Knowledge base dynamics, Abduction, and Database Updates

Chandrabose ARAVINDAN[1] and Phan Minh DUNG
Computer Science Program, Asian Institute of Technology
P.O. Box 2754, Bangkok 10501. Thailand.
Internet: <dung@cs.ait.ac.th>

June 1, 1995

[1]Current address: Universität Koblenz-Landau, Fachbereich Informatik, Rheinau 1, D-56075 Koblenz, Germany. Internet: <arvind@informatik.uni-koblenz.de>

Abstract

In this report[1], we argue that to apply rationality results of belief dynamics theory to various practical problems, it should be generalised in two respects: first of all, it should allow certain part of belief to be declared immutable; and secondly, the belief state need not be deductively closed. Such a generalisation of belief dynamics, referred to as knowledge base dynamics, is presented, along with the concept of generalised contraction to contract a sentence from knowledge base. We show that knowledge base dynamics has interesting connections with abduction, thus enabling us to use abductive procedures to realize contractions. Finally, we demonstrate how knowledge base dynamics can provide an axiomatic characterisation for deleting view atoms from databases.

KEYWORDS: epistemology, belief dynamics, rationality postulates, abduction, databases, view updates

[1]this report is an improved version of [AD4] and a part of first author's dissertation [A1]

Contents

Chapter 1

Introduction

We live in a constantly changing world, and consequently our beliefs have to be revised when there is new information. The central problem of epistemology is to study when we can be sure that we have revised our beliefs rationally. This has been studied extensively at a more general and abstract level by the researchers from the field of philosophy, leading to a new branch of study: *the belief dynamics*. Assuming a belief state of deductively closed set of sentences, Alchourrón, Gärdenfors, and Makinson, propose certain rationality postulates, popularly known as *AGM-postulates*, to be satisfied when the belief set is contracted or revised [AGM, AM, GM, G, Gr, M1]. The AGM-model for belief dynamics has gained popularity due to its attractive features like constructive models of contraction (partial meet contraction) and representation theorems. However, various researchers have pointed out the disadvantages of working with infinite belief sets and question the rationality of one of the AGM-postulates (recovery) [H1, H2, H3, H5, M2, N1, N2]. Consequently Fuhrman, Hansson, Nebel, and others propose alternate belief models based on set of sentences that need not be deductively closed. Other approaches to belief dynamics include: the model-theoretic approach of Katsuno and Mendelzon [KM]; relational belief revision of Lindström and Rabinowicz [LR]; and sphere model of Grove [Gr].

However, these philosophical works are mainly at an abstract level, paying little attention to computational aspects. It is important to study how the results of belief dynamics can be realized in real applications, such as view updates in databases, diagnosis, abductive reasoning, counterfactuals, non-monotonic reasoning etc. In his recent paper [H4], Hansson has pointed out the gap between philosophical approaches and computational approaches, and reviews how the researchers from the philosophical side strive to move towards realism. The belief base approach of Hansson, mentioned above, is in fact a result of such an attempt. However, the results of belief dynamics are still far away from concrete applications, like view updates in databases, and computational tractability and efficiency of algorithms satisfying the rationality postulates are still unclear.

On the other hand, works on concrete problems like view updates, concentrate on operational aspects, without explicitly formalising the intuition behind the update. A lot of algorithms have been proposed for view updates in relational databases [DB, Ke, La, and the references therein], and algorithms based on deduction trees have been proposed to carry out view updates on deductive databases [Br, D, GL1, GL2, KaM, PCA, SK, T, and the references therein] ([A] provides a good survey of view updates in databases). Clearly, these algorithms for view updates are not at all arbitrary, and there are certain guiding factors behind them. For example, all the view definitions are considered as immutable and any algorithm for view update changes only the base facts. Moreover, when a view atom A is to be deleted, any algorithm has to make sure that A does not follow from the updated database. Are these two the only guidelines for view updates? Will any algorithm that satisfies these two policies be considered rational? Let us motivate the discussion with an example.

Example 1 Consider a deductive database, as follows:

$$\begin{array}{l} student\_dean(X, Y) \leftarrow student\_school(X, Z), school\_dean(Z, Y) \\ school\_dean(cs, sharma) \leftarrow \\ student\_school(dung, cs) \leftarrow \\ student\_school(arvind, cs) \leftarrow \end{array}$$

Suppose the view atom $student\_dean(dung, sharma)$, which currently follows from the database, is to be deleted. Assume that we have an algorithm that deletes both the base facts $(student\_school(dung, cs) \leftarrow)$ and $(student\_school(arvind, cs) \leftarrow)$ from the database. Clearly, this satisfies both the guidelines set above. But, no rational person will agree with such an algorithm, because the fact $(student\_school(arvind, cs) \leftarrow)$ is not "relevant" to the view atom to be deleted. ■

The above example highlights the need for some kind of "relevance policy" to be adopted when a view atom is to be deleted from a deductive database. How many such axioms and policies do we need to characterise a "good" view update? When are we sure that our algorithm for view update is "rational"? Clearly, there is a need for an axiomatic characterisation of view updates. By axiomatic characterisation, we mean explicitly listing all the rationality axioms that are to be satisfied by any algorithm for view update.

Thus we have abstract notions and results from the field of belief dynamics, whose acceptability in terms of computational efficiency has to be studied in concrete applications, in one hand; and algorithms for view updates in databases, whose rationality has to be studied against a scale, on the other hand. Our goal now is to bring these two fields closer to share the concepts and results across the fields.

The above example, also illustrates two other important points: First of all, certain part of the belief is considered as immutable and not to be changed (view definitions, in this case). Secondly, our belief state is finite and we are not interested in every logical consequence of the belief (such as $student\_school(arvind, cs) \lor \neg student\_school(dung, cs)$). So, to suitably adopt belief dynamics for concrete applications, in this report we present a generalised theory, referred to as *knowledge base dynamics* and bring out its relationship with abduction. We believe that knowledge base dynamics can provide the vital link between theory of belief dynamics and concrete applications, and to this effect we demonstrate how it provides an axiomatic characterisation for deleting view atoms from definite deductive databases. We observe that the algorithms proposed in the literature, such as the ones given by Tomasic [T], Dayal & Bernstein [DB], Keller [Ke], Langerak [La], do comply with our axioms. An interesting point to note is that, in the case of deductive databases, Tomasic's algorithm that satisfies all our rationality requirements, is computationally inefficient, while efficient algorithms, such as the ones proposed by Decker [De], Guessoum & Lloyd [GL1, GL2], Kakas & Mancarella [KaM], violate certain axioms. This raises questions on rationality of those axioms on practical grounds. Thus, as a result of our axiomatisation, we raise an important problem of balancing rationality with computational efficiency, which has to be addressed from both philosophical and computational perspectives.

The rest of the report is organised as follows: In chapter 2, we review the concepts of belief dynamics, and introduce knowledge base dynamics along with the concept of generalised contraction. Chapter 3 studies the relationship between knowledge base dynamics and abduction and shows how abductive procedures could be used to realise contraction. In chapter 4, we discuss an important application of knowledge base dynamics in providing an axiomatic characterisation for deleting view atoms from databases. We also analyse how view update algorithms proposed in the literature comply with this axiomatic characterisation, pointing out that computationally more efficient algorithms do not satisfy some rationality postulates, calling for further research in this direction. Chapter 5 concludes the paper with a discussion on the results of this research and directions for further research.

Chapter 2

Epistemology: Rationality postulates and constructions

Dynamics of belief has been studied at a more abstract level of philosophy resulting in certain rationality postulates that have to be satisfied while revising or contracting a state of belief. The model of belief dynamics proposed by Alchourrón, Gärdenfors, and Makinson, popularly known as the AGM-model, assumes a belief state of deductively closed (according to a suitable logic) set of sentences from a language L [AGM, AM, GM, G, Gr, and the references therein]. However, researchers like Fuhrmann, Hansson, Nebel, and others have studied revision and contraction of belief bases (set of sentences that need not be deductively closed) also [F, H2, H3, H5, N1, N2, and the references therein]. In this section, we briefly review the concepts from these works, that are relevant to this paper, along with our contribution to this field.

A state of belief can be *expanded* by simply adding a new information that is consistent with the present state of belief; *revised* with a new information that contradicts the current belief state; or some belief from the current state can be *contracted*. AGM proposes the following rationality postulates that any contraction function $\dot{-}$ is expected to satisfy. The reader is referred to [AGM, G, Gr] for the motivation and intuition behind these postulates.

