Belief Revision:
A Computational Approach

by

Simon Edmund Dixon

A thesis submitted in fulfilment of
the requirements for the degree of
Doctor of Philosophy

Basser Department
of Computer Science

University of Sydney

January, 1994

ABSTRACT

In order to reason about the real world, intelligent systems must have ways of dealing with incomplete and inconsistent information. This issue is addressed in the study of nonmonotonic reasoning, truth maintenance and database update. Belief revision is a formal approach which is central to these areas. This thesis describes a computational approach to the logic of belief revision developed by Alchourrón, Gärdenfors and Makinson (AGM), culminating in the first implementation of an AGM belief revision system. The implementation is based on classical first-order logic, and for any finitely representable belief state, it efficiently computes expansions, contractions and revisions satisfying the AGM postulates for rational belief change. The epistemic state is represented by a finite entrenchment base, from which the full belief set may be derived by logical closure, and the entrenchment relation can be generated via the construction of a unique most conservative entrenchment. The entrenchment construction is motivated by considerations of evidence and by connections with truth maintenance and nonmonotonic reasoning. A minimal change policy is presented as the solution to the entrenchment revision problem. The belief change algorithms and design decisions are described in detail, with examples of the system's operation on some standard problems in the AI literature. Two extensions of the system are also described: firstly, an entrenchment generation algorithm which allows the AGM system to simulate the behaviour of the assumption-based truth maintenance system (ATMS); and secondly, a modification to the most conservative entrenchment in order to implement nonmonotonic reasoning using defaults with exceptions.

ACKNOWLEDGEMENTS

I wish to thank my supervisor, Norman Foo, for getting me started in belief revision and providing direction and ideas all along the way; Wayne Wobcke, my associate supervisor, for ironing out many technical flaws in my work; the rest of the Knowledge Systems Group, particularly Maryanne Williams and Pavlos Peppas for their suggestions and criticism, and Maurice Pagnucco for his library of papers and his proof-reading.

Recreational breaks on campus (squash, bridge and music) were provided by Dean, Groo, Paul, Morri, Mike, Rex, and Boon Toh. These either preserved my sanity or impeded my progress. Thanks! The undergraduates who experienced my tutorials also provided many hours of distraction from research (mostly enjoyable), as did the other visitors to the Knowledge Systems Laboratory. Telephone calls (and occasional visits) were courtesy of Leonie and Nicola.

Off-campus distractions were organised by The White Boys (blues), Cross Street (a cappella), and The Remnant (rock). Mudcake was supplied by Exclusive Cheesecakes, kebabs by Rowda's, and nachos by Dean's. Wisdom was supplied by Andrew Kitchen, Bryen Willems and Mark Maguire. A special mention must go to Mike Howell, who is my only friend who thought I should not do a PhD (and had the courage to say so). Other friends cannot go unmentioned: Adrian, Alvin, Annie, Antony, Ben, Dave, John, Katrin, M.C., Mick, Ming, Paul, Susan and Tim.

My family were always willing to help in any way they could, and their love is very much appreciated.

This work was supported financially by an Australian Postgraduate Research Award, plus other grants, travel allowances, and scholarships from the Australian Research Council, IJCAI Inc., the Knowledge Systems Group, Computer Science Department, and the University of Sydney.

"Of making many books there is no end, and much study wearies the body." – Ecclesiastes 12:12

CONTENTS

APPENDICES

1 Introduction

Artificial intelligence (AI) has been defined as "the study of ideas that enable computers to be intelligent" [Winston 1984] and "the study of intelligent behaviour" [Genesereth & Nilsson 1987]. These definitions are circular, in that they rely on an implicit understanding of the nature of intelligence itself. Computer systems have been built which are supposed to display intelligent behaviour, but how can such behaviour be categorised or recognised as intelligent?

One definition of intelligence is "the capacity to acquire and apply knowledge", and it is with this definition in mind that we introduce the topic of this thesis: *belief revision*. The study of belief revision is of central concern in artificial intelligence, as it involves the rational acquisition and application of knowledge in reasoning about the world. In philosophy, belief revision is part of the topic of *epistemology*, the study of the nature and origin of knowledge; although we are concerned with beliefs and their acceptance rather than knowledge and truth.

For a computer system to acquire knowledge successfully, there must be a suitable method of representing the knowledge so that it may be stored, retrieved and manipulated in an efficient manner. Many knowledge representation formalisms have been proposed, but there are other problems in knowledge acquisition apart from finding an adequate knowledge representation formalism. Our beliefs about the world can be incomplete, so that we do not know enough to come to the correct conclusion, either because we are unaware of certain facts, or else our beliefs are not sufficiently precise to distinguish between similar alternatives. A second problem, incorrect beliefs, can be caused by a faulty observation, an error in communication, or deceit on the part of the information source. Thirdly, beliefs can be inappropriate to a particular situation when we fail to observe an action or event that alters the world, or else we are unaware of the ramifications of the action or event, so that beliefs which were once correct no longer reflect the true state of affairs in the world.

Thus our beliefs about the world must constantly change as we acquire new information from more or less reliable sources. Hence the need for belief revision.

1.1 The Foundational and Coherence Theories of Justification

Before introducing belief revision, we present a description of the two main philosophical views of how knowledge may be justified [Pappas & Swain 1978]. These are best illustrated by two metaphors from [Sosa 1980]:

> "For the foundationalist every piece of knowledge stands at the apex of a pyramid that rests on stable and secure foundations whose stability and security does not derive from the upper stories or sections. For the coherentist a body of knowledge is a free-floating raft every plank of which helps directly or indirectly to keep all the others in place, and no plank of which would retain its status with no help from the others."

1.1.1 Static Epistemic States

In the foundational theory of justification, a proposition α is accepted (i.e. believed) if and only if:

1) α is self-evident, or
2) α can be derived from a set of accepted propositions.

Sentences fulfilling the first condition are called *foundational* beliefs. It is assumed that the acceptance of these beliefs is indisputable, since they form the basis of the whole epistemic state. Foundational beliefs usually correspond to "hard facts", such as observations about the physical world. This view (called radical foundationalism) is criticised in [Pastin 1978], where a weaker condition of self-warrant replaces the infallibility requirement for foundational beliefs. The assumption-based truth maintenance system [de Kleer 1986] operates along these lines, with the foundational beliefs being assumptions which can be added to or retracted from the belief set at any time.

Assuming a logic-based representation of our beliefs, then the sentences satisfying the second condition are those which are logical consequences of the foundational beliefs. Each belief has a justification (for logical representations this is a proof), using beliefs which are also justified, and their justifications must in turn be based on further justified beliefs; hence the belief set consists of propositions linked by chains of justifications. To conform with the above definition, these chains must end in the foundational beliefs; an infinite regression is not allowed, nor is any circular chain of justifications. Often we distinguish between the explicit (foundational) beliefs and the implicit (derived) beliefs, attaching a greater degree of importance to the explicit beliefs [Nebel 1990].

On the other hand, the coherence theory does not require beliefs to have a formal justification. Instead, propositions are accepted which improve the overall coherence of the belief set. [Cornman 1978] argues that the only rational non-foundational theory of justification is the coherence theory, which can be thought of as a generalisation of the foundational theory which allows cycles in the chains of justifications. Several types of coherence are identified in [Thagard 1989]: *deductive coherence* is based on the logical consistency of the belief set; *probabilistic coherence* depends on the propositions having probability assignments which are consistent with the axioms of probability; *semantic coherence* exists when propositions have similar meanings; and *explanatory coherence* occurs when there are consistent explanatory relationships between the propositions in the belief set. All except deductive coherence require some form of extralogical information in order to measure the coherence of a set of propositions. For this reason, deductive coherence is also known as *weak coherence*, whereas the other types are classed as *strong coherence* methods.

1.1.2 The Dynamics of Epistemic States

The two theories of justification usually relate to static belief states, but they are applied to belief revision in [Harman 1986] and [Gärdenfors 1990b]. According to the foundational theory, belief revision should consist of two steps: retracting beliefs which no longer have a satisfactory justification, and adding beliefs which now do have a satisfactory justification. In this sense the foundational theory is nonconservative, because the fact that a belief was held in the past has no bearing on its present status. Therefore, in foundationalism, the credibility of a belief is independent of the history of the belief set.

Conversely, the coherence theory provides a conservative approach to belief revision, by requiring a minimal change in the belief state that sufficiently increases the overall coherence of the beliefs. Thus the history of a belief set will determine much of its future contents.

It is natural to question at this point which approach is preferable. Intuitively, it is more rational to give up beliefs whose justifications have been discredited, and hence the foundational theory seems to model how beliefs ought to be revised. But Harman cites psychological studies which show that the coherence theory is closer in describing what people actually do when revising their beliefs. Summarising several such studies, [Ross & Anderson 1982] states: "It is clear that beliefs can survive ... the total destruction of their original evidential bases." The main reason for this is that people do not keep track of the justifications for their beliefs. But human behaviour does not necessarily bear upon a theory of rational belief revision. In [Dixon & Foo 1992b], we present examples which demonstrate the need for both styles of reasoning in a rational belief revision system. In particular, the coherence theory is useful for reasoning about actions or events whose effects persist beyond the duration of the action or event.

1.2 Introduction to AGM Belief Revision

The problem of belief revision can be simply stated as this: given a consistent belief state and some new information, how do we assimilate this new data into our belief set? Some basic requirements on this process are firstly that the new belief state is consistent; secondly, that the new information is reflected in the resulting state of belief; and thirdly, that any previous beliefs (unless contradicted) are preserved in the new belief state.

One formal approach to belief revision which satisfies these requirements has been developed by Alchourrón, Gärdenfors and Makinson [Alchourrón & Makinson 1982, 1985; Alchourrón *et al.* 1985; Gärdenfors 1988, 1990b; Gärdenfors & Makinson 1988; Makinson 1985, 1987], which has become known as the AGM approach to belief revision. In this approach, an idealised mathematical formalism is used for representing belief states and describing the ways in which these states change in the light of new information. There are three types of belief change in the AGM approach: expansion, contraction and revision. Expansion involves the acceptance of a new belief without giving up any previous beliefs, even if they are contradicted by the new information. Contraction is the converse of expansion; a belief is removed from the belief set and no new beliefs are added. The third operation, revision, adds a new belief to the belief set but also ensures that the resulting belief set is consistent, which may mean that some of the original beliefs must be given up.

Each AGM belief change operation is characterised by a set of *rationality postulates*; we shall describe a function which satisfies these postulates as a *(fully) rational* belief change operation. The postulates capture the basic requirements of belief change – the consistency of the resulting state, the acceptance of the new information, and the preservation of previous beliefs. That is, a rational belief change is defined to be the minimal change to the belief set which consistently incorporates new information into the belief set.

The AGM approach assumes a linguistic model of beliefs; each belief is expressed as a sentence in some (logical) language. Then a *belief set* is the collection of all the sentences in this language which we accept, or regard as certain. Also, a *belief state* is a belief set together with any metalogical epistemic attitudes, such as a preference for one belief over another. We shall use the terms *belief state* and *epistemic state* interchangeably. Also, the term *belief revision* is used in two senses: firstly, as a generic term equivalent to *belief change*; and secondly, as a specific belief change operation, as opposed to other AGM operations such as expansion and contraction. The appropriate meaning will be clear from the context.

At a first glance, the AGM postulates seem to define a coherence-based approach to belief revision, as the belief change operations compute the minimal changes to the belief set which are required to successfully assimilate the new information whilst retaining the logical consistency of the belief set; this satisfies the definition of weak coherence. But we show in chapter 6 that the extralogical information provided by the entrenchment relation is sufficient to obtain foundational behaviour from the AGM logic. Coherence-based behaviour can also be achieved by a different choice of entrenchment relation [Dixon & Foo 1992b]. Therefore, the AGM approach is sufficiently versatile for reasoning according to either the foundational or coherence theories of justification.

One idealisation of the AGM approach to belief revision is that the belief set is assumed to be logically closed. That is, the belief set contains all of its logical consequences. If the belief change functions are interpreted as modelling a rational agent, then this assumption corresponds to the agent being logically omniscient.

There are several constructive modellings of AGM belief change operations, based on a preference relation on the belief set (epistemic entrenchment and safety relations), a preference relation on maximal consistent sets of formulae in the language (systems of spheres), or a selection function on maximal consistent subsets of the belief set (transitively relational partial meet contraction and revision functions). All of these constructive modellings have the same problem that they do not present any finite algorithmic method of computing belief change operations.

Alternative formalisms using theory bases instead of belief sets have been proposed; these also use preference relations (comparative retractability, E-bases, most conservative entrenchments, ensconcements, database priorities, epistemic relevance and prioritized bases). Although these approaches can be used to define a finite representation for belief states, none define a computational approach to belief revision which satisfies the AGM rationality postulates. Similarly, the model-theoretic approaches to belief revision do not provide a computational account of belief revision.

1.3 Overview of the Thesis

The primary contribution of this thesis is the development of a complete computational approach to AGM belief revision, including representation formalisms, belief change algorithms, a working implementation, and the application of these algorithms to solve problems in fields outside of belief revision.

In chapter 2, we present a summary of some formal approaches to belief revision. The AGM belief revision framework is presented, in terms of the representation formalisms, a set of rationality postulates for belief change, and a number of constructive modellings for belief change operations which satisfy the AGM rationality postulates, such as epistemic entrenchment. Then several alternative approaches are presented which, for the sake of computational or representational efficiency, do not satisfy all of the AGM postulates. These methods are compared and contrasted with the AGM approach, identifying the strengths and weaknesses of each approach. We conclude that one deficiency is the lack of a computational model of AGM belief revision; the rest of the thesis addresses this issue.

In chapter 3, we describe our computational approach to AGM belief revision, addressing four main theoretical issues. The first issue is finding an efficient and clear representation of belief states which can be used in a computational setting. The second issue is defining a method of generating an AGM epistemic entrenchment relation from a finite representation. Thirdly, we discuss entrenchment revision, that is, how the entrenchment relation is modified by belief change operations. The policy adopted for entrenchment revision is based on the coherentist principle of minimal change, which is a basic tenet of the AGM paradigm. Finally, we discuss the relationship between coherentism and the (usually tacit) assumption of independence of beliefs, and describe its influence on the choice of belief revision policy.

The fourth chapter contains a set of algorithms used in computing AGM belief revision operations, based on the representations developed in chapter 3. We describe algorithms for determining rank (the finite representation of entrenchment), the three AGM operations (expansion, contraction and revision), plus some variations on these operations from the belief revision literature. Each algorithm is discussed in detail and illustrated with examples. We also develop formal correctness conditions for the operations which ensure that the minimal change entrenchment revision policy is followed, and formally prove that the algorithms conform to these specifications. Complexity considerations for the algorithms are also addressed in this chapter, and we show that if we choose a logic with a tractable decision procedure for derivability, then the belief revision operations will also be tractable.

Chapter 5 describes the implementation details of a first-order logic AGM belief revision system, including design decisions, data structures, the theorem proving procedure, the user interface, and several examples of its operation. The theorem prover is based on ordered linear resolution [Chang & Lee 1973], and is extended to handle the equality predicate by adding a paramodulation step which is equivalent to two ordered linear resolution steps. We also describe a method which ensures termination of the theorem proving procedure, but sacrifices the completeness of the decision procedure for derivability, for an identifiable class of cases. The examples demonstrate the use of the standard AGM operations (expansion, contraction and revision), as well as the queries of the belief set and the entrenchment relation, and a conditional query which is evaluated according to the Ramsey Test for conditionals [Ramsey 1931].

The next two chapters illustrate the application of belief revision to other forms of dynamic reasoning. In chapter 6, we discuss the relationship between belief revision and truth maintenance, showing that it is possible to define an entrenchment generation and revision algorithm which allows the AGM system to operate as a truth maintenance system. We give two algorithms for simulating the behaviour of the assumption-based truth maintenance system (ATMS) [de Kleer 1986], proving them correct relative to a functional specification of the ATMS. The ATMS was chosen because it is the most widely used dynamic reasoning system which is based on the foundational theory of justification, and this result demonstrates that it is possible to use the epistemic entrenchment relation to encode foundational information. This is contrary to the usual view of the AGM approach being based on the coherence theory of justification. The chapter concludes with a theoretical analysis of the general relationship between belief revision and foundational reasoning, and gives the conditions under which foundational reasoning can be performed using AGM belief revision.

The other application, to nonmonotonic reasoning, is presented in chapter 7, where the implementation of a nonmonotonic reasoning system based on the AGM system is described. The system reasons about defaults with exceptions, and uses a modified version of the revision algorithm from chapter 4. After discussing the theoretical issues and design decisions, we illustrate the use of the system on a number of benchmark problems from the nonmonotonic reasoning literature [Lifschitz 1989], and compare the system with other approaches to nonmonotonic reasoning.

The final chapter summarises the results contained in the thesis and outlines some directions for further research.

Some material in the thesis has been published; parts of chapters 3, 4 and 5 appear in [Dixon 1993] and [Dixon & Wobcke 1993], the first algorithm of chapter 6 is presented in [Dixon & Foo 1992a, 1993], and chapter 7 contains some joint work from [Dixon & Wobcke 1994]. Also, some of the discussion of chapters 2 and 6 first appeared in [Dixon & Foo 1992b].

