((\Delta' \in \Delta_i \cup \Delta_j) \in S'$. Let Δ' be constructed from a successful(unsuccessful) branch i via $\Delta_i(\Delta_j)$. Let X be the set of all input clauses used in the refutation i. Clearly $X \vdash A(X \nvdash A)$. Further, there exists a minimal (wrt set-inclusion) subset Y of X that derives A (i.e. no proper subset of Y derives A). Let $\Delta = Y \cap EDB$ ($Y \cup EDB$). Since IDB does not(does) have any unit clauses, Y must contain some EDB facts, and so Δ is not empty (empty) and obviously $\Delta \subseteq \Delta'$. But, Y need not (need) be a A-kernel for IDB_G since Y is not ground in general. But it stands for several A-kernels with the same (different) EDB facts Δ in them. Thus, from lemma 1, Δ is a DDB-closed locally minimal abductive explanation for A wrt IDB_G and is contained in Δ'.

3.Since this proof requires some details of acyclic programs that are not directly related to our discussion here, it is relegated [9].

Proof of Lemma 2.

1.(**Only if part**) Suppose H is a minimal hitting set for S. Since $S \subseteq S'$, it follows that $H \subseteq \bigcup S'$. Further, H hits every element of S' , which is evident from the fact that every element of S' contains an element of S. Hence H is a hitting set for S' . By the same arguments, it is not difficult to see that H is minimal for S' too.

(**If part**) Given that H is a minimal hitting set for S' , we have to show that it is a minimal hitting set for S too. Assume that there is an element $E \in H$ that is not in $\bigcup S$. This means that E is selected from some $Y \in S' \backslash S$. But Y contains an element of S, say X. Since X is also a member of S' , one member of X must appear in H. This implies that two elements have been selected from Y and hence H is not minimal. This is a contradiction and hence $H \subseteq \bigcup S$. Since $S \subseteq S'$, it is clear that H hits every element in S, and so H is a hitting set for S. It remains to be shown that H is minimal. Assume the contrary, that a proper subset H' of H is a hitting set for S. Then from the proof of the only if part, it follows that H' is a hitting set for S' too, and contradicts the fact that H is a minimal hitting set for S' . Hence, H must be a minimal hitting set for S.

2.(**If part**) Given that H is a hitting set for S' , we have to show that it is a hitting set for S too. First of all, observe that $\bigcup S = \bigcup S'$, and so $H \subseteq \bigcup S$. Moreover, by definition, for every non-empty member X of S' , $H \cap X$ is not empty. Since $S \subseteq S'$, it follows that H is a hitting set for S too.

(**Only if part**) Suppose H is a hitting set for S. As observed above, $H \subseteq \bigcup S'$. By definition, for every non-empty member $X \in S$, $X \cap H$ is not empty. Since every member of S' contains a member of S, it is clear that H hits every member of S' , and hence a hitting set for S' . ■

Proof of Theorem 3. From Lemma 1, it is clear that step 1 and 2 generate all DDB-closed locally minimum abductive explanation for $\mathscr{A}$ wrt IDB_G and some additional sets that contain a DDB-closed local minimal abductive explanation for $\mathscr{A}$ wrt IDB_G. Step 3 and 4 clearly computes an inclusion-minimal hitting set for this. This required now following Lemma 2 and Theorem 2. ■

Proof of Theorem 4. Follows from Lemma 1, Lemma 2, and Theorem 2. ■

Acknowledgement

The first author acknowledges the support of RWTH Aachen, where he is visiting scholar with an Erasmus Mundus External Cooperation Window India4EU by the European Commission when the paper was written.

References

[1] Alchourron, C.E., et al.(1985). On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic 50*, 510 - 530.

[2] Aliseda, A. (2006). *Abductive Resoning Logic Investigations into Discovery and Explanation*. Springer book series Vol. 330.

[3] Behrend, A.,& Manthey,R. (2008). A Transformation-Based Approach to View Updating in Stratifiable Deductive Databases. *FoIKS*, 253-271.

[4] Bessant, B., et al.(1998). Combining Nonmonotonic Reasoning and Belief Revision: A Practical Approach. *AIMSA*, 115-128.

[5] Biskup, J. (2012). Inference-usability confinement by maintaining inference-proof views of an information system. *IJCSE*. **7** (1), 17-37.

[6] Caroprese, L., et al.(2012). The View-Update Problem for Indefinite Databases. *JELIA*.

[7] Chandrabose, A.,& Dung, P.M.(1994). Belief Dynamics, Abduction, and Database. *JELIA*, 66-85.

[8] Chandrabose A.(1995), *Dynamics of Belief: Epistmology, Abduction and Database Update*. Phd Thesis, AIT.

[9] Calvanese, D., et al. (2012). View-based query answering in Description Logics Semantics and complexity. *J. Comput. Syst. Sci* **78**(1), 26-46.

[10] Chen, H., & Liao, H. (2010). A Comparative Study of View Update Problem. *DSDE*, 83-89.

[11] Christiansen, H., & Martinenghi, D. (2006) On Simplification of Database Integrity Constraints. *Fundam. Inform.* **71** (4), 371-417.

[12] Christiansen, H., & Dahl,V. (2009). Abductive Logic Grammars. *WoLLIC*, 170-181.

[13] Codd, E.F. (1979). Extending the Database Relational Model to Capture More Meaning. *ACM Trans. Database Syst.* **4**(4), 397-434.