Definition 1 (Rationality postulates for contracting a belief set)
For any sentence α and belief set[1] K:

(K1)	(Closure)	$K\dot{-}\alpha$ is a belief set.
(K2)	(Inclusion)	$K\dot{-}\alpha \subseteq K$
(K3)	(Vacuity)	If $\alpha \notin K$, then $K\dot{-}\alpha = K$
(K4)	(Success)	If $\nvdash \alpha$, then $\alpha \notin K\dot{-}\alpha$
(K5)	(Preservation)	If $\vdash \alpha \leftrightarrow \beta$, then $K\dot{-}\alpha = K\dot{-}\beta$
(K6)	(Recovery)	$K \subseteq (K\dot{-}\alpha) + \alpha$
(K7)	(Conjunction 1)	$K\dot{-}\alpha \cap K\dot{-}\beta \subseteq K\dot{-}(\alpha \wedge \beta)$
(K8)	(Conjunction 2)	If $\alpha \notin K\dot{-}(\alpha \wedge \beta)$, then $K\dot{-}(\alpha \wedge \beta) \subseteq K\dot{-}\alpha$ ■

A function to contract α from K can be defined using the set of inclusion-maximal subsets of K that do not imply α, which is formally defined as: $K\downarrow \alpha = \{K' \subseteq K \mid K' \nvdash \alpha$ and if $K' \subset K'' \subseteq K$, then K''

[1] A belief set K is a deductively closed (according to a suitable logic) set of sentences from L, i.e. K=Cn(K) where Cn is a consequence operator

$\vdash \alpha\}$. Now, a *partial meet contraction* of α from K is defined as: $K \dot{-}_p \alpha = \bigcap \gamma(K \downarrow \alpha)$, where, γ is a selection function that selects a non-empty subset of $K\downarrow\alpha$, when the later is not empty. When $K\downarrow\alpha$ is empty, $\gamma(K\downarrow\alpha)=\{K\}$. It is not difficult to see that the set of all partial meet contractions of α from K, which we refer to as the *solution space*, forms a complete semi-lattice[2] wrt set-inclusion. Two interesting limiting cases of partial meet contractions are: *maxichoice contraction*, when γ selects a unique element of $K\downarrow\alpha$; and *full meet contraction*, when γ selects all the elements of $K\downarrow\alpha$. It is clear that all maxichoice contractions are the maximal elements in the semi-lattice, while the full meet contraction is the minimal element. A partial meet contraction is said to be *transitively relational*, iff there exists a transitive and reflexive ordering relation $\leq$ s.t. for all nonempty $K\downarrow \alpha$: $\gamma(K\downarrow \alpha) = \{K' \in K\downarrow \alpha \mid \forall K'' \in K\downarrow \alpha\colon K'' \leq K'\}$.

Theorem 1 ([Gr]) *For every belief set K, $\dot{-}_p$ is a partial meet contraction function iff it satisfies postulates (K1) to (K6). It is a transitively relational partial meet contraction function iff it satisfies postulates (K1) to (K8).* ■

As we have argued in the introduction, in concrete applications such as diagnosis and database updates, it is not practical to work with infinite deductively closed belief sets. Further, in most of the applications, there is a need to declare certain part of the knowledge as immutable that should not be modified (system description in the case of diagnosis and view definitions in the case of view updates in databases). So, how the rationality axioms of AGM can be used to study the rationality of change in these applications?

The problem of immutability of certain knowledge could be conceptually solved in the epistemic entrenchment framework [AD2, GM, and the references therein], by selecting a suitable entrenchment order in which every immutable sentence is strictly more entrenched than other updatable sentences. However, the other problem of dealing with belief bases is not easy to capture in AGM framework. Both Reiter [R] and Fagin et al. [FKUV] follow the same intuition of AGM and construct contraction of α from B, based on maximal subsets of B that do not imply α. So, it is interesting to ask, suppose in AGM framework if we let K to be not deductively closed, then what rationality axioms are satisfied by partial meet contractions? This has been studied by Nebel [N1] and Hansson [H5], and following is their result.

Theorem 2 *Let K be a belief set that is not deductively closed and α a sentence. Then the partial meet contraction $K \dot{-} \alpha$ of α from K satisfy (K1) to (K5). The recovery postulate (K6) is not always satisfied when K is allowed not to be deductively closed.* ■

This definitely raises questions on the rationality of the recovery postulate, and this has been pointed out by Hansson [H1], Makinson [M2], and Nebel [N1]. The implementation problem of belief sets has been discussed by many researchers such as Fuhrmann [F], Hansson [H3], Kowalski [K], Nebel [N1], and consequently Hansson [H3] motivates the need for belief dynamics model based on belief bases that need not be deductively closed.

In the sequel, we generalise the dynamics of belief bases for the case where some part of the knowledge is declared to be immutable. We refer to the new theory as knowledge base dynamics, and propose generalised contraction for the same.

2.1 Knowledge base dynamics

Formally, a knowledge base KB is defined as a finite set of sentences from language L, and divided into two parts: an immutable theory KB_I, which is the fixed part of the knowledge; and an updatable theory

[2] a set S forms a complete semi-lattice wrt a partial order iff each non-empty subset of S has a greatest lower bound and each increasing sequence of S has a least upper bound.

KB_U. Because of the duality of revision and contraction, it is enough to consider one, and in the sequel, contraction of a sentence α from KB, written as KB—α is discussed.

Like in the case of belief sets, the rationality postulates for contracting α from KB can be formulated. First of all, when some information is thrown away, no new information can enter the knowledge base, and this is captured by the following postulate.

(KB—1) (Inclusion) KB—$\alpha \subseteq$ KB

The additional immutability restriction requires that KB_I is retained in any contraction of a sentence α from KB, since no sentence from KB_I can be thrown away. This introduces the following rationality postulate.

(KB—2) (Immutable-inclusion) $KB_I \subseteq$ KB—α

Since we wish to throw away as little information as possible, when α is not accepted in the current knowledge base, contraction of α from KB should not alter KB.

(KB—3) (Vacuity) If $\alpha \notin$ Cn(KB), then KB—α = KB

As in the case of belief sets, the contraction should be successful, i.e. if $\nvdash \alpha$, then $\alpha \notin$ Cn(KB—α). But, since the immutable theory never changes, any knowledge implied by it can also never change. Hence, the above success postulate is rewritten as follows.

(KB—4) (Immutable-success) If $KB_I \nvdash \alpha$, then $\alpha \notin$ Cn(KB—α)

To ensure that irrelevance of syntax is achieved, the belief set approach of AGM requires that contraction function behaves the same with two logically equivalent sentences. To generalise this for our case, where we have immutable knowledge, we need the notion of KB-equivalence.

Definition 2 Let KB be a knowledge base with an immutable part KB_I. Let α and β be any two sentences. Then, α and β are said to be *KB-equivalent* iff the following condition is satisfied:
$\forall$ set of sentences E: $KB_I \cup E \vdash \alpha$ iff $KB_I \cup E \vdash \beta$ ■

With this concept, the irrelevance of syntax can be captured by the following axiom.

(KB—5) (Preservation) If α and β are KB-equivalent, then KB—α = KB—β

Finally, to contract α from KB, only those information that are relevant to α in some sense can be thrown out (as illustrated by example in the introduction). As a first step, this relevance policy can be captured by the following strong axiom.

(KB—6.1) (Strong relevance) If $\beta \in$ KB\KB—α, then (KB—α) $\cup \{\beta\} \vdash \alpha$

This is a very strong axiom allowing only minimal changes, and certain rational contractions can not be carried out. This is illustrated by the following example.

Example 2 Consider a database with a rule that a student, who is currently doing her thesis is entitled for scholarship, represented as $schol(X) \leftarrow student(X) \wedge thesis(X)$. With additional facts that Arvind is a student ($student(Arvind)$), and he is doing his thesis ($thesis(Arvind)$), the database implies Arvind is entitled for a scholarship ($schol(Arvind)$). If this derived fact is to be deleted (perhaps Arvind has completed his thesis and graduated), it perfectly makes sense to delete both $student(Arvind)$ and $thesis(Arvind)$ from the database. ■

Moreover, in the context of deductive databases (discussed later), it is more expensive to compute this minimal change. So, relaxing this condition, like Hansson [H1, H2], this can be weakened to relevance as follows:

(KB—6.2) (Relevance) If $\beta \in$ KB\KB—α, then $\exists$KB' with KB—$\alpha \subseteq$ KB' $\subseteq$ KB s.t. $\alpha \notin$ Cn(KB') and $\alpha \in$ Cn(KB' $\cup \{\beta\}$)

The above relevance policy can still not permit rational contractions, and this is best illustrated by Hansson's example [H1] which is reproduced below.

Example 3 Consider the following real life conversation between two persons P1 and P2:

P1: Last summer I saw a three-toed woodpecker just outside my window. I could clearly see its red forehead and its red rump.