2 Overview of Belief Revision Models

In this chapter we present a survey of the theory of belief revision in artificial intelligence. We will mainly focus on the AGM paradigm, which is defined by a set of rationality postulates [Alchourrón *et al.* 1985, Gärdenfors 1988], and is also characterised by several constructive modellings for expansion, contraction and revision. The postulates provide a non-constructive approach to belief revision, as they do not define a unique set of belief change operations. Some constructive approaches are presented, including epistemic entrenchment [Gärdenfors & Makinson 1988], systems of spheres [Grove 1988], safe contraction [Alchourrón & Makinson 1985] and conditional functions [Spohn 1988]. All of these approaches are non-computational, as they do not provide an algorithmic means of revising finite representations of belief states. Nevertheless, they provide a theoretical foundation for idealised rational belief change, to which a computational model should aspire. We now present a brief outline of AGM belief revision followed by several variants of the AGM approach to belief revision.

2.1 The AGM Belief Revision Paradigm

2.1.1 Belief Sets

The first issue which must be addressed in developing a theory of belief revision is the method of representing the beliefs of an agent. The most common method, particularly in computational settings, is by a set of sentences in a logical language *L*. The belief set is intended to represent the set of all sentences accepted by the agent, that is all sentences which are believed to be true. We shall denote the belief set by the symbol *K*.

Assume the logical language *L* contains the standard logical connectives: negation ($\neg$), conjunction ($\wedge$), disjunction ($\vee$) and material implication ($\rightarrow$); plus the two constants truth ($\top$) and falsity ($\perp$).

In a consistent belief state, there are three possible epistemic attitudes towards a sentence α:

(1) α is accepted ($\alpha \in K$)
(2) α is rejected ($\neg\alpha \in K$)
(3) α is indetermined ($\alpha \notin K$ and $\neg\alpha \notin K$)

A belief set in which α and $\neg\alpha$ are both accepted is inconsistent, and hence it is disallowed in modelling epistemic states. Also, for an agent to be ideally rational, it must believe all of the consequences of its beliefs. These criteria may not be suitable for modelling realistic agents, which do not have this property of logical omniscience, but they are useful in defining ideally rational belief sets.

Hence the definition of a belief set relies heavily on the logical consequence relation, which we shall denote by the symbol $\vdash$. In the AGM framework, the logic is assumed to contain classical propositional logic, satisfy the deduction theorem, and be compact. The set of all consequences of a set *K*, denoted $Cn(K)$, is defined by $\{\alpha : K \vdash \alpha\}$. The above two conditions can then be expressed by the following definition [Gärdenfors 1988]:

A set *K* of sentences is a (nonabsurd) *belief set* if and only if:
(1) $K \nvdash \perp$, and
(2) $K = Cn(K)$.

The *absurd* belief set is denoted $K_\perp$, and contains all sentences in the language. This definition of a belief set corresponds to the definition of a *theory* in logic, and nonabsurd belief sets correspond to consistent theories.

2.1.2 AGM Belief Change Operations

The AGM framework is characterised by three ways of changing the belief state: expansion, contraction and revision. Each of these operations models an idealised rational choice of beliefs in the light of new information which may or may not be consistent with current beliefs. There are six possible ways the epistemic attitude towards a sentence may change between the three attitudes *accepted*, *rejected*, and *indetermined*. Each of the AGM belief change operations captures two of these types.

2.1

In an expansion, the attitude towards a sentence α is changed from indetermined to accepted (expansion by α) or rejected (expansion by $\neg\alpha$). This operation is called expansion because either α or $\neg\alpha$ is added to the belief set K, and none of the old beliefs are retracted. Expansion of K by α is denoted K^+_α. [Gärdenfors 1988] shows that the minimal change operation which accepts a new belief without removing any of the old beliefs is given by:

$$K^+_\alpha = Cn(K \cup \{\alpha\}).$$

By this definition, the expansion succeeds regardless of whether or not the new belief α is consistent with K. That is, expansion is defined even when the epistemic attitude towards the new sentence is already determined. If $\neg\alpha \in K$, then the expansion K^+_α results in the inconsistent belief set $K_\perp$; otherwise the expansion is guaranteed to return a consistent minimal belief set containing K and α.

The second type of epistemic change is contraction, which occurs when the attitude towards a belief α changes from accepted (contraction by α) or rejected (contraction by $\neg\alpha$) to indetermined. No new sentences are added to the belief set, so the contracted belief set is always a subset of the initial belief set K. The contraction of a belief set K by α is denoted K^-_α. Unfortunately, there is no definition of a unique contraction operation in purely logical and set-theoretic terms, since there is a choice of which belief or beliefs to retract when α is derivable from a combination of other beliefs. In the next subsection, we describe the postulates for a rational contraction operation, and then in the following section show how a preference relation can be used to define a unique contraction operation.

The revision operation covers the final two types of epistemic change, when the attitude towards α changes from accepted to rejected (revision by $\neg\alpha$) or from rejected to accepted (revision by α). The revision operation is defined with respect to the belief which is newly accepted, so that K^*_α denotes the acceptance of the previously rejected sentence α. In the case where $\neg\alpha \notin K$, the revision collapses to an expansion operation, and when $\alpha \in K$, it is the trivial expansion, leaving K unchanged. Like contraction, a unique revision operation cannot be defined without using a selection function or preference relation, which we shall describe in the following section.

2.1.3 Rationality Postulates

The following postulates for contraction and revision [Alchourrón *et al.* 1985] define *fully rational* contraction and revision functions. The first six postulates for each operation are known as the basic postulates, and the remaining two as the supplementary postulates. The postulates for contraction are as follows:

(K-1) $K_{\alpha}^{-} = Cn(K_{\alpha}^{-})$

(K-2) $K_{\alpha}^{-} \subseteq K$

(K-3) If $\alpha \notin K$ then $K_{\alpha}^{-} = K$

(K-4) If $\nvdash \alpha$ then $\alpha \notin K_{\alpha}^{-}$

(K-5) $K \subseteq (K_{\alpha}^{-})_{\alpha}^{+}$

(K-6) If $\vdash \alpha \leftrightarrow \beta$ then $K_{\alpha}^{-} = K_{\beta}^{-}$

(K-7) $K_{\alpha}^{-} \cap K_{\beta}^{-} \subseteq K_{\alpha\wedge\beta}^{-}$

(K-8) If $\alpha \notin K_{\alpha\wedge\beta}^{-}$ then $K_{\alpha\wedge\beta}^{-} \subseteq K_{\alpha}^{-}$

The first postulate requires the contraction operation to preserve logical closure; the second postulate ensures that no new sentences are added to the belief set; and (K-3) defines the trivial contraction – the belief set is unchanged by an attempt to retract a sentence which does not occur in the belief set. Postulate (K-4), the "success postulate", states that a retracted belief cannot remain in the belief set, unless it is a logical theorem. (K-5), known as the "recovery postulate", defines a notion of minimal change, whereby all beliefs removed by the contraction of α can be replaced by a subsequent expansion by α. This postulate has received some criticism [Makinson 1987, Fuhrmann 1991, Niederée 1991, Hansson 1991] but it appears to be suitable for the idealised situation of reasoning with closed theories, if not for their finite counterparts. The sixth postulate ensures that contractions by logically equivalent formulae yield the same results. The supplementary postulates define relationships between $K_{\alpha\wedge\beta}^{-}$ and K_{α}^{-}. (K-7) states that retracting the sentence $\alpha \wedge \beta$ from K cannot remove any more sentences than are removed by at least one of the contractions K_{α}^{-} and K_{β}^{-}. In other words, the beliefs remaining in both K_{α}^{-} and K_{β}^{-} must be contained in $K_{\alpha\wedge\beta}^{-}$. Finally, (K-8) is a conditional converse of (K-7); if the removal of $\alpha \wedge \beta$ forces α to be removed, then the removal of $\alpha \wedge \beta$ can be no stronger than the removal of the logically weaker sentence α. The two supplementary postulates add to the concept of minimal change defined by the basic postulates.

The following are the postulates for revision:

(K*1) $K^*_\alpha = Cn(K^*_\alpha)$
(K*2) $\alpha \in K^*_\alpha$
(K*3) $K^*_\alpha \subseteq K^+_\alpha$
(K*4) If $\neg\alpha \notin K$ then $K^+_\alpha \subseteq K^*_\alpha$
(K*5) $K^*_\alpha = K_\perp$ only if $\vdash \neg\alpha$
(K*6) If $\vdash \alpha \leftrightarrow \beta$ then $K^*_\alpha = K^*_\beta$
(K*7) $K^*_{\alpha\wedge\beta} \subseteq (K^*_\alpha)^+_\beta$
(K*8) If $\neg\beta \notin K^*_\alpha$ then $(K^*_\alpha)^+_\beta \subseteq K^*_{\alpha\wedge\beta}$

The revision operation also preserves logical closure (K*1); it must also be successful, that is, the sentence α by which we are revising the belief set K must always appear in the revised belief set K^*_α, by postulate (K*2). Thirdly, no extra sentences are added to the belief set except those which are logical consequences of the new sentence α and the belief set K. By (K*4), the converse applies for a revision by a sentence α which is consistent with K, so that no sentences are removed from the belief set, in which case the revision is equivalent to an expansion. (K*5) ensures that revision always results in a consistent belief set, unless the operation specifically introduces a logically inconsistent sentence into the belief set. Like contraction, revision respects logical equivalence, by (K*6). Finally, the supplementary postulates define a relationship between the revisions by the logically dependent formulae α and $\alpha \wedge \beta$. The justification for these postulates is that the minimal change required to accommodate both α and β in K should be the same as the expansion of K^*_α by β, in the case where the expansion is consistent, that is, $\neg\beta \notin K^*_\alpha$. This relationship is split into two containment conditions. The consistency check is not necessary in (K*7), since if $\neg\beta \in K^*_\alpha$, then $(K^*_\alpha)^+_\beta = K_\perp$, which trivially contains all other belief sets.

The rationality postulates for revision bear a very close resemblance to those for contraction. One reason for this is that the two operations are interdefinable by the following identities:

Levi identity: $K^*_\alpha = (K^-_{\neg\alpha})^+_\alpha$
Harper identity: $K^-_\alpha = K \cap K^*_{\neg\alpha}$

It has been shown [Alchourrón *et al.* 1985], that if a contraction function satisfies the first six postulates for contraction, (K-1) – (K-6), then the revision operation defined by the Levi identity satisfies postulates (K*1) – (K*6). Furthermore, if the contraction function also satisfies (K-7), then the revision satisfies (K*7), and similarly if all eight contraction postulates are satisfied, then the corresponding revision function satisfies all eight postulates for revision. The converse also holds for contraction functions defined by revision functions via the Harper identity – the first six, seven or eight contraction postulates are satisfied by the resulting contraction operation when the revision operation satisfies the first six, seven or eight (respectively) revision postulates.

2.2 Constructive Modellings for AGM Belief Revision

2.2.1 Epistemic Entrenchment

As stated previously, logical and set-theoretic postulates are not sufficient to define unique contraction or revision operations. The postulates define the class of rational belief change functions, but do not provide any way of choosing a particular revision or contraction function. To choose between the various belief change operations, the AGM model uses extralogical information in the form of an epistemic entrenchment relation [Gärdenfors & Makinson 1988].

The basis for the concept of epistemic entrenchment is that some propositions in a belief set are more useful or more important than others for reasoning. We shall say that such sentences have a higher degree of epistemic entrenchment, and assume that this will influence our choice of contraction and revision functions. If this entrenchment relation is to be useful, we must be able to determine the relation independently of the behaviour of a particular contraction or revision operation. Then the belief change operation can be defined from the entrenchment relation such that the sentences in the belief set with lower entrenchments are given up in preference to sentences with higher entrenchments.

Degrees of entrenchment are not measured quantitatively, but only relative to other degrees of entrenchment. For any two sentences α and β in L, we define the following symbols for the relation $\leq_E$: $\alpha \leq_E \beta$ denotes that β is at least as epistemically entrenched as α; $\alpha <_E \beta$ if and only if $\alpha \leq_E \beta$ and not $\beta \leq_E \alpha$; and $\alpha =_E \beta$ if and only if $\alpha \leq_E \beta$ and $\beta \leq_E \alpha$. We will sometimes use the term *ordering* to describe the entrenchment relation, although strictly speaking the relation is a *total pre-order* on the sentences in L. The entrenchment relation obeys the following postulates for all α, β, $\gamma \in L$:

(EE1) If $\alpha \leq_E \beta$ and $\beta \leq_E \gamma$ then $\alpha \leq_E \gamma$
(EE2) If $\alpha \vdash \beta$ then $\alpha \leq_E \beta$
(EE3) For any α and β, $\alpha \leq_E \alpha\wedge\beta$ or $\beta \leq_E \alpha\wedge\beta$
(EE4) When $K \neq K_\perp$, $\alpha \notin K$ iff $\alpha \leq_E \beta$ for all β
(EE5) If $\beta \leq_E \alpha$ for all β then $\vdash \alpha$

The first condition is that the relation is transitive; the second, called the "dominance" postulate, is motivated by considering contracting a belief set K containing α and β by either of the two sentences, and noting that it is a smaller change to give up α and retain β than to give up β, in which case α must also be removed if the contraction is to obey the rationality postulates. For (EE3), we note that the contraction of $\alpha\wedge\beta$ from a belief set containing this sentence forces the removal of at least one of α and β. Note also that the combination of (EE2) and (EE3) implies that the ordering is connected, that is, for all α and β, either $\alpha \leq_E \beta$ or $\beta \leq_E \alpha$. (EE4) provides a minimality condition: all sentences outside K have the least entrenchment, and no other sentences have this same entrenchment, except in the case where K is inconsistent, and therefore there are no sentences outside of K. Finally (EE5) is the upper limit on the entrenchment relation: those sentences which are maximal in the relation are logical theorems. Conversely, by (EE2), all logical theorems are maximal in the entrenchment ordering.

For the moment, we shall assume a given entrenchment relation $\leq_E$ which satisfies (EE1) – (EE5). This relation depends on K, since (EE4) implies that different belief sets have different epistemic entrenchments. It is possible, however, for one belief set to have different entrenchments on its members at different times.

Now we turn to the issue of constructing contraction and revision functions from the entrenchment relation. [Gärdenfors & Makinson 1988] shows that contraction functions and epistemic entrenchment relations are interdefinable via the following two conditions:

(C–) $\beta \in K^-_\alpha$ iff $\beta \in K$ and either $\vdash \alpha$ or $\alpha <_E \alpha \vee \beta$
(C$\leq_E$) $\alpha \leq_E \beta$ iff $\alpha \notin K^-_{\alpha\wedge\beta}$ or $\vdash \alpha\wedge\beta$

Two representation theorems are then proved, to show the equivalence of the different representations. These state that if an ordering $\leq_E$ satisfies (EE1) – (EE5), then the contraction function which is uniquely determined by (C–) satisfies (K-1) – (K-8) and also condition (C$\leq_E$); and, conversely, if a contraction function satisfies (K-1) – (K-8) then the ordering $\leq_E$ that is uniquely determined by (C$\leq_E$) satisfies (EE1) – (EE5) and also condition (C–).

Using the relationships between contraction and revision, the Levi and Harper identities, a constructive definition of revision based on epistemic entrenchment can also be derived:

$$\text{(C*)} \qquad \beta \in K^*_\alpha \text{ iff either } \vdash \neg\alpha \text{ or } \neg\alpha <_E \alpha \rightarrow \beta$$

Similarly, if the relation $\leq_E$ satisfies (EE1) – (EE5), then the revision function which is uniquely determined by (C*) satisfies (K*1) – (K*8), and the contraction function generated by the Harper identity satisfies (C$\leq_E$).

With these equivalences in place, the problem of choosing a contraction or revision function has been replaced by the problem of choosing an epistemic entrenchment ordering. Epistemologically, it is possible to take either entrenchment or contraction or revision as the fundamental notion from which the other models are derived. But, from a computational point of view, it is more promising to use epistemic entrenchment as the primitive notion, so this is the approach taken within this thesis.

Before describing some of the alternatives to AGM belief revision, we will present three other constructive models of AGM belief revision.

2.2.2 Maximal Consistent Subsets

The early work on contraction and revision functions [Alchourrón & Makinson 1982; Alchourrón *et al.* 1985] characterised contraction functions in terms of maximal subsets of the belief set which do not imply the sentence being contracted from the theory. If K is a belief set, then let $K\perp\alpha$ be the set of all maximal subsets of K that do not imply α. That is, for all $M \in K\perp\alpha$, $M \nvdash \alpha$ and if $M \subset M' \subseteq K$ then $M' \vdash \alpha$.

A contraction function that chooses a member of $K\perp\alpha$ is called a *maxichoice contraction function*. It was shown in [Alchourrón & Makinson 1982] that such contractions are too large. The maxichoice revision defined from a maxichoice contraction via the Levi identity has the property that if the new sentence is not consistent with K, then the revised belief set is a complete theory, even if K was not complete initially. Also, if $\alpha \in K$, then for all propositions β, either $\alpha\vee\beta \in K_\alpha^-$ or $\alpha\vee\neg\beta \in K_\alpha^-$, for a maxichoice contraction operation.