[14] Cong, G., et al. (2012). On the Complexity of View Update Analysis and Its Application to Annotation Propagation. *IEEE Trans. Knowl. Data Eng.***24**(3), 506-519.

[15] Delgrande, J.P, & Peppas, P. (2011). Revising Horn Theories. *IJCAI*, 839-844.

[16] Eiter, T., & Makino,K. (2007). On computing all abductive explanations from a propositional Horn theory. *J. ACM* **54** (5).

[17] Falappa, M.A., et al.(2012). Prioritized and Non-prioritized Multiple Change on Belief Bases. *J. Philosophical Logic* **41** (1), 77-113.

[18] Ferm, E.L., & Hansson, S.O. (2011). AGM 25 Years - Twenty-Five Years of Research in Belief Change. *J. Philosophical Logic* **40** (2),295-331.

[19] Godfrey, P., & et al. (1998). Integrity Constraints: Semantics and Applications. *Logics for Databases and Information Systems*, 265-306.

[20] Hansson, S.O. (1996). A Test Battery for Rational Database Updating. *Artif. Intell***82** (1-2), 341-352.

[21] Hansson, S.O. (1997).*A Textbook of Belief Dynamics*. Kluwer Academic Publishers, Dordrecht.

[22] Herzig, A. & Rifi,O. (1999). Propositional Belief Base Update and Minimal Change. *Artif. Intell.* **115**(1), 107-138.
[23] Keller, A. (1985).*Updating Relational Databases Through Views*. Phd Thesis.
[24] Kogalovsky, M.R. (2012). Ontology-based data access systems. *Programming and Computer Software* **38**(4), 167-182.
[25] Konieczny, S. (2011). Dynamics of Beliefs. *SUM*, 61-74.
[26] Kowalski, R. (1994). *Logic without model theory*. Technical Report, Department of Computing, Imperial College, London, U.K.
[27] Lakemeyer, G. (1995). A Logical Account of Relevance. *IJCAI* (1), 853-861.
[28] Langlois, M., et al. (2008). Horn Complements: Towards Horn-to-Horn Belief Revision. *AAAI*, 466-471.
[29] Laurent, D., et al. (1998). Updating Intensional Predicates in Deductive Databases. *Data Knowl. Eng.* **26**(1), 37-70.
[30] Liberatore, P. (1997). The Complexity of Belief Update (Extended in 2003). *IJCAI*(1), 68-73.
[31] Liberatore, P., & Schaerf, M. (2004). The Compactness of Belief Revision and Update Operators. *Fundam. Inform.* **62**(3-4), 377-393.
[32] Lobo, J., & Trajcevski, G. (1997). Minimal and Consistent Evolution of Knowledge Bases, *Journal of Applied Non-Classical Logics* **7**(1).
[33] Lu, W. (1999). View Updates in Disjunctive Deductive Databases Based on SLD-Resolution. *KRDB*, 31-35.
[34] Makinson, D. (1997). Screened Revision, *Theoria* **63**, 14-23.
[35] Mayol, E., & Teniente, E. (1999). A Survey of Current Methods for Integrity Constraint Maintenance and View Updating. *ER (Workshops)*, 62-73.
[36] Meyden, R. (1998). Logical Approaches to Incomplete Information: A Survey. *Logics for Databases and Information Systems*, 307-356.
[37] Minker, J. (1996). Logic and Databases: A 20 Year Retrospective. *Logic in Databases*, 3-57.
[38] Mota-Herranz, L., et al. (2000). Transaction Trees for Knowledge Revision, *FQAS*, 182-191.
[39] Nayak, A., et al. (2006). Forgetting and Knowledge Update. *Australian Conference on Artificial Intelligence*, 131-140.
[40] Nayak, A. (2011). Is Revision a Special Kind of Update? *Australasian Conference on Artificial Intelligence*, 432-441.
[41] Nebel, B. (1998). How Hard is it to Revise a Belief Base? *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, 77-145.
[42] Papini, O.(2000). Knowledge-base revision. *The Knowledge Engineering Review* **15**(4), 339 - 370.
[43] Qi, G., & Yang, F. (2008). A Survey of Revision Approaches in Description Logics. *Description Logics*.
[44] Schulte, O. (1999). Minimal Belief Change and Pareto-Optimality. *Australian Joint Conference on Artificial Intelligence*, 144-155.
[45] Segerberg, K. (1998). Irrevocable Belief Revision in Dynamic Doxastic Logic. *Notre Dame Journal of Formal Logic* **39**(3), 287-306.
[46] Siebes, A., et al. (1996). Deductive Databases: Challenges, Opportunities and Future Directions (Panel Discussion). *Logic in Databases*, 225-229.
[47] Teniente, E., & Urpí, T. (2003). On the abductive or deductive nature of database schema validation and update processing problems. *TPLP* **3** (3), 287-327.
[48] Wrobel, S. (1995). First order Theory Refinement. *IOS Frontier in AI and Application Series*.

R. Delhibabu is a Phd student in Computer Science and Engineering at Anna University, Chennai. He is working as a research scholar at SSN College of Engineering, Chennai. His research interests include application of mathematical logic in the areas such as database theory and cognitive robotics.

G. Lakemeyer received his Ph.D. from the University of Toronto in 1990 and currently heads the Knowledge-Based Systems Group at RWTH Aachen University. His research interests include knowledge representation and cognitive robotics. He has published more than one hundred scientific papers and has served on numerous program committees. He is an ECCAI Fellow and a member of the Editorial Board of Artificial Intelligence and other journals.