P2: You must be mistaken. The three-toed woodpecker does not have a red forehead or a red rump.

P1: You make me uncertain. Thinking about it, the only thing I am certain of is that the bird had a red forehead.

Here, P1's original beliefs were that the bird was a three-toed woodpecker (x), that it had a red forehead (y) and that it had a red rump (z),i.e. $B = \{x, y, z\}$. She contracted $x \wedge (y \vee z)$ that had been denied by P2, and retained only y, i.e. B—$(x \wedge (y \vee z))$={y}. It is easy to see that, this contraction violates relevance axiom. ■

So, the relevance axiom is further weakened to allow these possibilities, and arrive at the following, which is referred to as 'core-retainment' by Hansson [H1].

(KB—6.3) (Weak relevance) If $\beta \in$ KB\KB—α, then $\exists$KB' with KB' $\subseteq$ KB s.t. $\alpha \notin$ Cn(KB') and $\alpha \in$ Cn(KB' $\cup \{\beta\}$)

Having discussed the rationality postulates, the next step is to provide a construction for contraction of α from KB. The construction we provide below is a simple generalisation of partial meet contraction discussed before, and it corresponds to the weakest form of relevance policy.

Definition 3 Let KB be a knowledge base with an immutable part KB_I, and α a sentence. Then we define (KB$\downarrow\alpha$)* based on (KB$\downarrow\alpha$) as follows:

$$(KB\downarrow\alpha)^* = \begin{cases} \{KB\}, & \text{when } KB\downarrow\alpha = \emptyset \\ \{KB_i \mid \exists KB' \in KB\downarrow\alpha \ \& \\ \qquad \bigcap(KB\downarrow\alpha) \subseteq KB_i \subseteq KB'\}, & \text{otherwise} \end{cases}$$

Now, the set of all *generalised contractions* of α from KB, satisfying the immutability condition, is given as {KB' | KB' $\in$ (KB$\downarrow\alpha$)* and $KB_I \subseteq$ KB'}, when KB_I does not derive α; and is given as {KB} when $KB_I \vdash \alpha$. ■

Theorem 3 *For every knowledge base KB, — is a generalized contraction function iff it satisfies the postulates (KB—1) to (KB—5) and (KB—6.3).*

Proof: (If part) — satisfies (KB—1) to (KB—5) and (KB—6.3). We want to show that KB—α is a generalized contraction of α from KB. When $KB_I \vdash \alpha$, (KB—1) and (KB—6.3) imply that KB—α = KB, coinciding with generalized contraction. When $KB_I \nvdash \alpha$, the required result follows from the following two observations:

1. $\exists$KB' $\in$ KB$\downarrow\alpha$ s.t. KB—$\alpha \subseteq$ KB' (when $KB_I \nvdash \alpha$)

   Since — satisfies inclusion (KB—1), KB—α is contained in KB. Also, satisfaction of success postulate (KB—4) ensures that $\alpha \notin$ Cn(KB—α). Every element of KB$\downarrow\alpha$ is a inclusion maximal subset that does not derive α, and so any subset of KB that does not derive α must be contained in a member of KB$\downarrow\alpha$.

2. $\bigcap(KB\downarrow\alpha) \subseteq KB—\alpha$ (when $KB_I \nvdash \alpha$)

Consider any $\beta \in \bigcap(KB\downarrow\alpha)$. Assume that $\beta \notin KB—\alpha$. Since — satisfies weak relevance postulate (B—6.3), it follows that there exists a set KB' s.t. KB' $\subseteq$ KB; $\alpha \notin Cn(KB')$; and $\alpha \in Cn(KB' \cup \{\beta\})$. But this contradicts the fact that β is present in every maximal subset of KB that does not derive α. Hence β must be in $KB—\alpha$.

(Only if part) Let $KB—\alpha$ be a generalised contraction of α from KB. We have to show that $KB—\alpha$ satisfies the postulates (KB—1) to (KB—5) and (KB—6.3).

(KB—1) Obviously, $KB—\alpha \subseteq KB$.

(KB—2) $KB_I \subseteq KB—\alpha$ also follows obviously from the definition of generalised contraction of α from KB.

(KB—3) If, $\alpha \notin Cn(KB)$, then $KB\downarrow\alpha = \{KB\}$, and hence $KB—\alpha = KB$. So, vacuity is satisfied.

(KB—4) When $KB_I \nvdash \alpha$, every member of $KB\downarrow\alpha$ does not derive α. Thus $\alpha \notin Cn(KB—\alpha)$, and so success is satisfied.

(KB—5) If, α and β are KB-equivalent, then $KB\downarrow\alpha = KB\downarrow\beta$. So, preservation is satisfied.

(KB—6.3) Let $\beta \in KB\backslash(KB—\alpha)$. Then it is clear that $\beta \notin \bigcap(KB\downarrow\alpha)$, and hence $\exists KB' \in KB\downarrow\alpha$ s.t. $\beta \notin KB'$. From the definition of $KB\downarrow\alpha$, it is clear that $KB' \subset KB$; $\alpha \notin Cn(KB')$; and $\alpha \in Cn(KB' \cup \{\beta\})$. So, weak relevance is satisfied. ■

Chapter 3

Knowledge base dynamics and abduction

In this chapter, we study the relationship between knowledge base dynamics, discussed in the previous chapter, and abduction, a well-known form of reasoning. This study helps to bring these two fields together, so that abductive procedures could be used to implement contraction. For this purpose, we use the concepts of generalized kernel contraction, an extension of kernel contraction introduced for belief bases by Hansson [H6]. We first observe that generalized kernel contraction coincides with that of knowledge base contraction, and then proceed to show its relationship with abduction.

3.1 Generalized Kernel contraction

An alternate construction of contraction over belief bases (without any immutability part), referred to as kernel contraction, satisfying the postulates (KB—1) to (KB—5) and (KB—6.3), has been forwarded by [H6]. In this section, we generalize that idea to include immutable knowledge base.

To delete a belief α from a knowledge base KB, the idea of kernel contraction is to discard atleast one element from every inclusion-minimal subset of KB that derives α. Because of the immutable-inclusion postulate, no sentence from KB_I can be deleted.

Definition 4 Let KB be a knowledge base and α a belief. Then, the *kernel set*, written as $KB \perp \alpha$, is the set consisting of all inclusion-minimal subsets of KB that derive α, i.e. $X \in KB \perp \alpha$ iff: $X \subseteq KB$ and $X \vdash \alpha$ and if $Y \subset X$, then $Y \nvdash \alpha$. Every element of $KB \perp \alpha$ is referred to as an *α-kernel* of KB.

A *hitting set* H for $KB \perp \alpha$ is defined as a set s.t. (1) $H \subseteq \bigcup(KB \perp \alpha)$ and (2) $H \cap KB_I$ is empty and (3) If $\emptyset \neq X \in KB \perp \alpha$ 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 $\bigcup(KB \perp \alpha)$, and *minimal* if no proper subset of H is a hitting set for $KB \perp \alpha$.

An *incision function* σ for KB is a function s.t. for all α, $\sigma(KB \perp \alpha)$ is a hitting set for $KB \perp \alpha$. An operator $—_k$ for KB is a generalized kernel contraction iff there is some incision function σ for KB s.t. $KB —_k \alpha = KB \setminus \sigma(KB \perp \alpha)$ for all beliefs α. ■

From the definition of hitting set, it is clear that when $KB \vdash \alpha$, ϕ is the only hitting set of $KB \perp \alpha$. On the otherhand, when $KB_I \nvdash \alpha$, the definition ensures that only updatable elements are removed, and α does not follow from the contraction. Thus, immutable-inclusion (KB—2) and immutable-success (KB—4) are satisfied by gneralized kernal contraction of α from KB.

Theorem 4 *For every knowledge base KB, $-_k$ is a generalized kernel contraction iff it satisfies the postulates (KB—1) to (KB—5) and (KB—6.3).*

Proof: Follows from the above observation and the result of [H6]. ■

From the theorems 3 and 4, it immediately follows that a contraction operation on a knowledge base is a generalized kernel contraction iff it is a generalized contraction. The following theorem formalizes this with additional insights into the relationship between kernel and generalized contractions.

Theorem 5

1. *A contraction operation over a knowledge base KB is a generalized kernel contraction over KB iff it is a generalized contraction over KB.*
2. *When the incision function σ is minimal, i.e. the hitting set defined by σ is inclusion-minimal, then the generalized kernel contraction defined by σ is a maxichoice contraction of α from KB.*
3. *When the incision function σ is maximal, i.e. $\sigma(\mathrm{KB}\perp\alpha)$ consists of all updatable statements from $\bigcup(\mathrm{B}\perp\alpha)$, then the kernel contraction defined by σ is the minimal generalized contraction of α from KB.* ■

We close this section with rather obvious observation that generalized kernal contraction coincides with Hansson's kernal contraction when $\mathrm{KB_I}$ is empty.