Nevertheless, such contraction functions do satisfy the basic postulates (K-1) – (K-6). Also, if there is some partial ordering on the power set of K, and if the contraction function always chooses an element of $K\perp\alpha$ which is maximal with respect to this ordering, then it will satisfy (K-7) and (K-8).

The most simple alternative to maxichoice contraction is to define the contraction of K by α to be the intersection of all maximal consistent subsets which do not imply α. That is, $K_\alpha^- = \bigcap K\perp\alpha$, where $K\perp\alpha$ is nonempty, otherwise $K_\alpha^- = K$. Such a function is called a *full meet contraction function*, and yields a set which is too small, since [Alchourrón & Makinson 1982] shows that $K_\alpha^- = K \cap Cn(\neg\alpha)$. Also, the resulting revision function defined via the Levi identity has the property that if $\neg\alpha \in K$, then $K_\alpha^* = Cn(\alpha)$. That is, all the original contents of the belief set are lost.

Once again, this function satisfies the basic postulates (K-1) – (K-6), but not the supplementary postulates. It has been shown that the full meet contraction function is a lower bound for any contraction function satisfying the basic postulates; that is, $\bigcap K\perp\alpha$ must be contained in any contraction of K by α.

Finally, suppose there is a selection function S that chooses the "best" elements of $K\perp\alpha$. That is, $S(K\perp\alpha) \subseteq K\perp\alpha$, for nonempty $K\perp\alpha$, and $S(K\perp\alpha) = K$ otherwise. Then the contraction function defined by $K_\alpha^- = \bigcap S(K\perp\alpha)$ is called a *partial meet contraction function*.

Partial meet contraction functions are fully characterised by the basic postulates (K-1) – (K-6). That is, a contraction function is a partial meet contraction function if and only if it satisfies (K-1) – (K-6). Furthermore, if the selection function S chooses subsets of K which are maximal with respect to some relation on subsets in $K\perp\alpha$, then the contraction function is called a *relational partial meet contraction function*, and satisfies (K-7). Furthermore, if this relation is transitive, then the resulting function is called a *transitively relational partial meet contraction function*, and also satisfies (K-8). In fact, we have another representation theorem: a contraction function is a transitively relational partial meet contraction function if and only if it satisfies (K-1) – (K-8).

Although this gives us an alternative modelling for belief change functions, the ordering on sentences provided by epistemic entrenchment is a simpler construction than an ordering on maximal subsets of K which do not imply the sentence being removed. In the following chapter, we shall show that entrenchment also has an elegant computational model, and hence is preferable for the purposes of the current work.

2.2.3 Systems of Spheres

An alternative model of revision functions is found in [Grove 1988]. This work is based on the set M_L of all maximal consistent extensions of L, which may be seen as the set of possible worlds which can be described in L. If T is a set of sentences, then let $[T]$ denote the set of maximal consistent extensions of T, defined by $[T] = \{m \in M_L : T \subseteq m\}$. For a sentence α, we define $[\alpha]$ to be shorthand for $[\{\alpha\}]$. Also let $t(S)$ be the set of sentences true in all members of S, that is: $t(S) = \cap S$. Suppose the belief set K is represented by $X = [K]$. Then Grove defines a system of spheres S, centred on X, such that S is a collection of subsets of M_L with the following properties:

(S1) S is totally ordered by $\subseteq$
(S2) $X \in S$ and for all $U \in S$, $X \subseteq U$ (X is $\subseteq$-minimal)
(S3) $M_L \in S$ (M_L is $\subseteq$-maximal)
(S4) If α is a sentence and there is some sphere in S that intersects $[\alpha]$, then there is a smallest sphere in S that intersects $[\alpha]$ (denoted $c(\alpha)$).

Then we may define a function $f_S(\alpha) = [\alpha] \cap c(\alpha)$ which selects the "closest" worlds in M_L to X in which α holds. Then the revision function defined by $K^*_\alpha = t(f_S(\alpha))$ satisfies the AGM postulates (K*1) – (K*8) and, conversely, for any revision function satisfying (K*1) – (K*8), there exists a system of spheres S such that $t(f_S(\alpha)) = K^*_\alpha$. Hence we have another representation theorem for AGM belief change.

Grove provides a second modelling using an ordering on sentences, which is similar to entrenchment, but with the opposite sense. Define $\leq_S$ by $\alpha \leq_S \beta$ if and only if $c(\alpha) \subseteq c(\beta)$. Then $K^*_\alpha = \{\beta \in L : (\alpha \wedge \beta) <_S (\alpha \wedge \neg\beta)\}$ defines the same revision

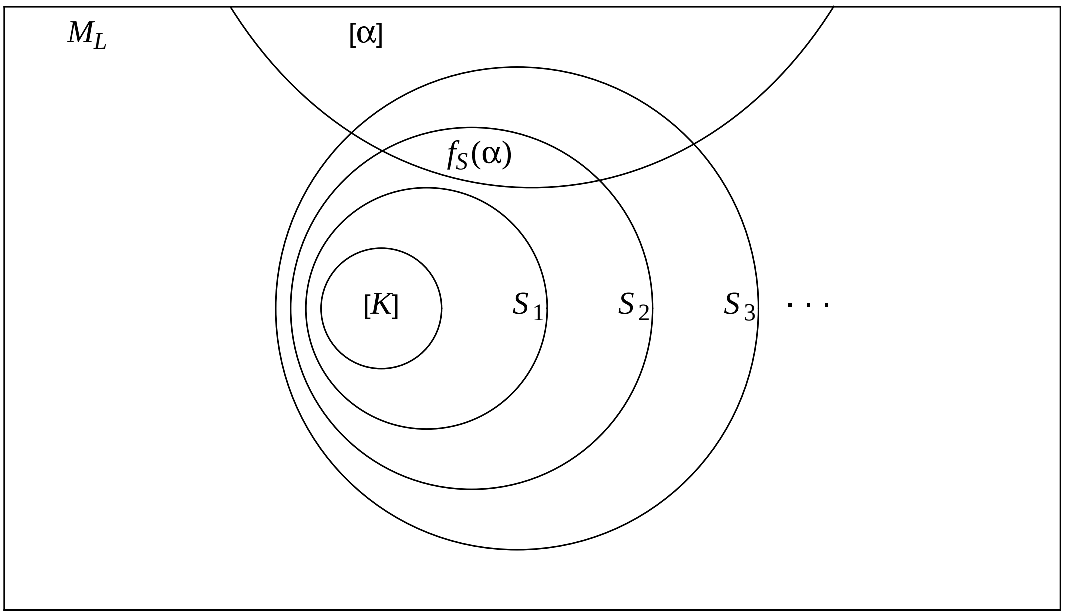


Figure 2.1: Belief revision using Grove's system of spheres

function as Grove's first model. Intuitively, we understand this as $\beta \in K^*_\alpha$ if and only if the closest worlds containing α also contain β, and none of them contain $\neg\beta$. This idea corresponds with the interpretation of the counterfactual conditional $\alpha \Rightarrow \beta$ in [Lewis 1973], and is also used for the ordinal conditional functions of [Spohn 1988].

2.2.4 Safe Contraction

An alternative approach to constructing a contraction operation is the *safe contraction* of [Alchourrón & Makinson 1985]. For a belief set K partially ordered by an irreflexive and transitive relation $<_S$, we say that an element $\beta \in K$ is *safe* with respect to α if and only if every minimal subset of K that implies α either does not contain β, or else contains at least one element $\gamma <_S \beta$. Then the safe contraction of α from K is the set of consequences of the safe (with respect to α) elements of K. Safe contraction satisfies the basic postulates for contraction, (K-1) – (K-6), and also satisfies the two supplementary postulates (K-7) and (K-8) if the safety relation satisfies the following two conditions for all α, β, $\gamma \in K$:

(1) If $\alpha <_S \beta$ and $\beta \vdash \gamma$ then $\alpha <_S \gamma$
(2) If $\alpha <_S \beta$ then either $\alpha <_S \gamma$ or $\gamma <_S \beta$

The notion of safety is somewhat different to epistemic entrenchment, yet [Rott 1992] showed that safe contraction defined as above corresponds exactly to the class of contraction functions generated from epistemic entrenchment (or equivalently to the class of partial meet contraction functions). Hence safe contraction is just another means of constructing AGM contraction operations.

2.3 Theory Base Approaches to Belief Revision

Interest in the area of belief revision amongst the computer science community has led to a number of different approaches using sets of sentences (*theory bases*) which are not closed under logical consequence. These sets are of particular interest when they are finite (*finite bases*), so that the belief set can be represented directly on a computer. Despite the intention of creating implementable models of belief revision, surprisingly few of these models have been implemented. In this section we summarise and compare the major contributions to this area.

Theory base approaches usually satisfy the "inclusion postulate", which states that when a base is contracted by any sentence, the resulting base is a subset of the original base. This condition is not compatible with the AGM recovery postulate, and it also makes the contraction operation $\ominus$ syntax-dependent, in the sense that we can have two bases Γ and Δ such that $Cn(\Gamma) = Cn(\Delta)$, but $Cn(\Gamma_\alpha^\ominus) \neq Cn(\Delta_\alpha^\ominus)$.

2.3.1 Minimal Contraction (Fuhrmann)

[Fuhrmann 1991] presents a theory base contraction operation which is minimal in the sense that formulae are not removed from the base unnecessarily. Suppose that the belief set *K* is represented by the base *B*, so that $K = Cn(B)$, and we wish to contract the sentence α from *K*. Firstly, the sentences in *B* which are essential to the derivation of α must be identified. A set *S* of these essential sentences forms an *entailment set* for α in *B*, satisfying:

(1) $S \subseteq B$

(2) $\alpha \in Cn(S)$

(3) If $S' \subset S$ then $\alpha \notin Cn(S')$

The set of all such entailment sets is denoted $E(\alpha, B)$. Clearly, for a contraction operation to succeed, at least one member of each entailment set for α must be removed from the base. The choice of which members to remove is determined by a preference relation <, called the ordering of comparative retractability. This ordering is only required to be acyclic, which ensures for finite bases that a minimal element always exists. Then the contraction operation is defined to remove all such minimal elements, and nothing else, from the base.

Minimal contraction corresponds to safe contraction, as defined in [Alchourrón & Makinson 1985] – although safe contraction is defined on closed theories and not theory bases – and thus it satisfies the basic rationality postulates (K-1) – (K-6) when B is closed. The recovery postulate (K-5) is not satisfied if *B* is not closed. If some further conditions on the preference relation are met, then the resulting contraction function also satisfies the supplementary postulates (K-7) and (K-8).

Fuhrmann argues that theory bases should satisfy the following *filtering condition*:

> If β has been retracted from a base *T* in order to bar derivations of α from *T*, then the contraction of *T* by α should not contain any sentences which were in *T* "just because" β was in *T*.

This condition captures a notion of foundationalism: when a belief is retracted from a theory, its consequences should also be removed from the theory. This idea will be explored further in chapter 6.

2.3.2 Minimal Contraction (Hansson)

[Hansson 1989] generalises the AGM partial meet contraction functions by considering contraction operations by sets of sentences rather than single sentences. To achieve this generalisation, he describes an extended set of basic postulates for contraction, with a constructive definition for *composite partial meet contraction functions*, which exactly characterises the functions which satisfy the extended postulates. Using these definitions, multiple successive belief changes can be avoided, by combining multiple contractions into a single operation.

Two theory base approaches to contraction are then presented, for which the inclusion postulate ($T^-_\alpha \subseteq T$ for any base *T* and any sentence α) is not required to hold. In these approaches, *full minimal contraction* and *partial minimal contraction*, disjunctions of explicit formulae from the base are added to the contracted base if they are contained in the full or partial meet contraction (respectively), and they are not already derivable from the base. Formally, if T is a theory base, then $\bigvee_n T$, with $n \geq 1$, is the set of expressions that are disjunctions of at most n distinct elements of T. Then the full (respectively partial) minimal contraction of T by α, denoted $T^{\ominus}_{\alpha}$, is defined as follows, where $-$ is a full (respectively partial) meet contraction operation:

$$D_n = (\bigvee_n T)^{-}_{\alpha}$$
$$E_1 = D_1$$
$$E_{n+1} = E_n \cup (D_{n+1} - Cn(E_n))$$
$$T^{\ominus}_{\alpha} = \bigcup_i E_i$$

For logically closed theories T, this definition is equivalent to full (respectively partial) meet contraction. For non-closed bases, the extended basic postulates for contraction hold, but recovery is vacuous, since it is conditional upon the base being logically closed. Thus the addition of the extra disjunctions does not give recovery, and hence we see it as an unnecessary inflation of the base, in the general case. In chapter 4, we describe an approach where no more than one sentence is added to the base for each sentence removed by the contraction operation, and this procedure ensures that the recovery postulate holds.

[Hansson 1991] claims that recovery is not a necessary property of a rational contraction operation, and replaces the recovery postulate with a weaker condition, *core-retainment*, which is equivalent to recovery for closed theories, but not for theory bases. An example is presented which shows that there are cases where a partial meet contraction on a theory base is not a suitable contraction operation, but a partial meet contraction operation on the closure of the base, such as an epistemic entrenchment contraction, can be constructed to give the intuitively desired results.

2.3.3 E-bases, Ensconcements, and Most Conservative Entrenchments

A more general approach to constructing theory base contraction and revision functions which satisfy the AGM postulates must specify an efficient means of representing the epistemic entrenchment relation or some equivalent preference relation. Using this representation, belief change operations can then be defined which achieve the same behaviour as the AGM operations. Several very similar methods have been developed independently, all of which provide the first step towards a computational approach to AGM belief revision.

[Rott 1991] provides an efficient representation of entrenchment relations for the purpose of discussing particular examples of entrenchment relations. Hence the representation is natural and easy to understand. Rott defines an E-base to be a pair $\langle B, \leq_R \rangle$ consisting of a set of sentences B from the language L, and an ordering $\leq_R$ on B which is reflexive, transitive and connected. Then a $B-cut$ is any subset S of B such that if $\alpha \in S$ and $\alpha \leq_R \beta$ then $\beta \in S$. The entrenchment relation $\leq_E$ generated from the E-base $\langle B, \leq_R \rangle$ is given by:

$$\alpha \leq_E \beta \text{ if and only if for all } B\text{-cuts } S, \text{ if } \alpha \in Cn(S) \text{ then } \beta \in Cn(S),$$

for all sentences α and β in L. The definition requires that the E-base satisfies the following condition, known as the *entailment condition*:

$$\text{If } \emptyset \neq \Gamma \vdash \alpha \text{ then } \beta \leq_R \alpha \text{ for some } \beta \in \Gamma$$

This condition, when applied to an entrenchment relation on a belief set satisfying entrenchment postulate (EE1), is equivalent to the conjunction of postulates (EE2) and (EE3). The entrenchment relation generated from an E-base satisfying the entailment condition preserves the ordering of all sentences in B; that is, for all $\alpha, \beta \in B$, $\alpha \leq_E \beta$ if and only if $\alpha \leq_R \beta$.

When B is finite, the relation $\cong_R = \leq_R \cap \leq_R^{-1}$ partitions B into finitely many equivalence classes $B_0, B_1, \cdots, B_n$, where the indices are chosen such that if $\alpha \in B_i$ and $\beta \in B_j$ then $\alpha \leq_R \beta$ if and only if $i \leq j$. The equivalence classes $K_0, K_1, \cdots, K_n$ of the entrenchment relation $\leq_E$ generated from the E-base $\langle B, \leq_R \rangle$ are given by $K_i = Cn(\bigcup_{j \geq i} B_j) - Cn(\bigcup_{j \geq i+1} B_j)$, for $i = 0, 1, \cdots, n-1$, and $K_n = Cn(\emptyset)$. Then the entrenchment of any belief may be represented by the integer i corresponding to the partition of the belief set to which it belongs. This integer will be called the *rank* of the belief in chapter 3.

Rott does not go on to define belief change algorithms based on this construction; instead he addresses some of the issues involved in choosing the most suitable entrenchment relation for a given problem. We shall discuss this further in chapter 3.

An equivalent representation of the entrenchment relation is found in [Williams 1992, 1993], under the name of the *ensconcement* relation. This relation is developed with a very different purpose: to support a computer-based implementation of belief revision. The epistemic state is represented by an ordered pair consisting of a set of sentences Γ and a total pre-order $\leq_W$ on Γ satisfying:

$$\text{For all } \beta \in \Gamma,\ \{\alpha \in \Gamma : \beta <_W \alpha\} \nvdash \beta$$

Clearly this condition is just the contraposition of Rott's entailment condition described above, and hence the two representations are equivalent. The representation is used two define two computationally efficient contraction operations in [Williams 1992], both of which satisfy the inclusion postulate and hence do not satisfy the AGM recovery postulate. The same applies to the contraction function $\ominus$ presented in [Williams 1993] (using the same notation as for E-bases):

$$\beta \in B^{\ominus}_{\alpha} \text{ if and only if } \beta \in B \text{ and either } \bigcup_{i=rank(\alpha)+1}^{n} B_i \cup \{\neg\alpha\} \vdash \beta \text{ or } \vdash \alpha,$$

where $rank(\alpha)$ is the index of the partition of the belief set containing α. It is shown that this contraction operation is as close as possible to the AGM contraction operation on belief sets, without sacrificing inclusion. That is:

$$B^{\ominus}_{\alpha} = B \cap (Cn(B))^{-}_{\alpha}$$

where – is the contraction function determined uniquely by the entrenchment relation generated from the ensconcement $\leq_W$. The corresponding revision function, defined by the Levi identity, satisfies all of the AGM postulates:

$$(Cn(B))^{*}_{\alpha} = Cn(B^{\ominus}_{\neg\alpha} \cup \{\alpha\})$$

An implementation of the theory change operations in [Williams 1993] is described in [Dixon 1993].

The third construction of an entrenchment relation from a partially specified relation is the *most conservative entrenchment* developed in [Wobcke 1992a]. Given a theory base Γ and an ordering $\leq_\Gamma$ on Γ satisfying (EE1) to (EE3), an entrenchment $\leq_e$ is defined to be *compatible* with $\leq_\Gamma$ if for all α, $\beta \in \Gamma$, $\alpha \leq_\Gamma \beta$ implies $\alpha \leq_e \beta$. Then if $\leq_e$ and $\leq'_e$ are two entrenchments which are compatible with $\leq_\Gamma$, the entrenchment $\leq_e$ is defined to be *more conservative* than $\leq'_e$ if there is a formula α such that for every

β, where $\beta \leq_e \alpha$ and $\beta \leq'_e \alpha$, we have $\{\gamma : \gamma \leq'_e \beta\} \subseteq \{\gamma : \gamma \leq_e \beta\}$ and $\{\gamma : \gamma \leq'_e \alpha\} \neq \{\gamma : \gamma \leq_e \alpha\}$.

Then the *most conservative entrenchment* compatible with $\leq_\Gamma$ is the unique entrenchment $\leq_e$ compatible with $\leq_\Gamma$ such that for all entrenchments $\leq'_e$ compatible with $\leq_\Gamma$, $\leq_e$ is more conservative than $\leq'_e$. The most conservative entrenchment is conservative in the sense that it places each formula as low as possible in the entrenchment ordering.

The three constructions presented in this section unambiguously extend a partial specification of an entrenchment relation to a complete epistemic entrenchment relation agreeing with the partial specification. In the case where the partial specification of the entrenchment is finite, we have an efficient representation of epistemic entrenchment, useful for a computational construction of AGM belief revision.

2.3.4 Database Update

In [Fagin *et al.* 1983, 1986], a theory of database update is developed which closely resembles belief revision using theory bases. Two operations are defined: insertion (expansion and revision) and deletion (contraction); a third operation, replacement, is also mentioned as a topic of further research. The aim of these operations is to accomplish the update successfully, whilst preserving as much as possible of the original database.

The minimal change for an update is defined as follows. Suppose our database is represented by the set T of logical formulae, and T_1 and T_2 both accomplish an update successfully. (T' accomplishes the insertion of α into T if $\alpha \in T'$, and T' accomplishes the deletion of α from T if $\alpha \notin Cn(T')$.) Then T_1 has fewer insertions than T_2 if $T_1 - T \subset T_2 - T$; also T_1 has no more insertions than T_2 if $T_1 - T \subseteq T_2 - T$; and T_1 has the same insertions as T_2 if $T_1 - T = T_2 - T$. Similarly, the deletions can be defined by comparing the sets $T - T_1$ and $T - T_2$. Then T_1 achieves an update u of T with *smaller change* than T_2 if both T_1 and T_2 achieve u, and either T_1 has fewer deletions than T_2 or T_1 has the same deletions as T_2 but T_1 has fewer insertions than T_2. Finally, the notion of *minimal change* is defined as expected: a set of formulae S accomplishes an update u of T *minimally* if there is no set S' that accomplishes u with smaller change than S.

Two representation theorems are then proved for minimal updates. If S and T are sets of sentences, and α is a sentence, then:

(1) S accomplishes the deletion of α from T minimally if and only if S is a maximal subset of T that is consistent with $\neg\alpha$

(2) $S \cup \{\alpha\}$ accomplishes the insertion of α into T minimally if and only if S is a maximal subset of T that is consistent with α

In general, there will be more than one such minimal update. In fact, for a closed theory T, the deletion of α from T is accomplished minimally by every member of $T\perp\alpha$, which is equivalent to maxichoice contraction. The insertion of T is accomplished minimally by $S\cup\{\alpha\}$, for each $S \in T\perp\neg\alpha$, which yields theory bases for maxichoice revision. Once more, some way of combining these possible solutions, or selecting a preferable solution, is needed. [Fagin *et al.* 1983] suggests two ways of approaching this problem. The first is to take the intersection of the theories, which is equivalent to full meet contraction and revision. This is too strong, as the insertion of a sentence that is inconsistent with the current theory causes the whole theory to be abandoned. They interpret this problem as being caused by the use of closed theories, and so the rest of the work considers only non-closed sets of sentences.

The second approach provides a preference relation of *database priorities* on the sentences in the database, by tagging each sentence with a natural number. The sentences with the lowest tags have the highest priority; for example, integrity constraints would normally be tagged by 0. Then a logical database D is a set of pairs $<i, \alpha>$ where α is a sentence, and i is its tag. Define $D^i = \{\langle j, \alpha\rangle \in D : j \leq i\}$ to be the set of sentences in D with tags less than or equal to i. If D is a database with greatest tag n, and E and F accomplish some update u, then E accomplishes u with smaller change than F if either:

(1) $\exists i, 0 \leq i \leq n$, such that $D^{i-1}-E^{i-1} = D^{i-1}-F^{i-1}$ and $D^i-E^i \subset D^i-F^i$, or

(2) $D^n-E^n = D^n-F^n$ and $E-D \subset F-D$.

Condition (1) states that E has fewer deletions at the highest priority at which E and F differ, and condition (2) states that E and F have the same deletions, but E has fewer insertions than F. Then a minimal update u of D is achieved by E if E accomplishes u and there is no logical database F which accomplishes u with smaller change than E. Finally, the following representation theorem is presented, for logical databases D and E, with highest tag n, and for any consistent sentence α:

(1) E accomplishes the deletion of α from D minimally if and only if E^i is a maximal subset of D^i that is consistent with $\neg\alpha$, for $i = 1, \cdots, n$.

(2) $E \cup \{\langle j, \alpha\rangle\}$ accomplishes the insertion of α into D minimally if and only if E^i is a maximal subset of D^i that is consistent with α, for $i = 1, \cdots, n$.

A third approach, presented in [Fagin *et al.* 1986], is to keep track of sets of possible databases generated by updates. These sets of databases are called *flocks*, and the update of a flock is the union of the updates of each member of the flock. From a computational perspective, this approach is much less appealing than alternatives which only keep track of a single state of belief.

Comparing this work with AGM belief revision, the main differences are that the results of updates are syntax-dependent and do not satisfy the AGM recovery postulate (K-5). Although this is common amongst methods using non-closed belief sets, we show in chapters 3 and 4 that these problems are easily overcome.

2.3.5 Epistemic Relevance and Prioritized Bases

The first theory base contraction operation to satisfy the AGM recovery postulate is found in [Nebel 1989], where symbol level belief revision operations, such as those used in reason maintenance and other artificial intelligence applications, are analysed at the knowledge (i.e. logically closed theory) level. Nebel demonstrates a relationship between the idealised belief revision processes of the AGM theory and the computational, "real world" operations, by showing that finite base operations such as reason maintenance are a special case of the more general AGM operations.

Suppose the beliefs are represented by a finite set of sentences B, where B is also used to represent the conjunction of these sentences. Then for any sentence α such that $\not\models \alpha$, the contraction of B by α is initially defined to be the disjunction of all maximal subsets of B not implying α, that is, $\bigvee(B \perp \alpha)$, which does not satisfy the recovery postulate. Nebel notes that although full meet contraction is very weak, as it is logically equivalent to $Cn(B) \cap Cn(\neg\alpha)$, it does satisfy recovery, and so recovery can be obtained by adding a conjunct to the contracted base, giving the contraction operation $\ominus$, defined by:

$$B^{\ominus}_{\alpha} = \bigvee(B \perp \alpha) \wedge (B \vee \neg\alpha)$$

Nebel shows that if we take the logical closure of the contracted base, $\ominus$ satisfies all of the basic postulates for contraction (K-1) – (K-6), as well as supplementary postulate (K-7), but (K-8) is not always satisfied. In fact, the operation is equivalent to the partial meet contraction using the following selection function S_B (where $K = Cn(B)$):

$$S_B(K\perp\alpha) = \{C \in (K\perp\alpha) : \forall C' \in (K\perp\alpha),\, C' \cap B \not\supset C \cap B\}$$

In [Nebel 1990], it is argued that the beliefs in the base usually represent observations, facts, laws or rules, and thus are more relevant than the derived beliefs. Thus the selection function S_B chooses from $K\perp\alpha$ the elements which contain maximal subsets of these relevant propositions; that is, it minimises the loss of epistemically relevant information.

Epistemic relevance is a very different notion from epistemic entrenchment. Firstly, only two degrees of relevance exist (relevant and irrelevant); secondly, entrenchment is strictly connected to the contraction operation via the (C-) condition, whereas epistemic relevance only constrains the syntactic form of the resulting belief base; and, most importantly, contraction and revision based on epistemic relevance do not satisfy the eighth rationality postulate for each operation.

The notion of epistemic relevance is then extended to a more fine-grained measure, by assigning natural numbers to all sentences in the logical language. This allows further distinctions such as assigning a higher degree of relevance to integrity rules than simple facts in a database, as suggested by [Fagin *et al.* 1983]. A *prioritized set inclusion* is then defined which tests inclusion for two sets of sentences at the highest level of relevance for which the sets differ. Like the original notion of relevance, this could also be used to define a selection function for a partial meet contraction function satisfying (K-1) – (K-7), but not (K-8), as the inclusion relation is not transitive.

[Nebel 1991] presents a further approach based on epistemic relevance, where a contraction is constructed by aggregating maximal subsets of the beliefs in the base at each level of relevance, working down from the sentences with the highest degree of relevance. The *prioritized base* contraction and revision functions generated in this way correspond exactly to the functions generated from epistemic relevance as described above.

Although similar to the AGM operations, Nebel's belief change operations are not fully rational, in that they are syntax-dependent and thus do not satisfy all of the AGM postulates.

2.4 Model-Theoretic Approaches to Belief Revision

In order to avoid the problem of syntax-dependence which occurs when computing belief changes from theory bases, several authors argue that revision operations should be characterised at the model-theoretic level [Dalal 1988, Katsuno & Mendelzon 1989]. In this framework, belief sets are viewed as the set of models that satisfy the given belief base, and belief change operations select sets of models that satisfy the new information and differ minimally from the models of the original belief base. [Dalal 1988] measures the difference between two models as the number of propositional variables that have different values in the two models. [Katsuno & Mendelzon 1989] presents a more general approach using a preorder over the models. [Peppas & Williams 1992] describes a constructive modelling of belief revision functions via a *nice preorder* on models, and gives explicit translations between the modelling and two other constructive modellings: epistemic entrenchment and systems of spheres.

From a computational perspective, model-theoretic belief change functions are not particularly useful, unless there is an efficient means of manipulating these representations. [Gärdenfors 1991] describes these approaches as "like putting the cart in front of the horse, since orderings of models seems epistemologically more advanced than orderings of sentences".

2.5 Conditional Functions and Iterated Belief Change

A very different approach to belief revision is found in [Spohn 1988]. In this work, the epistemic state is not represented directly by a set of formulae with an associated preference relation, but indirectly, by functions from sets of worlds to sets of worlds. Firstly let W be a set of possible worlds. Then a proposition is defined to be any subset of W. The net content of an epistemic state is defined to be the set of worlds satisfying all current beliefs, which is itself a proposition. (This concept is an analogue of representing a set of logical sentences by the conjunction of all of the sentences in the set.) Spohn defines two ways of representing an epistemic state: by a simple conditional function (SCF) and by an ordinal conditional function (OCF).

An SCF is a function g from non-empty subsets of W to subsets of W such that for all non-empty $A,\ B \subseteq W$:

(1) $\varnothing \neq g(A) \subseteq A$
(2) If $g(A) \cap B \neq \varnothing$, then $g(A \cap B) = g(A) \cap B$

Then g provides a response scheme to new information – if we receive new information A then the net content of our new epistemic state will be $g(A)$. The net content of the current epistemic state is given by $g(W)$, since the tautology W leaves the epistemic state unchanged. Hence g is a revision function, and it is straightforward to show that g satisfies the AGM postulates for revision (K*1) – (K*8). (Firstly, postulates 1 and 6 are trivially satisfied by the choice of representation. (K*2) follows from the second part of condition (1), $g(A) \subseteq A$. (K*3) and (K*4) are special cases of (K*7) and (K*8) respectively, which are combined in condition (2). This can be seen by translating the condition into AGM notation: if $B \in K^*_A$, then $K^*_{A \wedge B} = (K^*_A)^+_B$. Finally (K*5) comes from the first part of condition (1), $g(A) \neq \varnothing$.)

Spohn then proves a representation theorem not unlike [Grove 1988]'s systems of spheres. In this section, let lower case Greek letters denote ordinal numbers. Then the sequence $(E_\alpha)_{\alpha<\zeta}$ is a *well-ordered partition* (WOP) if and only if for all $\alpha,\ \beta < \zeta$:

$E_\alpha \neq \varnothing$,
$E_\alpha \cap E_\beta = \varnothing$, for $\alpha \neq \beta$, and
$\cup_{\alpha<\zeta} E_\alpha = W$.

The connection to Grove's spheres can be seen by setting $S_\alpha = \cup_{\beta<\alpha} E_\beta$. Then if $(E_\alpha)_{\alpha<\zeta}$ is a WOP and g is an SCF, we say that $(E_\alpha)_{\alpha<\zeta}$ represents g if and only if for each non-empty $A \subseteq W$:

$$g(A) = E_\beta \cap A, \text{ where } \beta = \min\{\alpha : E_\alpha \cap A \neq \emptyset\}.$$

The representation theorem proved by Spohn states that there is a one to one correspondence between SCFs and WOPs. The advantage with WOPs is that they have an intuitive interpretation as an *ordering of disbelief*; E_0 contains the worlds which are not disbelieved at all, E_1 contains the worlds with the lowest level of disbelief, E_2 the next least disbelieved, and so on. Then the revision by A consists of the least disbelieved worlds in which A is true, agreeing with Grove's construction using systems of spheres.

Like the AGM approach, SCFs and WOPs do not solve the problem of iterated belief change. That is, the original epistemic state is represented by an SCF or WOP, but the revised state is only represented by its net content, so there is no way of performing further belief changes, without constructing a new SCF or WOP.

Suppose the initial epistemic state is given by the WOP:

$$E_0, E_1, E_2, \cdots, E_\zeta.$$

The last term is placed in the sequence for illustrative reasons only. Suppose also that we want to accept the new belief A which is currently disbelieved at a level β. That is, the first A world appears in the partition E_β.

Spohn's first proposal for reconstructing the WOP after a belief change to accept A is to make all possible worlds in A less disbelieved than the worlds in $\bar{A}$ (where $\bar{A} = W - A$, the complement of A in W). The ordering of worlds within A and $\bar{A}$ are preserved, giving the new WOP:

$$E_\beta \cap A, \ \cdots, E_\zeta \cap A, \ E_0, \ \cdots, E_{\beta-1}, \ E_\beta \cap \bar{A}, \ \cdots, E_\zeta \cap \bar{A}.$$

To ensure that the above sequence is a WOP, all empty terms must be deleted. Unfortunately, this proposal is too simplistic. Spohn rejects it on three counts: epistemic changes are not reversible, nor are they commutative, and the proposal is too strong, as it assigns the highest degree of confidence to the new belief. Therefore, a subsequent change of belief will not dislodge our confidence in A, unless we accept a proposition P which entails the negation of A, that is, $P \subseteq \overline{A}$. In terms of epistemic entrenchment, this is equivalent to placing A at a level of entrenchment at least as high as all non-tautological beliefs.

The second proposal is the converse of the first: only the least disbelieved A worlds are moved to the top of the WOP, giving the sequence:

$$E_\beta \cap A,\ E_0,\ \cdots,\ E_{\beta-1},\ E_\beta \cap \overline{A},\ E_{\beta+1},\ \cdots,\ E_\zeta.$$

Again, the change is not reversible or commutative, and it is also too weak, as the new belief is given the least degree of confidence. That is, any further belief change to accept a proposition which is not consistent with the new state of belief will result in the belief in A being rejected. This is equivalent to assigning to A the least entrenchment of any sentence in the belief set.

The failure of these two proposals leads to the obvious conclusion that, for a general belief change operation, the degree of confidence in the new information which we are accepting must be specified. To do this, SCFs are generalised to OCFs (ordinal conditional functions), based on the following definitions. Let $\mathcal{A}$ be a complete field of propositions over W (i.e. closed under complementation, intersection and union), and let an *atom* of $\mathcal{A}$ be a non-empty set which has no non-empty proper subset in $\mathcal{A}$. Then κ is an *$\mathcal{A}$-OCF* if and only if κ is a function from W into the class of ordinals such that:

(1) $\kappa^{-1}(0) \neq \emptyset$, and
(2) for all atoms A of $\mathcal{A}$, and all $w, w' \in A$, $\kappa(w) = \kappa(w')$.