3.2 Relationship with abduction

In the sequel, we study the relationship between knowledge base dynamics and abduction, a different form of reasoning introduced by the philosopher Pierce (cf. [KaKT]). We show how abductive procedures could be used to realize contractions with immutability conditions. For abductive reasoning, a special subset of ground literals of language L, are designated as *abducibles Ab*. The predicates that appear in abducibles are referred to as *abducible predicates*. An *abductive framework* $<\mathrm{P},\mathrm{Ab}>$ stands for a theory P, which is a set of sentences from L, with possible hypotheses *Ab*. An abductive framework for a knowledge base $\mathrm{KB} = \mathrm{KB_I} \cup \mathrm{KB_U}$ can be given as follows: Let $\mathrm{P} = \mathrm{KB_I} \cup \{\alpha \leftrightarrow \beta \mid \alpha$ is a sentence in $\mathrm{KB_U}$ and β is an abducible from *Ab* that does not appear in KB$\}$; then the required abductive framework is $<\mathrm{P},\mathrm{Ab}>$.

Notation: In the sequel, without any loss of generality, we assume that $\mathrm{KB_U}$ is a set of abducibles, so that the abductive framework for a knowledge base KB is simply $<\mathrm{KB_I},\mathrm{Ab}>$. ■

Given a knowledge base KB and an observation (a sentence) α, the problem of abduction is to explain α in terms of abducibles, i.e. to generate a set of abducibles Δ s.t. $\mathrm{KB_I} \cup \Delta \vdash \alpha$.

Definition 5 Let KB be a knowledge base and α an observation (sentence) to be explained. Then, a set of abducibles Δ is said to be an *abductive explanation* for α wrt $\mathrm{KB_I}$ iff $\mathrm{KB_I} \cup \Delta \vdash \alpha$. Δ is said to be *minimal* wrt $\mathrm{KB_I}$ iff no proper subset of Δ is an abductive explanation for α, i.e. $\nexists\Delta'$ s.t. $\mathrm{KB_I} \cup \Delta'$ $\vdash \alpha$. ■

Since an incision function removes only updatable elements from each member of the kernel set, to compute a generalized contraction of α from KB, we need to compute only the abducibles 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 6 Let KB be a knowledge base and α an observation to be explained. Then, an abductive explanation Δ for α wrt KB_I is said to be *locally minimal*, iff there exists a subset KB_I' of KB_I s.t. Δ is a minimal abductive explanation of α wrt KB_I'. Δ is said to be *KB-Closed* iff $\Delta \subseteq KB_U$. ■

Example 4 Consider a knowledge base KB whose immutable part $KB_I = \{p \leftarrow q \wedge r,\ p \leftarrow r\}$, where q and r are abducibles. Clearly, $\Delta_1 = \{r\}$ is the only minimal abductive explanation for p wrt KB_I. $\Delta_2 = \{q,r\}$ is an abductive explanation for p wrt KB_I, but not minimal. However, Δ_2 is a locally minimal abductive explanation for p wrt KB_I, since it is a minimal explanation for p wrt $\{p \leftarrow q \wedge r\}$ which is a subset of KB_I. The concept of locally minimal abductive explanation is computationally attractive, since minimal abductive explanation is more expensive to compute. ■

The connection between locally minimal abductive explanation for α wrt KB_I and an α-kernel of KB, which is formalized by the following lemma, follows immediately from their respective definitions.

Lemma 1

1. *Let KB be a knowledge base and α a sentence s.t. $\not\vdash\alpha$. Let Δ be a KB-closed locally minimal abductive explanation for α wrt KB_I. Then, there exists a α-kernel X of KB s.t. $X \cap KB_U = \Delta$.*

2. *Let KB be a knowledge base and α a sentence s.t. $\not\vdash\alpha$. Let X be a α-kernel of KB and $\Delta = X \cap KB_U$. Then, Δ is a KB-closed locally minimal abductive explanation for α wrt KB_I.*

Proof:

1. The fact that $\not\vdash\alpha$ and there exists a KB-closed locally minimal abductive explanation for α wrt KB_I, it is clear that there exists atleast one α-kernel of KB. Suppose Δ is empty (i.e. $KB_I \vdash \alpha$), then the required result follows immediately. If not, since Δ is a locally minimal abductive explanation, there exists a minimal subset KB_I' $\subseteq KB_I$, s.t. Δ is a minimal abductive explanation of α wrt KB_I'. Since, Δ is KB-closed, it is not difficult to see that KB_I' $\cup\ \Delta$ is a α-kernel of KB.

2. Since X is a α-kernel of KB and Δ is the set of all abducibles in X, it follows that Δ is a minimal abductive explanation of α wrt $X \setminus \Delta$. It is obvious that Δ is KB-closed, and so Δ is a KB-closed locally minimal abductive explanation for α wrt KB_I. ■

An immediate consequence of the above lemma is that, it is enough to compute all the KB-closed locally minimal abductive explanations for α wrt KB_I to contract α from KB. Thus, well-known abductive procedures (e.g. [DDe, Du, KaKT]) to compute an abductive explanation for α wrt KB_I could be used to compute kernel contraction, and we arrive at Algorithm 1.

Theorem 6 *Let KB be a knowledge base and α a sentence.*

1. *If Algorithm 1 produces KB' as a result of contracting α from KB, then KB' is a generalized contraction of α from KB.*

2. *If KB' is a generalized contraction of α from KB, then there exists an incision function σ s.t. KB' is produced by Algorithm 1 as a result of contracting α from KB, using σ at step 2.*

Proof: Follows from Lemma 1 and Theorem 5 ■

Algorithm 1 General contraction algorithm

Input: A knowledge base $KB = KB_I \cup KB_U$ and a sentence α to be contracted.

Output: A new knowledge base KB' = $KB_I \cup KB_U$', s.t. KB' is a generalized contraction of α from KB.

begin

1. Construct a set $S = \{X \mid X$ is a KB-closed locally minimal abductive explanation for α wrt $KB_I\}$.
2. Determine a hitting set $\sigma(S)$.
3. Produce KB' = $KB_I \cup (KB_U \backslash \sigma(S))$ as a result.

end.

Chapter 4

Application: View updates in databases

An important application of knowledge base dynamics, discussed in the previous chapter, 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, a set of definite program clauses; and an *extensional database* EDB, 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*. The reader is referred to [L, and the references therein], for more information on definite programs, the least Herbrand model semantics, and SLD-derivations. 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 with a view predicate is said to be a *view atom*, and similarly an atom with base predicate is a *base atom*. Further we assume that IDB does not contain any unit clauses and no predicate defined in a given DDB is both view and base.

Two kinds of view updates can be carried out on a DDB: An atom, that does not currently follow from DDB, can be *inserted*; or an atom, that currently follows from DDB, can be *deleted*. In this paper, we consider only deletion of an atom from a DDB. When an atom A is to be deleted, the view update problem is to delete only some relevant EDB facts, so that the modified EDB together with IDB will satisfy the deletion of A from DDB. View update problem, in the context of deductive databases, has been studied by various authors and algorithms based on SLD-trees have been proposed [Br, D, GL1, GL2, KaM, PCA, T]. 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 a DDB can be considered as a knowledge base to be revised. The IDB is the immutable part of the knowledge base, 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 the language underlying a DDB is fixed and the semantics of DDB is the least Herbrand model over this fixed language. Therefore, the IDB is practically a shorthand of its ground instantiation[1] , written as $\mathrm{IDB_G}$. Thus, a DDB represents a knowledge base where the immutable part is given by $\mathrm{IDB_G}$ and updatable part is the EDB. Hence, the rationality postulates (KB—1), (KB—2), (KB—3), (KB—4), (KB—5), and (KB—6.3) provide an axiomatic characterization for deleting a view atom A from a definite database DDB, and a generalized contraction of A from DDB achieves deletion of A from DDB.

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

As observed by Kowalski in [K], logic can provide a conceptual level understanding of relational databases, and hence rationality postulates (KB—1), (KB—2), (KB—3), (KB—4), (KB—5), and (KB—6.3) can provide an axiomatic characterization for view deletion 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 delete view extensions from relational and deductive databases.

But before discussing the rationality postulates and algorithms, we want to make it precise, how a relational database, along with operations on relations, can be represented by a definite deductive database. We assume the reader is familiar with relational database concepts, as provided by, for example, Maier in [Ma]. A *relational scheme* R can be thought of as a base predicate whose arguments define the *attributes* $\mathbf{A}$ of the scheme. Its *relational extension* r, is a finite set of base atoms $R(\mathbf{A})$ containing the predicate R. A *database schema* 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, a relational database can be represented by a EDB $= \bigcup_{i=1,\ldots,n} R_i(\mathbf{A}_i)$.

An unary operator *selection* on r, selects those elements of r whose chosen attribute A matches a given value. $\sigma_{A=a}(r)$, which stands for selection operation on r s.t. value of attribute A is a, is defined as, $\sigma_{A=a}(r) = \{ t \in r \mid t(A) = a\}$. This selection operator can be captured by a constraint clause, $R_{A=a}(\mathbf{A}) \leftarrow A = a, R(\mathbf{A})$, where $R_{A=a}$ is a new predicate (representing a new relational scheme) and $\mathbf{A}$ is the vector of all attributes of R.

Let r be a relational extension of scheme R with attributes $\mathbf{A}$, and $\mathbf{X}$ a subset of $\mathbf{A}$. The *projection* of r onto $\mathbf{X}$, written as $\pi_{\mathbf{X}}(r)$, is the relation $\pi_{\mathbf{X}}(r) = \{ t(\mathbf{X}) \mid t \in r\}$. Projection can be captured by a constraint clause, $R_{\mathbf{X}}(\mathbf{X}) \leftarrow R(\mathbf{A})$, where $R_{\mathbf{X}}$ is a new predicate.

Join is a binary operator for combining two relations. Let r and s be two relational extensions of schema R (with attributes $\mathbf{R}$) and S (with attributes $\mathbf{S}$) respectively. Let $\mathbf{T} = \mathbf{R} \cup \mathbf{S}$. The join of r and s, written as $r \otimes s$, is the relational extension $q(\boldsymbol{T})$ of all tuples t over $\mathbf{T}$ s.t. there are tuples $t_r \in r$ and $t_s \in s$, with $t_r = t(\mathbf{R})$ and $t_s = t(\mathbf{S})$. Join can be captured by a constraint clause $Q(\mathbf{T}) \leftarrow R(\mathbf{R}), S(\mathbf{S})$.

Example 5 Consider two relational schema R and S, with attributes R = {SCHOOL,DEAN} and S = {STUDENT,SCHOOL,YEAR}. Consider the following extensions r and s:

r	*SCHOOL*	*DEAN*
	cs	sharma

s	*STUDENT*	*SCHOOL*	*YEAR*
	arvind	cs	1989
	dung	cs	1986
	phien	wre	1980

Now a slection of tuples from s, whose SCHOOL value equals 'cs' is given by

$$S_{SCHOOL=cs}(STUDENT, SCHOOL, YEAR) \leftarrow \\ SCHOOL = cs, S(STUDENT, SCHOOL, YEAR)$$

which is easily evaluated to:

$\sigma_{SCHOOL=cs}(s)$	*STUDENT*	*SCHOOL*	*YEAR*
	arvind	cs	1989
	dung	cs	1986

Let $\mathbf{X}$={STUDENT,SCHOOL}. Then, projection of s on $\mathbf{X}$ is given by

$$S_{\mathbf{X}}(STUDENT, SCHOOL) \leftarrow S(STUDENT, SCHOOL, YEAR)$$

which evaluates to:

$\pi_{\mathbf{X}}(s)$	*STUDENT*	*SCHOOL*
	arvind	cs
	dung	cs
	phien	wre

The following rule,

$$T(STUDENT, SCHOOL, YEAR, DEAN) \leftarrow$$
$$S(STUDENT, SCHOOL, YEAR), R(SCHOOL, DEAN)$$

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

$s \otimes r$	*STUDENT*	*SCHOOL*	*YEAR*	*DEAN*
	arvind	cs	1989	sharma
	dung	cs	1986	sharma

■

To sum up, we showed how relational databases and operators on relations can be conceptually captured by definite deductive databases. Further, a definite deductive database can be considered as a knowledge base, and thus rationality postulates and deletion algorithm of the previous section can be applied for view updates in databases.

4.1 View deletion algorithm

Since relational and definite deductive databases can be considered as knowledge bases, and deleting a view atom (tuple) A can be considered as contraction of A, a specific instance of Algorithm 1 can be used to compute deletion of a view atom from 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.

Since we consider definite deductive databases, SLD-trees 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. This proposal is formalised by Algorithm 2.

Algorithm 2 Algorithm to compute all DDB-closed locally minimal abductive explanations of an atom

Input: a definite deductive database $DDB = IDB \cup EDB$ and an atom A

Output: Set of all DDB-closed locally minimal abductive explanations for A wrt IDB_G

begin

1. Construct a complete SLD-tree for $\leftarrow A$ wrt DDB.
2. For every successful branch i: construct $\Delta_i = \{C \mid C \in EDB$ and C is used as an input clause in branch i$\}$.
3. Produce set of all Δ_i computed in the previous step as the result.

end.

Unfortunately, this algorithm does not work as intended for any deductive database, and a counter-example is produced below.

Example 6 Consider a deductive database DDB as follows:

$$\begin{array}{llll} \text{IDB:} & p \leftarrow p, q & \text{EDB:} & q \leftarrow \\ & p \leftarrow r & & r \leftarrow \end{array}$$

An SLD derivation for p wrt DDB is shown in Figure 4.1.

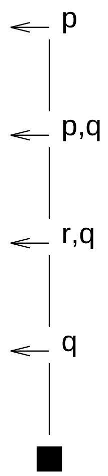

Figure 4.1: SLD derivation for example 6

Collecting all the EDB facts used in this derivation, we obtain $\Delta = \{q, r\}$, which is *not* a locally minimal abductive explanation. In fact, $\{r\}$ is the only locally minimal abductive explanation for p wrt IDB_G. ■

Thus, in general algorithm 2 produces some unexpected sets in addition to locally minimal abductive explanations. The problem is clearly due to the unwanted recursion $p \leftarrow p, q$. Will the algorithm work as intended if we restrict ourselves to acyclic programs[2] [AB] that excludes such loops? One would expect a positive answer, but unfortunately still some unwanted sets may be produced as the following example highlights.

Example 7 Consider the following deductive database, which is clearly acyclic:

$$\begin{array}{llll} \text{IDB:} & p \leftarrow q, q & \text{EDB:} & r \leftarrow \\ & q \leftarrow r & & s \leftarrow \\ & q \leftarrow s & & \end{array}$$

For the atom p, algorithm 2 will compute a Δ as shown in Figure 4.2.

Collecting all the EDB facts used as input clauses in this successful branch, one obtains $\Delta = \{r, s\}$. Clearly this is not a locally minimal abductive explanation for p wrt IDB_G. In fact, this is the union of two locally minimal abductive explanations $\{r\}$ and $\{s\}$. ■

So even for acyclic programs, algorithm 2 does not work as intended (that is to generate all and only the DDB-closed locally minimal abductive explanations). Does this mean that generalised contraction can not be carried out for databases in general?

Probably we should approach the problem from a different perspective. We have seen that algorithm 2 may compute some unwanted sets in addition to the required ones. What exactly are those sets? Is it possible to characterise them? The following lemma answers these questions.

[2]necessary notions of acyclic programs are provided in the appendix

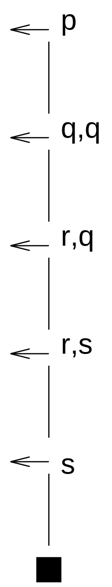

Figure 4.2: SLD derivation for example 7

Lemma 2 *Let DDB = IDB ∪ EDB be a definite deductive database and* A *an atom. Let* S *be the set of all DDB-closed locally minimal abductive explanations for* A *wrt* $\mathrm{IDB_G}$*. Let* S' *be the set of explanations returned by algorithm 2 given DDB and* A *as inputs. Then, the following propositions hold:*

1. $S \subseteq S'$
2. $\forall \Delta' \in S' : \ \exists \Delta \in S$ s.t. $\Delta \subseteq \Delta'$
3. *Suppose DDB is restricted to be acyclic then:* $\forall \Delta' \in S' : \ \Delta' \subseteq \bigcup S$

Proof:

1. Consider a $\Delta \in S$. We need to show that Δ is generated by algorithm 2 at step 2. From lemma 1, it is clear that there exists a A-kernel X of $\mathrm{DDB_G}$ s.t. $X \cap \mathrm{EDB} = \Delta$. Since $X \vdash A$, there must exist a successful derivation for A using only the elements of X as input clauses, and consequently Δ must have been constructed at step 2.
2. Consider a $\Delta' \in S'$. Let Δ' be constructed from a successful branch i. Let X be the set of all input clauses used in the refutation i. Clearly $X \vdash A$. 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 $\Delta = Y \cap \mathrm{EDB}$. Since IDB does not have any unit clauses, Y must contain some EDB facts, and so Δ is not empty and obviously $\Delta \subseteq \Delta'$. But, Y need not be a A-kernel for $\mathrm{IDB_G}$ since Y is not ground in general. But it stands for several A-kernels with the same EDB facts Δ in them. Thus, from lemma 1, Δ is a DDB-closed locally minimal abductive explanation for A wrt $\mathrm{IDB_G}$ 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 to appendix. ■

Having characterised what exactly is computed by algorithm 2, we now proceed to show that this algorithm is useful for view deletion. The key to the solution is the following lemma, which establishes the preservance of hitting set computation among two sets.