Also, for any non-empty proposition $A \in \mathcal{A}$, define:

$$\kappa(A) = \min\{\kappa(w) : w \in A\}.$$

Then κ is a measure of disbelief, and we say that A is believed with firmness α if $\kappa(A) = 0$ and $\alpha = \kappa(\overline{A})$, or else $\kappa(A) > 0$ and $\alpha = -\kappa(A)$. A is believed (respectively disbelieved) if its degree of firmness is positive (respectively negative). Also, $\kappa^{-1}(0)$ represents the net content of the epistemic state, which is why it must be non-empty, for a consistent state of belief. Now we may finally define Spohn's belief change operation, called A, α*–conditionalisation*:

$$\kappa_{A,\alpha}(w) \;=\; \begin{cases} -\kappa(A) + \kappa(w)\,, & \text{if } w \in A \\ \alpha - \kappa(\overline{A}) + \kappa(w)\,, & \text{if } w \in \overline{A} \end{cases}$$

If κ is the current epistemic state, then $\kappa_{A,\alpha}$ is the resulting state after revising κ by A so that A is believed in $\kappa_{A,\alpha}$ with firmness α. Note also that $\kappa_{\overline{A},0}$ implements a contraction of the epistemic state by *A*, and that expansion is the special case of revision where the new proposition is consistent with the current beliefs. Hence, all of the AGM belief change operations are implemented by A, α-conditionalisation, and the resulting state is a new OCF, so that iterated changes of belief may be performed. Spohn also shows that the belief changes are commutative and reversible, conditions which are not required by the AGM postulates.

2.6 Summary

The AGM approach to belief revision provides an idealised mathematical formalism for modelling rational change of belief. A non-constructive formulation of belief change operations is given by sets of rationality postulates, which define the classes of rational contraction and revision operations. Several constructive modellings of AGM belief change operations exist, based on selection functions or preference relations such as epistemic entrenchment, but none of these methods provide a computational approach to belief change. The model-based approaches, and the ordinal conditional functions of [Spohn 1988], also provide non-computational frameworks for belief revision.

The alternative constructions of belief change operations using theory bases rather than logically closed belief sets to represent the beliefs of an agent do not satisfy all of the AGM postulates, and thus do not represent ideally rational changes of belief. An efficient representation of an entrenchment relation has been developed independently in [Rott 1991], [Williams 1992] and [Wobcke 1992a], but this representation has not been used to provide a direct construction of AGM belief change operations. In the following chapters, we shall use this representation to develop a computational approach to belief change which satisfies all of the AGM rationality postulates.

3 A Computational Approach to AGM Belief Revision

As discussed in chapter 2, current approaches to formalizing belief revision are based on mathematical methods, using non-computational set-theoretic definitions for belief sets and belief change operations. The formal properties of the AGM belief revision functions have been examined in depth, but there have been few attempts to specify computational versions of belief revision systems, and none which satisfy all of the AGM postulates for fully rational belief revision.

In this chapter we present a computational model of the AGM approach to belief revision, suitable for implementation over any logical language with an implementable decision procedure for derivability. The belief set is represented by a finite set of formulae, from which the remaining beliefs are derived using a sound and complete inference procedure. If this procedure is tractable, then the computational approach using the algorithms described in chapter 4 will also be tractable.

The main theoretical issues addressed in this chapter are: the representation of an epistemic state, the extension of this representation to a most conservative entrenchment, the relationship between the principles of conservatism and independence of beliefs, and the specification of an entrenchment revision policy.

In order to specify belief change functions which satisfy the AGM postulates, we represent the epistemic state by an indexed set of formulae, called the *entrenchment base*, which is a finite representation of an entrenchment relation. The index to each formula in the entrenchment base is a natural number, which gives rise to a pre-order on the formulae in the base, creating a structure equivalent to a finite E-base [Rott 1991] or ensconcement [Williams 1992].

The full entrenchment relation is generated from the partially specified relation on the entrenchment base via a construction which assigns to each formula the lowest entrenchment consistent with the partial relation and the entrenchment postulates. Following [Wobcke 1992a], we call the relation generated by this construction the *most conservative* entrenchment. The most conservative entrenchment is equivalent to the entrenchment generated from an E-base in [Rott 1991] and the entrenchment obtained as the extension of the ensconcement ordering in [Williams 1992]. Conservative entrenchments are motivated firstly by considerations of entrenchments as representing evidence for beliefs, so that no sentence is assigned a higher ranking than that warranted by the explicit evidence for the belief, and secondly by connections with truth maintenance and nonmonotonic reasoning, which shall be discussed in chapters 6 and 7.

The next issue discussed is how the most conservative entrenchment induces an implicit assumption of dependence between the explicit beliefs when performing AGM belief change operations. This relates to the coherentist style of reasoning on which the AGM approach is based, but as we show in chapter 6, independence can easily be encoded into the entrenchment relation to provide foundational reasoning.

The final section describes the specification of an entrenchment revision policy. The AGM formalism offers no guidelines in this area: AGM belief change operations map a belief set with an entrenchment to a belief set without an entrenchment. Therefore, the AGM approach does not specify the result of two successive changes to a belief state. We propose an entrenchment revision policy based on applying the minimal change principle to the entrenchment relation. Intuitively, it is not just the beliefs themselves, but also the degree of evidence for the beliefs, which should be protected against unnecessary changes. That is, those beliefs which remain in the belief set after a belief change should retain their entrenchment ranking, unless the belief change provides new evidence which increases the rank of the belief.

In the following chapter, we specify efficient algorithms for the implementation of belief revision operations over any logic, and then in chapter 5, we describe a particular implementation of a first-order logic AGM belief revision system.

3.1 The Representation of Belief States

In the AGM approach, a belief state consists of a logically closed set of beliefs together with a set of operations for revising this belief set, which may be characterised by an epistemic entrenchment relation, an ordering on complete theories such as a system of spheres, a selection function on maximal consistent subsets, or by the choice of a particular contraction or revision function. In order to implement an AGM belief revision system, a representation must be found for both belief sets and belief change operations. The obvious choice for representing the belief set is that of a finite set of formulae, a *finite base,* from which all other beliefs can be derived using a sound and complete decision procedure for logical consequence.

The AGM model of belief revision has been criticised for its reliance on logical closure, which may not be suitable for modelling agents with limited resources, but we wish to model ideally rational agents which are "logically omniscient" [Williams 1993]. We argue that, for consistent reasoning, it is necessary to be able to query the beliefs that follow from the representation of the belief set, and therefore rational agents are always limited by the logic within which they can reason. Various alternative reasoning methods have been proposed, such as the theory base operations of [Williams 1993], but these still rely on the consequence operator to some extent.

On a more practical note, if a system is based on first-order logic, for example, there is no sound and complete decision procedure for logical consequence which is guaranteed to terminate. There are several possible solutions to this problem: we may restrict the logical language to a subset with known computational advantages, such as Datalog [Ullman 1988], propositional logic, or a tractable subset of first-order logic [Levesque & Brachman 1985]; we may give up completeness of the decision procedure for derivability; or we may allow the derivations to proceed indefinitely, relying on the user of the system to ensure that all necessary proofs will terminate. The implementation supports all three of these approaches, leaving the choice up to the user of the system. This is not an immediate concern, as the representations and algorithms developed in chapters 3 and 4 are independent of the base logic.

Particular contraction and revision functions can be specified using an epistemic entrenchment, and the (C–) and (C*) conditions, but in general it is not possible to represent the entrenchment directly, as it is a pre-order on all sentences in the language. In fact, a particular epistemic entrenchment may have no finite representation, as some entrenchment orderings allow infinitely many different levels of entrenchment of the beliefs. Moreover, even with a finite number of degrees of entrenchment, it is still not possible for the user of a system to enter a degree of entrenchment for each formula in the language: a more compact representation must be found.

Clearly what is necessary is a method of specifying an entrenchment partially, by a finite and intuitively appealing representation, from which any further entrenchment information can be generated automatically, without consulting the user. A preorder on the explicit beliefs (the formulae in the finite base) is the obvious candidate for such a representation, and it satisfies both the size and clarity requirements. To specify belief revision functions using this finite partially specified entrenchment, we make use of the most conservative entrenchment construction to extend the given relation to a total preorder on *L*. Note that we allow redundancy in the finite base, so that we can distinguish between differing degrees of evidence for sets of formulae which are logically dependent.

3.2 Most Conservative Entrenchments

The idea behind most conservative entrenchments is the following: the relative entrenchment of a formula with respect to other formulae is intended to represent the degree of evidence possessed by the system for that particular belief in relation to the other beliefs. A representation of a belief state is therefore a collection of formulae (those of the base) together with the degrees of evidence for each formula in the base. The belief set represented is the logical closure of the belief set base. Similarly, the most conservative entrenchment is the "closure", in some sense, of the "entrenchment base". To define this sense of closure, we take it that the only evidence possessed by the system for a belief is that derived from the evidence for formulae in the base: the evidence for a formula derived from a non-redundant collection of formulae is the evidence for the conjunction of these formulae, which is just the evidence for the weakest formulae in the conjunction. In this sense, the generated entrenchment is conservative in not attributing evidence for a belief other than that warranted by the evidence for the base beliefs from which that belief is derived.

There is a strong connection between most conservative entrenchments and foundational reasoning. In a static epistemic state, the only beliefs accepted are those which have evidence (a logical proof) which is derived from the explicit belief base (the foundational beliefs), which corresponds exactly to the foundational notion of justification. This correspondence also applies to the dynamics of belief. Since the derived beliefs are given an entrenchment no higher than the explicit beliefs from which they are derived, when one of these explicit justifying beliefs is removed, the derived beliefs do not remain in the belief set either. We will discuss this topic further in chapter 6, where we demonstrate a direct relationship between belief revision based on the most conservative entrenchment, and truth maintenance systems.

The most conservative entrenchment generated from an entrenchment base can be formalized as follows. We represent the degree of evidence for a belief as a natural number known as the *rank* of the formula, so a belief set is a set of formulae each with an associated rank. The higher the rank, the more evidence there is for the belief; and logical theorems have the highest rank of all. The entrenchment base Γ can then be represented by a partition of the formulae in the base determined by the subsets of formulae of the base all of which have the same rank. Suppose the logical theorems have rank n, and let the partitions be $\Gamma_1, \Gamma_2, \cdots, \Gamma_n$, some of which may be empty. The rank of a formula in the base is just the index of the partition to which it belongs. To define the most conservative entrenchment generated from this partition, we extend the definition of rank to all formulae. First we define:

$$\overline{\Gamma_i} = \bigcup_{j=i}^{n} \Gamma_j$$

Intuitively, $\overline{\Gamma_i}$ is the set of formulae in the base for which there is degree of evidence at least i. Then $Cn(\overline{\Gamma_i})$ is the set of all formulae for which there is degree of evidence at least i. Note that for $i \le j$, $\overline{\Gamma_j} \subseteq \overline{\Gamma_i}$, and also $Cn(\overline{\Gamma_j}) \subseteq Cn(\overline{\Gamma_i})$. Thus the rank of a formula α, representing the degree of evidence for α, is defined to be the largest i such that α is a consequence of $\overline{\Gamma_i}$. If α is not derivable from the belief base (α is not believed), then its rank is defined to be 0. Thus the set $\overline{\Gamma_1}$ is a base for the belief set K; that is, $K = Cn(\overline{\Gamma_1})$. Then the rank of a formula α is defined by (Def_Rank):

$$\mathbf{rank}(\Gamma, \alpha) = \begin{cases} \max(\{i : \overline{\Gamma_i} \vdash \alpha\}) & \text{if } \overline{\Gamma_1} \vdash \alpha \\ 0 & \text{if } \overline{\Gamma_1} \nvdash \alpha \end{cases}$$

3.2

The most conservative entrenchment determined by the partition of the base is then given by (Def_MCE):

$$\alpha \leq_E \beta \quad \text{iff} \quad \mathbf{rank}(\Gamma, \alpha) \leq \mathbf{rank}(\Gamma, \beta)$$

To ensure consistency with the AGM entrenchment postulates (EE1) – (EE5), the following two conditions on the entrenchment base must hold:

$$\text{(R1)} \quad \forall\, i, \ \ \forall\, \beta \in \Gamma_i, \ \ \overline{\Gamma_{i+1}} \nvdash \beta$$
$$\text{(R2)} \quad \alpha \in \Gamma_n \quad \text{if and only if} \quad \vdash \alpha$$

The first condition captures the requirement that an explicit belief must not have a derived rank which is greater than its explicit rank. Otherwise, the explicit evidence for the belief would be redundant, since higher ranked evidence could be found elsewhere, and thus the explicit rank would be incorrect. The algorithms in chapter 4 require a stronger notion of nonredundancy, defined by:

> An entrenchment base Γ is *nonredundant* iff for all *i*, and for all $\alpha \in \Gamma_i$, $\overline{\Gamma_i} - \{\alpha\} \nvdash \alpha$.

Condition (R2) states that only logical theorems are given the maximum rank value, to satisfy postulate (EE5). Then, for a nonredundant entrenchment base, Γ_n is always empty, since all logical theorems can be derived from the empty set.

We can now prove that the ordering generated from a ranked finite base satisfying conditions (R1) and (R2) is an epistemic entrenchment ordering:

> **Theorem 3.1:** Let Γ be an entrenchment base satisfying (R1) and (R2). Then the relation $\leq_E$ generated from Γ, defined by $\alpha \leq_E \beta$ iff $\mathbf{rank}(\Gamma, \alpha) \leq \mathbf{rank}(\Gamma, \beta)$ for all $\alpha, \beta \in L$, is the most conservative entrenchment, and satisfies postulates (EE1) - (EE5) for an epistemic entrenchment relation.

The proof of theorem 3.1 is found in appendix A.

In the rest of this section we describe the conditions under which a given epistemic entrenchment relation can be represented by a finite entrenchment base. Firstly, we note two representability conditions from [Williams 1992]:

(1) A belief set can be represented by a finite base if and only if it is finitely axiomatizable.

(2) An epistemic entrenchment relation can be represented by a finite entrenchment base if and only if it has a finite number of natural partitions each of which is finitely axiomatizable.

If an epistemic state fulfills both of these conditions we say it is *finitely representable*. The finite representation of an epistemic state may not be unique, since it depends on the choice of axiomatization of the partitions. Clearly, for any finite representation, the partitions of the entrenchment base will contain finite axiomatizations of the partitions of the underlying entrenchment, and the union of these bases will be the base of the belief set. The algorithms in the next chapter can be used to apply the AGM belief change operations to any finitely representable belief state. Note that by using the minimal change entrenchment revision policy described in the next section, the application of any AGM operation to a finitely representable belief state always yields another finitely representable belief state. Hence we have a solution to the problem of iterated revision [Spohn 1988; Boutillier 1993].

There exist theories which are not finitely axiomatizable, but they never occur as the result of an AGM operation on a finitely representable belief state, so that if the belief state is built from an empty belief set using the AGM operations, the resulting state is always finitely representable, for any finite number of operations.

For example, we show in chapter 6 that the most popular form of foundational reasoning can be performed using only five distinct levels of entrenchment, including the minimum and maximum levels for the non-beliefs and theorems respectively.

3.3 Conservatism and Independence of Beliefs

The most conservative entrenchment generated from an entrenchment base carries an implicit assumption of dependence between beliefs, particularly those which are equally ranked. For example, consider an entrenchment base Γ containing just two explicit beliefs, α and β, where $\mathbf{rank}(\Gamma, \alpha) = \mathbf{rank}(\Gamma, \beta)$. Then by the most conservative entrenchment, $\mathbf{rank}(\Gamma, \alpha\vee\beta) = \mathbf{rank}(\Gamma, \alpha)$, and so, by the (C–) condition, $\beta \notin K^{-}_{\alpha}$ and similarly $\alpha \notin K^{-}_{\beta}$. Thus α and β are implicitly dependent on each other, as evidenced by the behaviour of the contraction operation.

This behaviour cannot be avoided by assigning a unique rank to each belief in the base. Suppose the base Γ again contains two beliefs, this time with $\mathbf{rank}(\Gamma, \alpha) < \mathbf{rank}(\Gamma, \beta)$. From the (C–) condition we have $\beta \in K^-_\alpha$, but we still have $\alpha \notin K^-_\beta$, since $\mathbf{rank}(\Gamma, \alpha) < \mathbf{rank}(\Gamma, \beta) = \mathbf{rank}(\Gamma, \alpha\vee\beta)$. Thus in this case, the dependence is in only one direction: the lower ranked belief is implicitly dependent on the belief with the higher rank.