Lemma 3

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 a 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.* $S \subseteq S'$ *and for every member X of* $S' \backslash S$*: X contains a member of S and X is contained in* $\bigcup S$*. Then, a set H is a hitting set for S iff it is a hitting set for S'.*

Proof:

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' fo 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'. ■

Thus algorithm 2 in conjunction with an algorithm to compute minimal hitting set can be used to compute maxi-choice contraction of A from DDB. Further, when DDB is acyclic, generalised contraction of A from DDB can be obtained. These algorithms are discussed in the sequel. We start with Algorithm 3 that computes maxi-choice contractions.

Observe that the first two steps of Algorithm 3 are same as those of algorithm 2, 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 2 and lemma 3, this algorithm produces a maxi-choice contraction of A from DDB. This result is formalised below.

Theorem 7 *Let DDB be a definite deductive database and* A *an atom to be deleted. Then DDB' is a result of algorithm 3 given DDB and* A *as inputs, iff DDB' is a maxichoice contraction of* A *from DDB, satisfying the postulates (KB—1) to (KB—5) and (KB—6.1).*

Proof: From lemma 2, it is clear that steps 1 and 2 generate all DDB-closed locally minimal abductive explanations for A wrt IDB_G and some additional sets that contain a DDB-closed locally minimal abductive explanation for A wrt IDB_G. Steps 3 and 4 clearly computes an inclusion-minimal hitting set for this. The required result now follows from lemma 3 and theorem 5. ■

Algorithm 3 Maxi-choice contraction for definite deductive databases

Input: a definite deductive database $DDB = IDB \cup EDB$ and an atom A

Output: A maxi-choice contraction of A from DDB.

begin

1. Construct a *complete* SLD-tree for $\leftarrow A$ wrt DDB.
2. For every *successful* branch i: construct $\Delta_i = \{C \mid C \in EDB$ and C is used as an input clause in branch $i\}$. Let there be m such sets.
3. Let $D^* = \{\{C_1, \ldots, C_m\} \mid C_i \in \Delta_i\}$
4. Let D be a inclusion-minimal set among D^*, i.e. $\nexists D' \in D^*$ s.t. $D' \subset D$.
5. Produce $DDB \backslash D$ as the result.

end.

Interestingly, Tomasic presents a similar algorithm [T] to compute view deletions for definite deductive databases. While, Tomasic arrives at his algorithm *intuitively*, we have *methodically* obtained this algorithm starting from rationality considerations. Since, Tomasic computes maxichoice contractions, his algorithm is fully in accordance with our axiomatisation.

Now we proceed to present Algorithm 4 to compute generalised contraction for definite deductive databases. As observed before, this is not possible in general, but for a restricted case of acyclic programs.

Algorithm 4 generalised contraction for acyclic definite deductive databases

Input: an acyclic definite deductive database $DDB = IDB \cup EDB$ and an atom A

Output: a generalised contraction of A from DDB.

begin

1. Construct a complete SLD-tree for $\leftarrow A$ wrt DDB.
2. For every successful branch i: construct $\Delta_i = \{C \mid C \in EDB$ and C is used as an input clause in branch $i\}$.
3. Construct a hitting set D for all Δ_i's computed in the previous step.
4. Produce $DDB \backslash D$ as the result.

end.

The correctness of this algorithm follows from discussions before, and is formalised below.

Theorem 8 *Let DDB be an acyclic definite deductive database and A an atom to be deleted. Then DDB' is a result of algorithm 4 given DDB and A as inputs, iff DDB' is a generalised contraction of A from DDB, satisfying the postulates (KB—1) to (KB—5) and (KB—6.3).*

Proof: Follows from lemma 2, lemma 3, and theorem 6. ■

Algorithms 3 and 4 are inefficient, as they need to build a complete SLD-tree. Unfortunately, any rational algorithm for deletion can not avoid constructing complete SLD-trees. If these algorithms are changed to extract input clauses from incomplete SLD-derivations, then the new algorithm should check the derivability of an atom from a deductive database, before any deletions are carried out (otherwise, vacuity can not be satisfied). Checking derivability is also computationally expensive and more than that, weak relevance policy (KB—6.3) will *not* be satisfied in general. These points are illustrated by the following example.

Example 8 Consider the following definite database DDB:

$$\begin{array}{l} p \leftarrow q, r \\ q \leftarrow \end{array}$$

To delete p, suppose the algorithm does not construct a complete SLD-tree, then $(q \leftarrow)$ can be deleted. Since p does not follow from the current database, deleting $(q \leftarrow)$ violates the vacuity postulate. So, in order to avoid constructing a complete tree, any algorithm should check the derivability of p first. Consider another DDB:

$$\begin{array}{l} p \leftarrow t \\ p \leftarrow q, r \\ q \leftarrow s \\ r \leftarrow \\ t \leftarrow \end{array}$$

Again to delete p, without constructing a complete SLD-tree, the algorithm may delete both $(r \leftarrow)$ and $(t \leftarrow)$. Clearly, deletion of $(r \leftarrow)$ violates weak-relevance, as it is deleted from a failing branch. So, any rational algorithm must construct a complete SLD-tree. ■

In fact algorithms based on incomplete derivation trees have been presented by various researchers like Decker [De], Guessoum & Lloyd [GL1, GL2], Kakas & Mancarella [KM]. These works do not explicitly discuss the rationality behind these algorithms, and they do not comply with our axioms. Obviously, these algorithms are more efficient, but it is not clear what kind of relevance policy is satisfied by them.

Numerous algorithms have been proposed to update relational databases [DB, Ke, La], and it is not difficult to observe that algorithms by Dayal & Bernstein, Keller, and Langerak do comply with our axioms,though they do not explicitly discuss the rationality. In fact all these algorithms, with the exception of Dayal & Bernstein's, compute the maxichoice contraction. More discussion with formal theorems can be found in [A1].

Chapter 5

Concluding 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 generalisation 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 generalisation was achieved by introducing the concepts of knowledge base dynamics and generalised contraction for the same. Further, we also studied the relationship between knowledge base dynamics and abduction resulting in a general algorithm for contraction based on abductive procedures. We also successfully demonstrated how knowledge base dynamics can provide an axiomatic characterisation for deleting an atom from a definite deductive database.

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 generalised contraction 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 optimising the algorithms that would comply with the proposed declarative semantics. Annotation technique, proposed by Tomasic in [T], is in line with this, and can effectively improve the algorithms of the previous section. However, the declarative semantics of annotation has to be studied from the belief dynamics angle. We belief that notions like epistemic entrenchment [GM, LR], epistemic relevance [N2], order for safe contraction [AM], will be useful in this regard.

We believe that partial deduction [AD1, LS, and the references therein] and loop detection techniques [BAK, De, and the references therein], will play an important role in optimising algorithms of the previous section. Note that, loop detection could be carried out during partial deduction [AD1, B], 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 optimised query evaluation. This, however, opens up a new problem of dynamic partial deduction [BS], to be scrutinised further.

For technical simplicity, we have considered definite deductive databases & relational databases, and deletion of atoms only. The current framework can be extended for normal deductive databases, with insertion of atoms also. Further, extending this framework to disjunctive [GHLM, MD, NK] and extended logic programs [AP, ADP, PAA, PA, PDA] must be considered.

Though we have discussed only about view updates, we belief that knowledge base dynamics can also be applied to other applications such as view maintenance [GKM, GMS], diagnosis, and we plan to explore it further in our future works. It would also be interesting to study how results from other models of belief dynamics, especially the relational approach of Lindström and Rabinowicz [LR], could be applied in real world problems.

In [K], Kowalski proposes a related notion of knowledge assimilation, and argues that this new notion can provide an alternate account of the semantics of logic, with the implication that present conventional model theory is not necessary. It is important and useful to study how the results of this paper can be understood in Kowalski's knowledge assimilation framework.