It is important to note that pairs of beliefs are not dependent of necessity; if we wish to express the fact that α and β are independent beliefs with the same ranks, then we should ensure that the entrenchment base contains $\alpha\vee\beta$ at a greater rank than either α or β. So the default behaviour of the most conservative entrenchment with the AGM operations is that explicit beliefs are dependent on the other explicit beliefs which have at least the same rank; if this behaviour is not desired, independence must be stated explicitly. The default behaviour is in accordance with the coherence theory of justification, which may not be suitable for all purposes, but we show in chapter 6 that it is possible to obtain foundational behaviour by automatically generating an entrenchment relation which encodes the independence of beliefs explicitly. We also show that the most conservative entrenchment generated in this way is still highly efficient with regards to the amount of explicit information needed by the system, in comparison with the explicit rank generation method of [Dixon & Foo 1993].

An important point to note with regard to dependence is that a minimal change in the entrenchment ordering does not necessarily correspond to a minimal change in the belief set. That is, conservatism with respect to the ordering on beliefs produces different results to conservatism with respect to the beliefs themselves. Consider, for example, a belief set over a first-order language, which contains the single sentence $\forall x\, P(x)$. Then suppose we wish to retract the belief $P(c)$, for some particular constant c. The minimal change to the belief set would still contain all beliefs $P(c')$, where $c'\neq c$, and could be described by the sentence $\forall x\, (x\neq c \rightarrow P(x))$. On the other hand, the most conservative entrenchment, in the absence of further information, would retract all instances of $P(x)$ simultaneously, as if the belief in one instance of the predicate P were dependent on the belief in all other instances of P. In chapter 7, we explore an alternative to the most conservative entrenchment, for the special case of default rules with exceptions, and show how the modified system can be used to implement nonmonotonic reasoning. For the general case, we justify our approach by requiring minimal change in the entire epistemic state, and not in the belief set alone. We show in chapter 4 that the algorithms for expansion, contraction and revision do provide the minimal change in the belief state which is consistent with the AGM postulates.

3.4 Entrenchment Revision

The AGM belief change operations are defined as mappings from belief states and formulae to belief sets. That is, the revision of a belief state by a formula maps the belief set with its entrenchment ordering to a revised belief set without specifying a revised entrenchment ordering. There are no constraints placed on the entrenchment of formulae in the belief sets resulting from revisions, contractions or expansions. This means that it is not possible to perform two successive contraction or revision operations without respecifying the entire entrenchment.

From the approaches of [Spohn 1988] (see chapter 2), we learn an important principle. The two approaches based on simple conditional functions (SCFs) are incorrect in the general case. The first method always accepts new information with maximal confidence, so that the new information is given a greater entrenchment than all other beliefs, and hence in subsequent belief changes, all previous beliefs are given up more easily than the most recently acquired belief. The assumption that new information is more reliable than everything that was believed previously is much too strong. The second method assigns a minimal entrenchment to new information, and thus any new belief is always believed with less confidence than all other beliefs. In this case, subsequent revision operations always destroy the information gained by the previous revision, except in the trivial case where the revision collapses to an expansion. Once more, this result is generally undesirable, although it was recently suggested as a means of performing iterated revision, under the name of *natural revision* [Boutillier 1993]. The lesson we learn is that an arbitrary assignment of an entrenchment level to a new belief cannot be suitable for a general theory of epistemic change.

Spohn's approach to belief revision based on ordinal conditional functions (OCFs) solves this problem, by requiring a degree of firmness to be given for any new belief when it is added to the belief set. We likewise define expansion and revision as operations which take an epistemic state, a new belief, and a rank for the new belief, and return the new epistemic state, where epistemic states are represented by entrenchment bases. A contraction does not require any rank to be specified, as no new beliefs are added to the belief set during a contraction operation.

We now describe our approach to specifying the entrenchment of the entire belief set that results from a belief change operation. Our idea is to interpret the principle of minimal change, originally formulated as applying to the contents of a belief set, as also applying to the entrenchment of formulae in the belief set. That is, we desire a minimal change in the entire belief state, not just in the contents of the belief set. This is also justified by the interpretation of the rank of a belief in our system as representing the degree of evidence for that belief. Under this interpretation, if a belief is not affected by a belief change operation, its degree of evidence should not be affected either. That is, a formula in the base which remains in the new belief base retains its rank unless an inconsistency with (R1) or (R2) forces it to be changed. This can also be viewed as a weak coherence approach to entrenchment revision, since ranks are only changed to avoid inconsistency.

As there is no way to automatically determine the degree of evidence associated with newly acquired information, the user must supply the rank of the added formula for an expansion or revision. But at this point we may face some subtle difficulties if the new belief is already a member of the belief set, based on whether or not the new evidence for the belief supercedes previous evidence, or is to be taken in addition to other evidence.

The AGM postulates imply that expansions and revisions by formulae which are already members of the belief set result in an unchanged belief set, but this principle should not be extended to apply to the whole belief state, in which case the ranking of beliefs would not be affected by the operation. Intuitively, to ignore a revision or expansion, irrespective of its rank, purely because there already exists *some* evidence for the belief, no matter how small, seems to be much too strong. Clearly, if the new rank for a belief is higher than its previous rank, this can be interpreted as new and stronger evidence for the belief, and the rank can be increased accordingly. But when the new rank is lower than the initial rank of the belief, there are two possible interpretations of the operation. If we interpret the expansion or revision as providing additional evidence for the belief, its rank is not changed, but if the operation is taken as referring to the total evidence for the belief, the operation is interpreted as discrediting previous evidence for the belief, and its rank ought to be reduced.

In the second case, it is not clear how the rank of a formula should be reduced. If the formula is explicitly in the base, and cannot be derived from other formulae in the base, then simply adjusting the rank of that formula is sufficient. But if the formula can be derived from other formulae in the base, then the ranks of some of these formulae may also need adjustment. Since the AGM postulates ensure that an expansion or a revision by a member of the belief set does not alter the belief set, we may not remove any of these justifying beliefs, but the rank of at least one of them must be reduced to be no greater than the intended rank of the new belief. Then we must choose which ones to adjust. To perform a minimal change, we could choose to reduce the rank of only those formulae which have the least rank for each derivation, which may be the best solution for this interpretation, but there is no reason for this to be the correct approach in the general case. Also, any such approach is computationally expensive, as it will generally require the generation of multiple derivations.

For these reasons, we interpret revision and expansion as presenting additional evidence for beliefs. The computational advantage is that the expansion or revision by an existing belief which already has a rank higher than the rank given in the operation makes no change to the epistemic state. Also, intuitively, if none of our beliefs are contradicted by a belief change operation, then we expect that the evidence for these beliefs, and also the evidence provided by them, should not be discredited by the operation.

Now we want to ensure that in the general case, after a belief change operation, the entrenchment base is still consistent with (R1) and (R2). (R2) is easily satisfied by disallowing operations which place formulae at a rank greater than or equal to the reserved maximum value. However, the addition of a new formula, or an increase in the rank of an existing formula, can lead to an entrenchment base which does not satisfy (R1). For example, if $\alpha \vee \beta$ is a belief in the base with rank r_1 and a revision to accept α with rank $r_2 > r_1$ is performed, the rank of formula $\alpha \vee \beta$ must be adjusted to regain consistency with (R1). Now **rank**$(\Gamma, \alpha \vee \beta)$ is (by entrenchment postulate (EE2)) at least r_2, since $\alpha \vdash \alpha \vee \beta$. To obtain the most conservative entrenchment, and since we have no greater evidence for $\alpha \vee \beta$, it is given a rank of exactly r_2. Note that this can be derived from the base without $\alpha \vee \beta$ being explicitly included in the base at rank r_2, and thus $\alpha \vee \beta$ may be deleted from the base, as it is derivable from α and its rank is thus automatically computed to be equal to that of α. In general, we require that the entrenchment base be kept free from redundancy, so any belief which can be derived from beliefs of equal or greater rank is removed from the base. This does not mean that the belief set base is nonredundant, since derived beliefs may have higher ranks than their derivations in the belief base. For example, a belief base containing α at rank r_1 and $\alpha \vee \beta$ at a greater rank r_2 is redundant as a belief base, but nonredundant as an entrenchment base.

The exact details of the entrenchment modification procedure are described along with the belief change algorithms in chapter 4, as the necessary adjustments to the entrenchment relation are performed within the belief change algorithms.

3.5 Summary

In this chapter we have presented a computational model of belief revision, designed for implementing a system satisfying the AGM rationality postulates. The belief set is represented by a finite base, from which all other beliefs can be derived using a sound and complete inference procedure. The base is partitioned into a finite entrenchment base, so that each belief in the base is assigned a rank, represented by a natural number. The complete entrenchment relation, known as the most conservative entrenchment, is then generated from the entrenchment base.

We argued that the most conservative entrenchment corresponds to our intuition of evidence, and that in the default case it corresponds to the coherence theory of justification, where beliefs are dependent on each other. We also showed that a foundational style of justification, where explicit beliefs are independent, can be obtained by entrenching the disjunction of pairs of independent beliefs higher than either of the beliefs themselves.

The final section of this chapter addressed the issue of entrenchment revision. Surprisingly little has been said in the belief revision literature about this area, despite the fact that it is a central issue in any computational model of belief revision. We presented an approach to entrenchment revision, which follows the principle of minimal change, and ensures that any consistent AGM operation on a finitely representable epistemic state can be modelled by an operation on an entrenchment base. We now describe the algorithms which implement these operations.

3.5

4 Algorithms for AGM Belief Revision

In this chapter, we present specific algorithms for the implementation of the AGM belief revision operations on a belief state represented by a finite entrenchment base. For each of the algorithms, expansion, contraction and revision, we explain in detail the operation of the algorithm, and also define conditions on the resulting belief state which guarantee the operations produce a minimal change in the belief state satisfying the AGM postulates. These conditions uniquely define the ranks of all formulae in the revised belief state. We then show that the algorithms do satisfy these conditions, and hence the algorithms are correct relative to the AGM postulates and the minimal change entrenchment revision policy. We also illustrate each algorithm with examples, and give an analysis of its worst case complexity relative to the given proof procedure. Finally, we describe some alternative algorithms, the theory base contraction operation from [Williams 1993] and the revision algorithm from [Dixon & Wobcke 1993].

The AGM belief change and entrenchment postulates are independent of the logic over which the belief change functions operate. The logic must satisfy certain minimal requirements; that it is consistent, compact, contains the Propositional Calculus and modus ponens, and satisfies the deduction theorem. For generality, the belief change algorithms described in this chapter may be used with any logic which satisfies these conditions and has an implementable proof procedure. In the algorithms we assume that we are given a logic of the user's choice, together with its proof procedure, and we analyse the complexity of the algorithms in terms of the number of calls to this proof procedure. A specific implementation over first-order logic is described in chapter 5.

The following conventions are used throughout this chapter: lower case Greek letters (α, β, $\cdots$) represent logical formulae; upper case Greek letters without subscripts (Γ, Δ, $\cdots$) are used for partitioned sets of logical formulae (entrenchment bases); upper case Greek letters with subscripts (Γ_1, Γ_2, $\cdots$) denote the members of the partitions (sets of logical formulae); and lower case italic letters (i, j, p, $\cdots$) are integers representing the ranks of beliefs.

For the purposes of this chapter, we shall assume that the entrenchment base, usually denoted Γ, is stored as a relation in a database, with two fields for the formula and the rank, and we assume the relation is indexed by the formula. The database can be modified by the following two operations: **update**(Γ, α, *newrank*) either adds the formula α to the relation Γ with rank *newrank* if α is not already contained in the relation, or changes the rank of α to *newrank* if α is already in the relation; and **delete**(Γ, α) deletes α from the relation Γ. The database may be queried by the operation **select**(Γ, α, *field_name*), which returns the value of the field *field_name* of the record with index α in the relation Γ.

4.1 Determination of Rank

Let the proof procedure associated with the given logic be denoted **prove**(Γ, α, r), which, given an entrenchment base $\Gamma = (\Gamma_1, \Gamma_2, \cdots, \Gamma_n)$, a formula α, and a rank r, determines the rank of α in Γ, if it is at least r. If the rank of α is less than r, then **prove** returns the value 0, otherwise it returns the value of the rank of α. Recall that the rank of a formula α in the belief set is given by:

$$\mathbf{rank}(\Gamma, \alpha) \; = \; \max \{ i : \overline{\Gamma_i} \vdash \alpha \},$$

where $\overline{\Gamma_i} \; = \; \bigcup_{j=i}^{n} \Gamma_j$. Equivalently, we may say that **prove**(Γ, α, r) returns the rank of α in the belief set $\overline{\Gamma_r}$.

The rank of a finite set Φ of formulae can then be defined as:

$$\mathbf{rank}(\Gamma, \Phi) \; = \; \min \{ \mathbf{rank}(\Gamma, \phi) : \phi \in \Phi \},$$

which, by entrenchment postulates (EE2) and (EE3), is also equal to the rank of the conjunction of all the formulae in the set Φ.

The rank of a single formula α can be determined by generating all proofs of α in some arbitrary order. Each proof has an associated rank, equal to the rank of the lowest ranked formulae involved in the proof, which is a lower bound for the rank of α. In generating all the proofs of α, successive attempts to prove α are restricted so that only formulae of greater rank than the current lower bound are used. When all proofs of α have been found, the greatest lower bound of the ranks is returned as the rank of α. If no proof is found, the rank of α is zero, indicating that α is not a current belief.

Thus the rank *r* which is supplied to the **prove** procedure is used to stop the procedure from attempting every proof of α in Γ. If the highest ranked proof were guaranteed to be found first, only one proof would need to be generated. By setting *r* to 0, **prove**(Γ, α, 0) can be used to determine whether or not α is a member of the belief set.

A request to determine the rank of α is processed by first checking whether $\alpha \in \Gamma_i$, for some *i*. If there is some such *i*, then this is the rank of α, and no further calculations have to be made. Thus to determine the rank of an explicit belief requires only a single database lookup. Note that this operation relies on the fact that an explicit belief cannot be derived from beliefs of strictly higher rank, which is ensured by condition (R1).

In the other case, when $\alpha \notin \Gamma_i$, for all *i*, the rank of α is the value returned by **prove**(Γ, α, 0). The complexity of this procedure is dependent on the choice of logic and inference procedure, and the algorithms in this chapter can be implemented over any logic with an implementable proof procedure, so it is the responsibility of the person implementing the algorithms to ensure the correct balance between expressibility and computability [Levesque & Brachman 1985] in the choice of logic.

The algorithm (in pseudo-code) to find the rank of the formula α in the base Γ is:

```
rank($\Gamma$, $\alpha$)
    if $\alpha$ is in $\Gamma$
        return select($\Gamma$, $\alpha$, rank)
    else
        return prove($\Gamma$, $\alpha$, 0)
```

4.2 Expansion

The AGM expansion operation is the simplest of the three belief change operations, as, in the AGM framework, it is calculated without using the entrenchment ordering. The result of an expansion is simply the set of consequences of the given belief set together with the new belief. That is, $K^+_\alpha = Cn(K \cup \{\alpha\})$. If $\overline{\Gamma_1}$ is a belief set base for *K*, that is $Cn(\overline{\Gamma_1}) = K$, then clearly $\overline{\Gamma_1} \cup \{\alpha\}$ is a belief set base for K^+_α.

Of course the AGM postulates for expansion specify only the resulting belief set, and not a complete belief state, whereas we wish to specify an algorithm which provides the entrenchment of the expanded belief set along with the belief set itself. The first problem is to allocate a rank to the new belief α. As argued in chapter 3, there is no way to generate ranks for new beliefs automatically for an arbitrary belief set. Therefore a new belief which is added to the belief set is always accompanied by a rank assigned by the user of the system. Accordingly, the expansion algorithm takes three arguments: the original belief state, the new belief, and the intended rank of the new belief.

Since we want the entrenchment ranking on an expanded belief set to be a minimal change of the ranks of the initial beliefs, it would be absurd to implement an expansion operation which is independent of the ordering on the given belief set. Instead, using the principle of minimal change as applied to entrenchment revision in section 3.4, we require the expansion operation to keep intact as many as possible of the ranks of formulae within the base. We must also ensure that the new entrenchment base satisfies conditions (R1) and (R2). In the rest of section 4.2, we show that the expansion operation and the minimal change principle can be implemented elegantly and efficiently, where the ranked base is interpreted as a most conservative entrenchment.

4.2.1 The Expansion Algorithm

Let **expand**(Γ, α, *newrank*) denote the expansion of an entrenchment base Γ by a belief α, where *newrank* is the intended rank of α. Then the following algorithm implements the AGM expansion operation, whilst also preserving the ranks of as many beliefs as possible, and satisfying conditions (R1) and (R2).

```
**expand**($\Gamma$,$\alpha$,*newrank*)
    if **rank**($\Gamma$, $\neg\alpha$) > 0   { $\alpha$ is inconsistent with $\Gamma$ }
        return($\Gamma$)
    else
        *oldrank* := **rank**($\Gamma$, $\alpha$)
        if *newrank* $\leq$ *oldrank*
            return($\Gamma$)
        else
            $\Delta$ := **update**($\Gamma$, $\alpha$, *newrank*)
            for each $\beta \in \overline{\Gamma_1}$ with *oldrank* $\leq$ **select**($\Gamma$, $\beta$, *rank*) $\leq$ *newrank*
                if **prove**($\Delta - \{\beta\}$, $\beta$, **select**($\Gamma$, $\beta$, *rank*) ) > 0
                    $\Delta$ := **delete**($\Delta$, $\beta$)
            return($\Delta$)
```

Some comments on this algorithm are in order. First, when the base is to be expanded by a formula, the formula is tested for logical consistency with the current base and the expansion rejected if it is not consistent. This disagrees with the AGM definitions, but we justify this on the basis that a user would not wish to have an inconsistent belief set, even in the case when the new data is inconsistent. If we allowed inconsistency in the belief set, then all formulae ranked no higher than either of the two inconsistent formulae would be removed from the base as being redundant. Thus, even when consistency was restored, much of the belief state would have been destroyed by the inconsistency. It is sufficient for a system to warn the user that an operation has been aborted on the grounds that it would have led to an inconsistent belief state.