Acknowledgements

We express our gratitude to H.N.Phien for his constant support and encouragements. It was S.O.Hansson who pointed out the connection between generalised and kernel contractions, and we are grateful to him for that. Encouraging and useful comments received from Decker, Hansson, Nebel, Rabinowicz, and the anonymous referees of JELIA'94 and JANCL are also gratefully acknowledged.

Bibliography

[A] Abiteboul,S., *Updates: A new Frontier*, In: M.Gyssens, J.Paredaens, and D.Van Gucht (eds.), Proc. of the second international conference on database theory, LNCS 326, Springer-Verlag, 1988, pp. 1–18.

[AGM] Alchourrón,C.E., Gärdenfors,P., and Makinson,D., *On the logic of theory change: Partial meet contraction and revision functions*, The Journal of Symbolic Logic 50 Number 2 (1985) 510–530.

[AM] Alchourrón,C.E., and Makinson,D., *On the logic of theory change: Safe contraction*, Studia Logica 44:405–422 (1985).

[AP] Alferes,J.J. and Pereira,L.M., *Contradiction: when avoidance equals removal*, In: R. Dyckhoff (ed.), 4th International Workshop on Extensions of Logic Programming, LNAI vol. 798, Springer-Verlag, 1994.

[ADP] Alferes,J.J., Damásio,C.V., and Pereira,L.M., *A logic programming system for non-monotonic reasoning*, Technical Report, U. Nova de Lisboa, Portugal, 1994.

[AB] Apt,K.R. and Bezem,M., *Acyclic programs (extended abstract)*, In: Proc. of the Seventh International Conference on Logic Programming, MIT Press, 1990, pp. 617–633.

[AD1] Aravindan,C. and Dung,P.M., *Partial deduction of logic programs wrt well-founded semantics*, New Generation Computing 13 (1994) 45–74. (a short version of this paper appears in: H.Kirchner and G.Levi (eds.), Proc. of the 3rd International Conf. on Algebraic and Logic Programming, LNCS 632, Springer-Verlag, 1992, pp. 384–402)

[AD2] Aravindan,C. and Dung,P.M., *Relationship between Gärdenfors-Makinson's epistemic entrenchment and Katsuno-Mendelzon's faithful total pre-orders*, In: R.Sadananda (Ed.), Proc. of the International Conference on Expert Systems Development, IEEE, March 1994, pp. 182–186.

[AD3] Aravindan,C. and Dung,P.M., *Belief dynamics, abduction, and databases*, In: C.MacNish, D.Pearce, and L.M.Pereira (eds.), Proc. of fourth European Workshop on Logics in AI (JELIA '94), LNAI Vol. 838, Springer-Verlag, 1994, pp. 66–85.

[AD4] Aravindan,C. and Dung,P.M., *Knowledge base dynamics, Abduction, and Database Updates*, Journal of Applied Non-Classical Logics, Vol. 5 No. 1 (1995) 51–76.

[A1] Aravindan,C., *Dynamics of belief: Epistemology, Abduction, and Database Updates*, PhD Thesis, Computer Science Program, Asian Institute of Technology, Bangkok, Thailand, 1995.

[BS] Benkerimi,K. and Shepherdson,J.C., *Partial deduction of updatable definite logic programs*, The Journal of Logic Programming 18 (1994) 1–26.

[BCL] Blakeley,J.A., Coburn,N., and Larson,P., *Updating derived relations: Detecting irrelevant and autonomously computable updates*, ACM Transactions on Database Systems 14:3 (1989) 369–400.

[BAK] Bol,R.N., Apt,K.R., and Klop,J.W., *An analysis of loop checking mechanisms for logic programs*, Theoretical Computer Science 86 No. 1 (1991) 35–79.

[B] Bol,R.N., *Loop checking in partial deduction*, The Journal of Logic Programming 16 No. 1&2 (1993) 25–46.

[BV] Brosda,V. and Vossen,G., *Update and retrieval in a relational database through a universal schema interface*, ACM Transactions on Database Systems 13:4 (1988) 449–485.

[Br] Bry,F., *Intensional Updates: Abduction via deduction*, In: Proc. of the 7th International Conf. on Logic Programming, 1990.

[DB] Dayal,U. and Bernstein,P.A., *On the correct translation of update operations on relational views*, ACM Transactions on Database Systems 8:3 (1982) 381–416.

[D] Decker,H., *Drawing updates from derivations*, Technical Report IR-KB-65, ECRC, Germany, September 1989.

[DDe] Denecker,M. and de Schreye,D., *SLDNFA: an abductive procedure for normal abductive programs*, Technical Report, Department of Computer Science, K.U. Leuven, Belgium. (available on internet from LPNMR archive <ftp.ms.uky.edu>)

[De] Demolombe,R., *A strategy for the computation of conditional answers*, In: Proc. of ECAI'92, 1992.

[Du] Dung,P.M., *An argumentation theoretic framework for Logic Programming*, To appear in: The Journal of Logic Programming.

[FKUV] Fagin,R., Kuper,G.M., Ullman,J.D., and Vardi,M.Y., *Updating logical databases*, In: Advances in Computing Research, Volume 3, Jai Press Inc., 1986, pp. 1–18.

[F] Fuhrmann,A., *Theory contraction through base contraction*, Journal of Philosophical Logic 20 (1991) 175–203.

[GM] Gärdenfors,P. and Makinson,D., *Revisions of knowledge systems using epistemic entrenchment*, In: M.Y.Vardi (ed.), Proc. of the second conf. on Theoretical aspects of Reasoning about Knowledge, Morgan Kaufmann, 1988, pp. 83–95.

[G] Gärdenfors,P., *Belief Revision: An Introduction*, In: P.Gärdenfors (ed.), Belief Revision, Cambridge University Press, 1992, pp. 1–28.

[GR] Gärdenfors,P. and Rott,H., *Belief Revision*, To appear in: Handbook of logic in AI and logic programming, Vol. IV: Epistemic and Temporal Reasoning.

[GHLM] Grant,J., Horty,J., Lobo,J., and Minker,J., *View updates in stratified disjunctive databases*, Journal of Automated Reasoning 11 (1993) 249–267.

[Gr] Grove,A., *Two modellings for theory change*, Journal of Philosophical Logic 17 (1988) 157–170.

[GL1] Guessoum,A. and Lloyd,J.W., *Updating Knowledge Bases*, Technical Report TR-89-05, Department of computer science, University of Bristol, U.K., December 1989.

[GL2] Guessoum,A. and Lloyd,J.W., *Updating Knowledge Bases II*, Technical Report TR-90-13, Department of computer science, University of Bristol, U.K., May 1990.

[GKM] Gupta,A., Katiyar,D., and Mumick,I.S., *Counting solutions to the view maintenance problem*, Technical Report ATT-DB-92-8, AT&T Bell Labs., U.S.A., 1992.

[GMS] Gupta,A., Mumick,I.S., and Subrahmanian,V.S., *Maintaining views incrementally (Extended abstract)*, In: Proc. of the International Conf. on Management of Data, ACM, 1993, pp. 157–166.

[H1] Hansson,S.O., *Belief contraction without recovery*, Studia Logica 50(2):251–260 (1991).

[H2] Hansson,S.O., *Belief base dynamics*, Doctoral dissertation, Uppsala university, Sweden, 1991.

[H3] Hansson,S.O., *In defense of base contraction*, Synthese 91:239–245 (1992).

[H4] Hansson,S.O., *Bridging a gap between AI research and philosophy*, In: E.Sandewall, and C.G.Jansson (eds.), Proc. of the Scandinavian conf. on Artificial Intelligence '93, IOS Press, 1993, pp. 1–9.

[H5] Hansson,S.O., *Theory contraction and base contraction unified*, Journal of Symbolic Logic, in press.

[H6] Hansson,S.O., *Kernel Contraction*, Journal of Symbolic Logic, in press.

[KaKT] Kakas,A.C., Kowalski,R.A., and Toni,F., *Abductive logic programming*, Journal of Logic and Computation 2 (1992) 719–770.

[KaM] Kakas,A.C., and Mancarella,P., *Database updates through abduction*, Technical Report, Department of Computing, Imperial College, London, U.K., 1990.

[KM] Katsuno,H. and Mendelzon,A.O., *Propositional knowledge base revision and minimal change*, Artificial Intelligence 52 (1991) 263–294.

[Ke] Keller,A.M., *Algorithms for translating view updates to database updates for views involving selections, projections, and joins*, In: Proc. of the 4th ACM Symposium on Principles of Database Systems, ACM, 1985, pp. 154–163.

[K] Kowalski,R., *Logic without model theory*, Technical Report, Department of Computing, Imperial College, London, U.K., 1994. (available on Internet from LPNMR Archive <ftp.ms.uky.edu>)

[La] Langerak,R., *View updates in relational databases with an independent scheme*, ACM Transactions on Database Systems 15 No. 1 (1990) 40–66.