Otherwise, if the formula is consistent with the belief set, its rank is checked for consistency with the rest of the entrenchment. The given rank *newrank* is taken to indicate further evidence for α, so if α already has rank at least *newrank*, no change is made to Γ. Otherwise, if the rank of α in Γ is less than *newrank*, then α is explicitly added to the base with rank *newrank* if it was not already in the base, and if it was already in the base, its rank is updated to be *newrank*.

Then we must ascertain whether any other changes to the base are necessary. According to the minimal change principle discussed in section 3.4, a change will only be necessary when the addition of the new formula α allows a new proof of one of the formulae β in the database, and the rank of the new proof is greater than the original rank of β. The rank of the new proof can be no greater than *newrank*, since α must be involved in the new proof, and thus formulae of rank higher than *newrank* need not be checked. Similarly, the rank of a formula β which is lower than the original rank of α, *oldrank*, cannot be affected by the increase in the rank of α. The reason for this is as follows: since *oldrank* is greater than the rank of β, it must be non-zero, and hence α was already believed in Γ before the expansion. Thus any proof of β which involves α must also use another belief in the proof which has a rank no greater than the rank of β (or else Γ did not satisfy (R1)), and so the rank of the proof is not affected by the change in rank of α. Finally, for beliefs β with rank between *oldrank* and *newrank*, if β is derivable from the other formulae in the new base with the same or higher rank, β is redundant and may be deleted. In fact, if all the formulae from which β can be derived are of higher rank than β, then condition (R1) requires that β be removed from the database.

Note that for each iteration of the loop, the test for whether β is derivable uses the base which may have been reduced after previous iterations of the loop have deleted some formulae. This does not affect the end result because a formula γ is deleted if and only if it is derivable from the formulae in the new base of at least the same rank. So β is derivable from the new base with some rank if and only if it is also derivable from the new base with the same rank after γ has been deleted. In other words, for any formula β and rank r, **prove**(Δ, β, r) remains constant throughout all iterations of the loop. This means that the expansion algorithm does not need to make a copy of the original base Γ but instead can repeatedly update the one database.

4.2.2 Correctness of the Expansion Algorithm

We now prove that the expansion algorithm is correct relative to the AGM postulates and our minimal change entrenchment policy. Let Δ = **expand**(Γ, α, r) be the entrenchment base produced by the expansion algorithm, where Γ is a consistent and nonredundant entrenchment base, α is a formula and r is a positive integer less than the maximum rank n.

For a consistent expansion (that is, $\overline{\Gamma_1} \nvdash \neg\alpha$), the AGM postulates require that the entrenchment base generated by the algorithm is a base for the belief set computed by the AGM expansion algorithm. That is:

$$Cn(\overline{\Delta_1}) = Cn(\overline{\Gamma_1})^{+}_{\alpha}$$

We also want to show that the new entrenchment relation calculated from Δ is as close as possible to the entrenchment relation derived from Γ. That is, the rank of any formula β in the new base Δ is different from its rank in Γ if and only if α combined with Γ provides higher ranked evidence for β than Γ alone. This will only occur when $\mathbf{rank}(\Gamma, \alpha\rightarrow\beta) > \mathbf{rank}(\Gamma, \beta)$ and $r > \mathbf{rank}(\Gamma, \beta)$. Thus the new evidence for β in the expanded belief set is given by $\min(\mathbf{rank}(\Gamma, \alpha\rightarrow\beta), r)$. If the new evidence for β has a higher rank than β had in Γ, this becomes the new rank of β, otherwise the rank of β remains unchanged. Stated formally, we define the (R+) condition:

$$\mathbf{rank}(\Delta, \beta) = \begin{cases} \min(\mathbf{rank}(\Gamma, \alpha\rightarrow\beta), r) & \text{if } \overline{\Gamma_1} \nvdash \neg\alpha \text{ and} \\ & \quad \mathbf{rank}(\Gamma, \beta) < \min(\mathbf{rank}(\Gamma, \alpha\rightarrow\beta), r) \\ \mathbf{rank}(\Gamma, \beta) & \text{otherwise} \end{cases}$$

The expansion algorithm is correct if it satisfies the (R+) condition. In appendix B, we prove that the expansion algorithm satisfies the (R+) condition, and therefore it implements the AGM expansion function and computes a new entrenchment base with minimal change to the entrenchment relation.

The correctness proof is based on the following formal specification of the expansion algorithm. Let $\Delta = \mathbf{expand}(\Gamma, \alpha, r)$, where Γ is a consistent and nonredundant entrenchment base, α is a formula and *r* is a positive integer less than the maximum rank *n*. Then if $\Gamma \vdash \neg\alpha$ or $\mathbf{rank}(\Gamma, \alpha) \geq r$, the base is returned unchanged, and we have $\Delta_i = \Gamma_i$, for all *i* such that $1 \leq i \leq n$. Otherwise the **expand** algorithm computes, for all *i* with $1 \leq i \leq n$:

$$\begin{aligned} \Delta_i = \; & \{\beta\in\Gamma_i : \beta\neq\alpha \wedge (\mathbf{rank}(\Gamma, \beta) > r)\} \;\cup \\ & \{\beta\in\Gamma_i : \beta\neq\alpha \wedge (\mathbf{rank}(\Gamma, \beta) < \mathbf{rank}(\Gamma, \alpha))\} \;\cup \\ & \{\beta\in\Gamma_i : \beta\neq\alpha \wedge ((\overline{\Gamma_i}-\{\beta\}) \cup \{\alpha\} \nvdash \beta)\} \;\cup \\ & \{\alpha : i=r\} \end{aligned}$$

4.2.3 Examples

In this section we present a series of expansions which illustrate how the algorithm works. We shall represent the entrenchment base, Γ, by a list of all of the formulae in Γ, in order of decreasing rank. For each formula, we will write the rank followed by a colon followed by the formula, for example, 40 : $a \vee b$. Assume that Γ is initially empty.

The first operation we perform is **expand**(Γ, $a \vee b$, 40), which results in the following entrenchment base:

40 : $a \vee b$

Suppose we then perform the expansion **expand**(Γ, a, 20). Since $a \vee b$ has a higher rank than the new formula, it cannot be affected by the operation. Hence the new base Γ is:

40 : $a \vee b$
20 : a

The next operation we attempt is **expand**(Γ, $a \vee c$, 10). This time, $oldrank = 20$, because $a \vdash a \vee c$, so $oldrank \geq newrank$, and Γ is returned unchanged.

Now let us expand Γ with **expand**(Γ, $a \wedge b$, 30). Since $a \wedge b$ is not derivable from the formulae in Γ, *oldrank* is 0. After performing the update, we test all formulae in Γ with a rank between 0 and 30, which in this case is just the formula a, to see if any of these formulae are derivable from the rest of the new base with a greater rank than they originally had. The call to **prove**($\Delta - \{a\}$, 30, a) succeeds, returning 30, since $a \wedge b \vdash a$, and hence a is deleted from the database. The resulting base is:

40 : $a \vee b$
30 : $a \wedge b$

Finally, suppose we request the operation **expand**(Γ, $\neg b$, 20). Then **rank**(Γ, b) returns 30, indicating that $\neg b$ is inconsistent with Γ, and so the operation is aborted and Γ is left unchanged.

4.2.4 Complexity of Expansion

Although the expansion of a belief set can be implemented by a single database update, considerably more computation is required to ensure a nonredundant and consistent entrenchment base. We assume in this and subsequent discussion of complexity that arithmetic operations, comparisons, and database operations are negligible compared to the complexity of the procedure **prove**. Therefore all complexity will be analysed with respect to the number of calls to **prove**.

In the nontrivial case, where α is consistent with K, there are 2 calls to **prove** outside the loop, and either one call or no calls to **prove** for each iteration of the loop. Note that the condition tested on the head of the loop, that the rank of β is between *oldrank* and *newrank*, does not require any call to **prove**, because β is in the entrenchment base, and hence its rank is stored explicitly in the database. In the worst case, where the added formula is given a rank at least as high all other formulae in the entrenchment base, and its previous rank was not greater than any other formula in the entrenchment base, then $m+2$ calls to the procedure **prove** are made, where m is the number of formulae in the entrenchment base. That is, for an expansion, the number of calls to **prove** is linear in the size of the base.

As we might expect, it is more difficult to add a new belief with a high rank than with a low rank, as this represents a more radical alteration of the epistemic state. The difficulty of accepting new evidence for a belief is related to the increase in the rank of the belief; in fact, it is proportional to the number of beliefs with ranks between the old and new values of the rank. Therefore, the difference between these two values could be taken to be a measure of the degree of epistemic change.

Note that when creating a database from the empty belief set, the efficiency can be greatly increased by inserting the formulae in descending rank. Then the average case performance, rather than being quadratic in the size of the created database, can be linear if we assume a fixed limit on the number of formulae at any one rank.

The expansion algorithm is very efficient with regard to space. In the worst case, the database increases in size by one formula, as we would expect. By removing redundancies, it is often the case that the base for the expanded belief set is in fact smaller than the original base. The working space required is also negligible, apart from the space used by **prove**. As noted earlier, we do not need to make a copy of the given base Γ, but can perform all operations directly on Γ, so that Δ and Γ are aliases for the same database relation.

4.3 Contraction

The second AGM operation we shall specify is contraction. The contraction of a belief set is calculated from the belief set and the entrenchment via the (C–) condition:

$$\beta \in K^-_\alpha \text{ iff } \beta \in K \text{ and either } \vdash \alpha \text{ or } \alpha <_E \alpha \vee \beta$$

The main difficulty in providing an algorithm for contraction on finitely representable belief sets is that although $K^-_\alpha \subseteq K$, for all α and K, it is often the case that no subset of the base Γ is a base for K^-_α. Alternatives to the AGM contraction operation, which have the property that the contracted base is a subset of the original base, do not satisfy the AGM recovery postulate (K–5). An example of this is the theory base contraction operation of [Williams 1993]. [Nebel 1989] suggested that to satisfy the recovery postulate, it is sufficient to add the formula $\alpha \rightarrow \gamma$, where γ is a conjunction of all the formulae in Γ. Although this is true, it effectively doubles the size of the database each time a contraction is performed, and hence is not desirable for a computational model of contraction. All that is necessary is that for each formula β which is removed from the base, the formula $\alpha \rightarrow \beta$ must be derivable from the new base. The simplest way to ensure this condition is to replace each deleted formula β with the formula $\alpha \rightarrow \beta$ at the same rank, unless it is already derivable from the belief set with at least the same rank, and hence is redundant.

The reason that it is necessary to replace β by $\alpha \rightarrow \beta$ is that from the recovery postulate, $K \subseteq (K^-_\alpha)^+_\alpha$, we can derive that if $\beta \in K$ then $K^-_\alpha \vdash \alpha \rightarrow \beta$. (If $\beta \in K$, $K^-_\alpha \cup \{\alpha\} \vdash \beta$ by (K–5), so $K^-_\alpha \vdash \alpha \rightarrow \beta$ by the deduction theorem.)

Entrenchment revision is not a difficult issue for contraction, since no new beliefs are added to the belief set. That is, we can simply insist on a minimal change policy which states that all remaining beliefs in a contracted belief set must have exactly the same entrenchment as they had before the contraction was performed.

We will now describe such a contraction algorithm, and then show that it does satisfy all of the AGM postulates for contraction.

4.3.1 The Contraction Algorithm

Let **contract**(Γ, α) denote the contraction of the entrenchment base Γ by the formula α. Recall that *n* is the number of elements of the partition Γ, and is also equal to the rank of the theorems.

```
contract($\Gamma, \alpha$)
    if rank($\Gamma, \alpha$) = $n$        { $\alpha$ is a theorem }
        return($\Gamma$)
    else
        $\Delta := \Gamma$
        $oldrank$ := rank($\Gamma, \alpha$)
        for each $\beta \in \Gamma$ such that select($\Gamma, \beta, rank$) $\leq oldrank$
            if prove($\Delta, \alpha \vee \beta, oldrank+1$) = 0  { (C–) condition }
                $r$ := select($\Delta, \beta, rank$)
                $\Delta$ := delete($\Delta, \beta$)
                if $r < oldrank$ or prove($\Delta, \alpha \rightarrow \beta, oldrank+1$) = 0
                    $\Delta$ := update($\Delta, \alpha \rightarrow \beta, r$)
        return($\Delta$)
```

If an attempt is made to contract a theorem, the attempt is rejected in accordance with the AGM definitions. Otherwise, a change is made to the base if and only if α is derivable from the base. In this case, each member β of the base with rank less than or equal to α is checked to see whether β is to be deleted from the base, and further, if $\alpha \rightarrow \beta$ is to be added to the base. By the (C–) condition, β is removed from the base if $\alpha =_E \alpha \vee \beta$. (Note that $\alpha >_E \alpha \vee \beta$ is impossible because it contradicts entrenchment postulate (EE2).) In order to satisfy the recovery postulate, the formula $\alpha \rightarrow \beta$ must be in the contracted belief set. Therefore it needs to be added to the base if it is not already derivable in the new base. By the minimal change principle, it is given the same rank as that of β in the original base. This is the appropriate rank because, given that $\alpha \rightarrow \beta$ is not provable from the set of formulae ranked higher than β, its rank in Γ is the same as the rank of β in Γ, since $\beta \vdash \alpha \rightarrow \beta$. Also, if α is added back to the belief set at its original rank, then the rank of β will also be restored to its original value.

As shown in the algorithm, this test for whether $\alpha \rightarrow \beta$ is derivable in the new base is greatly simplified. In the case where β is ranked less than α, given that $\alpha =_E \alpha \vee \beta$, $\alpha \rightarrow \beta$ must be ranked at the same level as β by (EE2) and (EE3), hence $\alpha \rightarrow \beta$ cannot be derivable after β is deleted since the original base Γ is not redundant. The other case is where α, β and $\alpha \vee \beta$ are all ranked the same in Γ. In this case $\alpha \rightarrow \beta$ must be added to the base with that rank if it is not already derivable from the set of higher ranked formulae. This is the only case in which it is necessary to make the second call to **prove**. Note that this algorithm repeatedly tests the ranks of formulae in Δ rather than in the original Γ as the (C–) condition requires. Again, as with expansion, this does not affect the result because of the nature of the tests performed. The first test is for whether there is a proof of $\alpha \vee \beta$ using only formulae ranked higher than α; the second is for whether there is a proof of $\alpha \rightarrow \beta$ using the same set of formulae. Since none of these formulae are affected by earlier iterations of the loop, this set of formulae is the same in each Δ as in the original Γ.

Note that the contraction of α is successful even if α is not contained in the base: in this case, at least one formula from the base contributing to each proof of α is removed. This is ensured by the AGM contraction and entrenchment postulates.

4.3.2 Correctness of the Contraction Algorithm

We can now prove that the contraction operation defined by the above algorithm coincides exactly with the AGM contraction operation, and also performs a minimal change to the entrenchment ordering. Firstly we must show that the set of all formulae in the new base Δ is a base for the theory obtained by applying the AGM contraction operation to the logical closure of the beliefs in the base Γ. That is:

$$Cn(\overline{\Delta_1}) = Cn(\overline{\Gamma_1})^{-}_{\alpha}$$

This guarantees that the algorithm satisfies all of the contraction postulates. We also require that the new entrenchment base represents a minimal change in the entrenchment ordering and also satisfies (R1) and (R2). A minimal change in the entrenchment is obtained if the rank of each formula remaining in the belief set after the contraction is the same as its rank before the contraction. These requirements are entailed by the (R–) condition:

$$\mathbf{rank}(\Delta, \beta) = \begin{cases} \mathbf{rank}(\Gamma, \beta) & \text{if } \vdash\alpha \text{ or } \mathbf{rank}(\Gamma, \alpha) < \mathbf{rank}(\Gamma, \alpha\vee\beta) \\ 0 & \text{otherwise} \end{cases}$$

The proof that the contraction algorithm satisfies this condition can be found in appendix C.