[LR] Lindström,S. and Rabinowicz,W., *Epistemic entrenchment with incomparabilities and relational belief revision*, In: A.Fuhrmann and M.Morreau (eds.), Proc. of the Workshop on The logic of theory change, LNAI 465, Springer-Verlag, 1991, pp. 93–126.

[L] Lloyd,J.W., *Foundations of logic programming*, Second extended edition, Springer-Verlag, 1987.

[LS] Lloyd,J.W., and Shepherdson,J.C., *Partial evaluation in logic programming*, Technical Report No. CS-87-09, University of Bristol, U.K., 1987.

[Ma] Maier,D., *The theory of relational databases*, Computer Science Press, 1983.

[M1] Makinson,D., *How to give it up: A survey of some formal aspects of the logic of theory change*, Synthese 62 (1985) 347–363.

[M2] Makinson,D., *On the status of the postulate of recovery in the logic of theory change*, Journal of Philosophical Logic 16 (1987) 383–394.

[MD] Muthukaruppan,A. and Dung,P.M., *Updating disjunctive databases*, Technical Report, Computer Science Program, Asian Institute of Technology, Bangkok, Thailand, 1992.

[NK] Naqvi,S. and Krishnamurthy,R., *Database updates in logic programming*, ACM Press, 1988, pp. 251–262.

[N1] Nebel,B., *A knowledge level analysis of belief revision*, In: R. J. Brachman, H.J.Levesque, and R.Reiter (eds.), Proc. of the first international conference on principles of knowledge representation and reasoning, Morgan Kaufmann, 1989, pp. 301–311.

[N2] Nebel,B., *Belief revision and default reasoning: Syntax-based approaches*, In: J.A.Allen, R.Fikes, and E.Sandewall (eds.), Proc. of the second international conference on Principles of knowledge representation and reasoning, Morgan Kaufmann, 1991, pp. 417–428.

[PCA] Pereira,L.M., Calejo,M., and Aparício,J.N., *Refining knowledge base updates*, In: Proc. of the 7th Brazilian Symposium on Artificial Intelligence, November, 1990.

[PAA] Pereira,L.M., Alferes,J.J., and Aparício,J.N., *Contradiction removal within well-founded semantics*, In: A.Nerode, W.Marek, and V.S.Subrahmanian (eds.), Proc. of the 1st international workshop on LPNMR, MIT Press, 1991, pp. 105–119.

[PA] Pereira,L.M., and Alferes,J.J., *Discretionary reasoning with scenario semantics*, Technical Report, U. Nova de Lisboa, Portugal.

[PDA] Pereira,L.M., Damásio,C.V., and Alferes,J.J., *Diagnosis and debugging as contradiction removal*, In: L.M.Pereira and A.Nerode (eds.), Proc. of the 2nd international workshop on LPNMR, MIT Press, 1993, pp. 316–330.

[R] Reiter,R., *A theory of diagnosis from first principles*, Artificial Intelligence 32, pp. 57–95. (Also in: Ginsberg,M.L. (ed.), Readings in non-monotonic reasoning, Morgan Kaufmann, 1987, pp. 352–371)

[SK] Sadri,F. and Kowalski,R., *A theorem proving approach to database integrity*, In: J.Minker (ed.), Foundations of deductive databases and logic programming, Morgan Kaufmann, 1988, pp. 313–362.

[T] Tomasic,A., *View update translation via deduction and annotation*, In: M. Gyssens, J. Paredaens, and D. Van Gucht (eds.), Proc. of ICDT '88, LNCS 326, Springer-Verlag, 1988, pp. 338–352.

Appendix A

Acyclic Programs

Acyclic programs were introduced and studied by Apt and Bezem in [AB].

Definition 7 Let P be a normal logic program and B_P its Herbrand Base. A *level mapping* for P is a function $\|: B_P \to \mathbb{N}$ of ground atoms to natural numbers. The mapping $\|$ is also extended to ground literals by assigning $|\neg A| = |A|$ for all ground atoms $A \in B_P$. For every ground literal L, $|L|$ is called as the *level* of L in P. ■

Definition 8 Let P be a normal logic program and $\|$ a level mapping for P. P is called as *acyclic wrt* $\|$ if for every ground clause $A \leftarrow L_1, \ldots, L_n$ $(n \geq 0)$ in the ground instantiation of P, the level of A is higher than the level of every L_i $(1 \leq i \leq n)$. Moreover, P is said to be *acyclic* if it is acyclic wrt some level mapping for P. ■

Some important properties of acyclic programs are summarised by the following theorem.

Theorem 9 *Let P be an acyclic program. Then we have:*

1. T_P *has a unique fixpoint* M_P.
2. M_P *is the perfect model of* P.
3. M_P *is the unique Herbrand model of* $\mathrm{comp}(P)$.
4. M_P *is the unique fixpoint of Fitting's operator* Φ_P.
5. $\forall$ *variable-free atoms* A, $A \in M_P$ *iff* $\mathrm{comp}(P) \cup DCA \models A$.
6. $\forall$ *variable-free atoms* A *that do not flounder,* $A \in M_P$ *iff* $\exists$ *an SLDNF-refutation of* $P \cup \{\leftarrow A\}$.
7. $\forall$ *variable-free atoms* A *that do not flounder,* $A \in M_P$ *iff* $\exists$ *an SLS-refutation of* $P \cup \{\leftarrow A\}$.
8. M_P *is recursive.* ■

In section 4.1, definite acyclic program was used at length and an algorithm to compute generalised contraction for definite acyclic programs, that represent deductive or relational databases, was presented. Some restrictions were enforced on the database and are recalled below.

The EDB is a set of *ground facts*. The IDB is a set of definite program clauses and does not contain any unit clauses. No predicate defined in the database appears both in EDB and IDB. When an acyclic program represents a DDB, all EDB atoms are at level 0 and all IDB atoms are strictly at a higher level.

In order to prove part (3) of lemma 2, we first provide the notion of dependency graph.

Definition 9 Let P be an acyclic program. The *dependency graph* of P is the directed graph in which the nodes are the ground atoms from its Herbrand base B_P and there is a directed arc from node p to node q if there exists a clause in ground instantiation of P, in which p is the head and q appears in the body. A ground atom p is said to *depend upon* another ground atom q wrt P iff there is a path from node p to node q in the dependency graph of P. ■

For a definite acyclic DDB, it is not difficult to see that the dependency graph of DDB forms a forest of trees. We are now in a position to provide proof of lemma 2.

Proof of lemma 2: Let Δ be generated from a successful branch i of a SLD-tree for $\leftarrow A$ wrt DDB. For every EDB fact $q \in \Delta$ we show the following proposition: q appears in a locally minimal abductive explanation for A wrt IDB_G. If this is shown, then the required result follows immediately.

First of all observe that there exists a path from a ground instantiation p of A to q in the dependency graph of DDB. More specifically, consider a path that is based only on the clauses used in refutation i. By induction, we show that for every ground IDB atom in the path, q appears in a locally minimal abductive explanation of that atom.

The base case is when the length of path from p to q is one, i.e. there exists a directed arc from p to q. Consider the ground instantiation C_G of C, the clause that corresponds to that arc, with p as its head and q in the body. Let $p_1, \ldots, p_n$ be the other atoms in the body, and X a inclusion minimal subset of DDB_G that derives all of them. It is clear that such a X exists since we are considering a successful refutation i. Since DDB is acyclic and definite, $X \cup \{C_G, q\}$ is a A-kernel of DDB_G and consequently, from lemma 1, q is in a locally minimal abductive explanation for A.

For the induction, assuming that our proposition holds for all atoms in tha path with distance I from q, we have to show that it holds for every atom p at distance I+1 also. Consider again the clause C_G that corresponds to the arc out of p. Let it be of the form $p \leftarrow p_1, \ldots, p_n$. Some p_j in the body appears in the path from p to q, i.e. there is a directed arc from p_j to q of length I. So, by induction assumption, there exists a locally minimal abductive explanation Δ for p_j that contains q. For every other atom $p_k \in C_G$ consider Δ_k, a locally minimal abductive explanation for p_k that differs minimally wrt Δ. By this we mean that, among all locally minimal abductive explanation for p_k, $\Delta_k \backslash \Delta \cup \Delta \backslash \Delta_k$ is minimal wrt set-inclusion. Since we consider definite acyclic databases, it is not difficult to see that $\bigcup p_k \cup \Delta \cup \{C_G\}$ is a A-kernel for DDB_G. From lemma 1, it follows that q is in a locally minimal abductive explanation for A. ■