We now give a formal specification of the contraction algorithm. If Γ is a consistent and nonredundant entrenchment base, α a formula, and $\Delta = \mathbf{contract}(\Gamma, \alpha)$, then for all *i* such that $1 \le i \le n$:

$$\begin{aligned} \Delta_i = \{&\beta\in\Gamma_i : \mathbf{rank}(\Gamma, \alpha) < \mathbf{rank}(\Gamma, \alpha\vee\beta)\} \cup \\ \{&\alpha\to\beta : \beta\in\Gamma_i \wedge \mathbf{rank}(\Gamma, \alpha) = \mathbf{rank}(\Gamma, \alpha\vee\beta) \wedge \\ &(\mathbf{rank}(\Gamma, \beta) < \mathbf{rank}(\Gamma, \alpha) \vee \mathbf{rank}(\Gamma, \alpha) = \mathbf{rank}(\Gamma, \alpha\to\beta))\} \end{aligned}$$

4.3.3 Examples

We shall now illustrate the operation of the contraction algorithm with several examples. Assume we have the entrenchment base Γ given by:

30 : $a\wedge b\to c$
20 : a
10 : b

Suppose first that we wish to remove the explicit belief *a*, so we call **contract**(Γ, *a*). Then for each of the formulae in the base with rank 20 or less, we test whether that belief is to remain in the base. The result is that *a* is removed, and *b* is replaced by $a\to b$, giving the entrenchment base:

30 : $a\wedge b\to c$
10 : $a\to b$

For the second example, let us remove the implicit belief *c* from the original belief set, using **contract**(Γ, *c*). The most conservative entrenchment gives the rank of *c* as 10, so only the formula *b* is affected, giving:

30 : $a\wedge b\to c$
20 : a
10 : $c\to b$

Third, we show an example where the implication $\alpha\rightarrow\beta$ is redundant. Consider the following base Γ:

$30 : a$

$20 : b$

$10 : a\wedge b\rightarrow c$

Now we apply **contract**(Γ, *c*), where the rank of *c* is 10, and so only the formula $a\wedge b\rightarrow c$ can be affected. Since $c\rightarrow(a\wedge b\rightarrow c)$ is a logical theorem, it is redundant, so $a\wedge b\rightarrow c$ is not replaced in the base. Thus the new base is:

$30 : a$

$20 : b$

4.3.4 Complexity of Contraction

Once more, we consider the complexity of the algorithm in terms of the number of calls to **prove**. Outside the loop, there is one call to **prove**, to calculate the rank of α. This value can be stored in *oldrank* when it is first calculated, rather than computing its value twice. Note that the database assignment operation, $\Delta := \Gamma$, does not need to copy the contents of the database, but merely assigns an alias to the database relation. The loop is iterated once for each member of Γ with a rank less than or equal to *oldrank*. This rank needs no calculation, as it is stored explicitly in the database. Within the loop, there are at most 2 calls to **prove**, to calculate the ranks of $a\vee b$ and $a\rightarrow b$ respectively. So in the worst case, the algorithm requires $2m+1$ calls to **prove**, where there are *m* formulae in the database. As was true for expansion, the number of calls to **prove** is linear in the size of the base.

It is also interesting to note that the complexity is directly proportional to the relative rank of the formula which is being removed. This is because a contraction cannot affect formulae with greater rank than the formula being removed, and agrees with the intuition that it is more difficult to remove highly entrenched beliefs than those with a relatively low rank. Thus the initial rank of a contracted formula can be used as a measure of the degree of epistemic change caused by the contraction.

The contraction algorithm is similar to expansion with regard to space efficiency. After a contraction, the number of formulae in the database is no greater than were there previously, although some formulae have been made longer by substituting $\alpha \to \beta$ for β. The working space required is again constant, not taking into account the space used by **prove**, since once more we may perform all operations directly on the base Γ, without making a copy of it.

4.3.5 Theory Base Contraction

As an alternative to the AGM contraction operation, we describe an algorithm which implements the theory base contraction function of [Williams 1993]. It is interesting to note that this algorithm only differs from the AGM algorithm in that it does not replace the deleted formula β with the formula $\alpha \to \beta$. Hence the operation is not fully rational, as it does not satisfy the recovery postulate. Nevertheless, the revision function derived from this contraction operation via the Levi Identity does satisfy all of the AGM requirements.

This form of contraction can be described as a syntax-based approach, as the principle of minimal change in the belief state has been replaced by a principle of minimal change in the syntactic form of the base.

Let the contraction of a theory base Γ by a formula α be given by **TB_contract**(Γ, α), defined as:

```
TB_contract($\Gamma, \alpha$)
    if rank($\Gamma, \alpha$) = $n$        { $\alpha$ is a theorem }
        return($\Gamma$)
    else
        $oldrank$ := rank($\Gamma, \alpha$)
        for each $\beta \in \Gamma$ such that select($\Gamma, \beta, rank$) $\le oldrank$
            if prove($\Gamma, \alpha \vee \beta, oldrank + 1$)) $= 0$  { (C–) condition }
                $\Gamma$ := delete($\Gamma, \beta$)
        return($\Gamma$)
```

This approach gains a higher level of efficiency for contraction than its AGM counterpart, as there are at most $m+1$ calls to **prove** using this algorithm, about half the number required for the full AGM contraction operation.

4.4 Revision

We now present our specification of an AGM revision algorithm. The revision of a belief set in the AGM paradigm can be described directly, via the (C∗) condition:

$$\beta \in K^*_\alpha \text{ iff } \vdash \neg\alpha \text{ or } \neg\alpha <_E \alpha \rightarrow \beta$$

It is also possible to define the revision operation in terms of the contraction and expansion operations, via the Levi identity:

$$K^*_\alpha = (K^-_{\neg\alpha})^+_\alpha$$

It is the latter definition that we shall use. Once more, we shall show that our algorithm executes an AGM revision with a minimal change in the belief state.

4.4.1 The Revision Algorithm

The algorithm **revise**(Γ, α, *newrank*) for revising a belief base Γ by a formula α with rank *newrank,* is the composition of a contraction and an expansion, as specified by the Levi identity.

```
revise(Γ, α, newrank)
        return(expand(contract(Γ, ¬α), α, newrank))
```

4.4.2 Correctness of the Revision Algorithm

Now we show that the algorithm defined in this section satisfies the AGM postulates for rational belief change. Let Γ be a consistent and nonredundant entrenchment base, α a consistent formula, and *r* a natural number less than the maximum rank *n*. Then if Δ = **revise**(Γ, α, *r*), the AGM postulates are satisfied if

$$Cn(\overline{\Delta_1}) = Cn(\overline{\Gamma_1})^*_\alpha$$

To define the (R∗) condition for minimal change in the belief state, we first note that for an inconsistent ($\vdash \neg\alpha$) or trivial ($\vdash \alpha$) revision, Γ is returned unchanged. Otherwise, we combine the (R+), (R–) and (C∗) conditions to give the (R∗) condition:

$$\mathbf{rank}(\Delta, \beta) = \begin{cases} 0 & \text{if } \mathbf{rank}(\Gamma, \neg\alpha) \geq \mathbf{rank}(\Gamma, \alpha\rightarrow\beta) \\ \mathbf{rank}(\Gamma, \beta) & \text{if } \mathbf{rank}(\Gamma, \neg\alpha) < \mathbf{rank}(\Gamma, \alpha\rightarrow\beta) \\ & \quad \text{and } (\mathbf{rank}(\Gamma, \alpha\rightarrow\beta) \leq \mathbf{rank}(\Gamma, \beta) \\ & \qquad \text{or } r \leq \mathbf{rank}(\Gamma, \beta)\,) \\ \min(\mathbf{rank}(\Gamma, \alpha\rightarrow\beta), r) & \text{otherwise} \end{cases}$$

We do not give a separate formal specification of the revision algorithm, since it is a composition of the expansion and contraction operations, which have already been defined precisely. In appendix D, we prove that the revision algorithm satisfies the (R$*$) condition, and thus it computes the AGM revision operation with a minimal change to the epistemic state.

4.4.3 Examples

Consider an initial database Γ containing $a\rightarrow b$ at rank 30, and suppose it is revised by **revise**(Γ, $\neg b$, 20). Then, since Γ is consistent with $\neg b$, the operation is equivalent to an expansion, and results in the following base:

30 : $a \rightarrow b$
20 : $\neg b$

Now suppose we perform **revise**(Γ, a, 10). Then the rank of $\neg a$ is 20, and the formula $\neg b$ is replaced with $\neg a \rightarrow \neg b$, since the rank of $\neg a \vee \neg b$ is also 20. The revised entrenchment base is:

30 : $a \rightarrow b$
20 : $\neg a \rightarrow \neg b$
10 : a

4.4.4 Complexity of Revision

Clearly the revision algorithm can be analysed in terms of the contraction and expansion operations from which it is made up. The worst case number of calls to **prove** is $3m+3$, for an entrenchment base containing m formulae, which again is linear in the size of the entrenchment base. Similarly, the space requirements are the same as for contraction and expansion; the entrenchment base nevers grows by more than one formula.

The difficulty of performing a revision by a formula α is dependent on both the initial entrenchment of $\neg\alpha$ and the intended entrenchment of α in the revised belief base. In the first part of the revision, the amount of work performed increases approximately linearly with the relative entrenchment of $\neg\alpha$. On the other hand, if $\neg\alpha \notin K$, its rank is zero, and the revision collapses to an expansion operation. Similarly, the work required in the second half of the revision is proportional to the requested entrenchment of α. So for revision, we could use the sum of these two ranks as a measure of the degree of epistemic change.

4.4.5 Fast AGM Revision

It is possible to find a more efficient algorithm for calculating the revision directly, rather than via the Levi identity, which uses AGM contraction and expansion operations. It has been noted several times [Makinson 1987; Nebel 1989; Williams 1993] that many contraction functions that are simpler than AGM contraction may be substituted into the Levi identity without affecting the resulting revision operation. Such contraction functions are called *revision equivalent* to AGM contraction; an example is the theory base operation of [Williams 1993], which does not satisfy the AGM recovery postulate, but when used in the Levi identity to implement revision, satisfies all of the AGM revision postulates.

The reason that supposedly "irrational" contraction functions can produce "rational" revision functions is explained by considering the difference between the AGM contraction and the theory base contraction algorithms presented in the previous section. Recall that for AGM contraction, when a belief β is removed from the base, it usually has to be replaced by $\alpha\rightarrow\beta$, where α is the sentence being contracted. In the case of a revision, where the negation of the contracted belief is added to the base, the implications of the form $\alpha\rightarrow\beta$ are derivable from the newly added sentence, so they do not need to be explicitly included in the base. Apart from this difference, such a revision algorithm bears a close resemblance to the composition of an AGM contraction and expansion algorithm. The algorithm **fast_revise** is an example of such a function.

Unfortunately, the revision equivalent contraction functions do not satisfy the minimal change requirement for the entrenchment relation. This is due to the fact the rank of $\alpha\rightarrow\beta$ may be lower in the revised belief state than it was in the given belief state, which occurs when *newrank* is less than **rank**$(\Gamma, \alpha \rightarrow \beta)$, and thus the rank of other derived beliefs may also be reduced. Hence our AGM revision algorithm makes the minimal change to the epistemic state consistent with the postulates for revision, whereas the **fast_revise** algorithm makes a minimal change to the syntactic form of the entrenchment base.

We now present the algorithm **fast_revise**$(\Gamma, \alpha, newrank)$ for revising a belief base Γ by a formula α with rank *newrank*. Recall that *n* is the rank of the theorems.

```
fast_revise($\Gamma$, $\alpha$, newrank)
    if rank($\Gamma$, $\alpha$) $= n$ or rank($\Gamma$, $\neg\alpha$) $= n$
    { $\alpha$ either a theorem or contradiction }
        return($\Gamma$)
    else
        $\Delta := \Gamma$
        oldrank := rank($\Gamma$, $\neg\alpha$)
        for each $\beta \in \Gamma$ such that select($\Delta$, $\beta$, rank) $\leq$ oldrank do
            if prove($\Delta$, $\alpha \rightarrow \beta$, oldrank+1) ) $= 0$ { (C$*$) condition }
                $\Delta :=$ delete($\Delta$, $\beta$)
        $\Delta :=$ expand($\Delta$, $\alpha$, newrank)
        return($\Delta$)
```

If α is a theorem, the revision is trivial and no change is made to the base. If α is inconsistent, the operation is disallowed for the same reasons as those for disallowing inconsistent expansions. (Note that the AGM postulates imply that the revision by an inconsistent belief is the same as the expansion by that belief.) Otherwise, by the (C$*$) condition, we remove from the base any formula β such that $\neg\alpha$ is of rank equal to $\alpha \rightarrow \beta$ in the original base. It suffices to consider only those β ranked less than or equal to $\neg\alpha$ because if $\neg\alpha <_E \beta$ then $\neg\alpha <_E \alpha \rightarrow \beta$ (by EE2 and EE1), so β remains in the revised set. Finally, the set of formulae remaining in the base is expanded by α.

Note that, again, the repeated calls to **prove** use only the formulae ranked higher than $\neg\alpha$, and this set does not change with repetitions of the loop, so only one copy of the database is required to implement the algorithm.

This revision operation can be analysed in much the same way as the contraction; there are two calls to **prove** outside the loop, and one call for each iteration of the loop. For a database with m formulae, this gives us a worst case of $2m+4$ calls to **prove**, including those made from the **expand** algorithm. This is considerably better than the $3m+3$ calls made in the worst case by invoking the Levi identity with the AGM contraction operation, reducing the time cost of revision by a third.

In terms of space, this revision algorithm is very efficient. No more than one formula is added to the base, and, unless the revision collapses to an expansion operation, at least one formula is removed from the base. Once more the only notable temporary space required is for the **prove** algorithm.

4.5 Summary

We have presented a complete computational model for AGM belief revision, in the form of algorithms for the efficient computation of expansion, contraction and revision operations. These algorithms are based on the interpretation of a partially specified entrenchment as standing for the most conservative entrenchment, and satisfy the AGM postulates for rational belief change.

The algorithms also can be implemented easily over a theorem prover for the chosen logic. We have shown that the algorithms are efficient in terms of space and time; the entrenchment base never grows by more than one formula per operation, and the number of calls to the proof procedure is linear in the size of the entrenchment base. In the following chapter we describe a specific implementation of these algorithms over first-order logic.

5 A First-Order Logic AGM Belief Revision System

In this chapter, we describe the implementation of a belief revision system, built over classical first-order logic with equality, which satisfies all of the AGM postulates for rational belief change, for any finitely representable belief state. The internal representation of the belief set is in conjunctive normal form, and logical consequence is computed by a resolution theorem prover using ordered linear resolution (OL-resolution) and paramodulation [Chang & Lee 1973]. The OL-resolution procedure determines the rank of a formula by generating, in a depth-first fashion, the highest ranked OL-refutation of the negation of the formula. The program uses the algorithms described in chapter 4 to implement the AGM belief revision operations. This is the first implementation of a general AGM belief revision system.

First, the data structures and other details of the implementation will be described, followed by a section which outlines the resolution theorem prover and a further section on the implementation of the equality predicate by adding paramodulation to the resolution procedure. The interface to the system is then explained, including the command-line options, and the syntax of commands and formulae. Finally, the operation of the system is illustrated by several examples.

5.1 Internal Data Representation and Code

As described in chapter 3, a belief set is represented by a finite set of formulae in the chosen logic which, in this case, is classical first-order logic with equality. The input to the program is not restricted to any particular representation, as the system accepts any well-formed formulae, but stores the formulae in the database in conjunctive normal form (CNF) for use by the resolution theorem prover. Therefore, formulae are parsed and converted into CNF as they are entered into the system. This is done by standard techniques; each formula is parsed by a recursive descent parser, creating a parse tree which is manipulated to convert the formula into CNF. The conversion into CNF is achieved by applying the following steps:

Algorithm Convert_to_CNF

```
remove implications
move negations in
rename variables (each variable name is replaced by a unique identifier)
replace existentially quantified variables by Skolem functions and constants
remove universal quantifiers
recursively apply distributive laws
```

The resulting formula, in CNF, is stored in the database for use by the resolution theorem prover.

The database contains a linked list of records representing the formulae, sorted in descending order of rank. Each of these records contains three different representations of the formula. First there is the character string, in the same format as the formula was input, and then the CNF forms of the formula and the negation of the formula. The clausal form of the negation is needed by the refutation procedure when querying the formula, and this cannot be calculated directly from the clausal form of the formula because of information lost in Skolemization. (In particular, after Skolemization, it is not possible in general to ascertain whether universally quantified variables appeared within the scope of any existentially quantified variables.)

Each formula is represented by a linked list of records, each of which represents a clause. The records contain a pointer to the data structure representing the clause, an integer representing the rank of the clause, an integer which is used to count how many times the clause has been used in the current proof, a pointer to a data structure containing the substitutions which have been applied to the formula, and a pointer to the next clause in the formula.

Clauses are also represented by linked lists of records, where each of these records represents a literal. The information stored for each literal is the name of the literal, a flag indicating whether it is a positive or negative literal, a flag indicating whether or not the literal is framed (by the OL-deduction procedure), a pointer to a list of arguments, a pointer to a list of previous forms of the literal to increase the efficiency of paramodulation, and a pointer to the next literal in the clause.

The occurrence of literals within clauses is recorded by an entry in a hash table, which is used by the resolution procedure to avoid searching for literals to resolve against. For small databases, this has not produced a measurable increase in efficiency, but for larger databases we expect that it would provide much better performance.

Terms, which occur as arguments to literals and arguments to other terms, have a data structure containing the name of the term, a pointer to the list of the term's arguments, a pointer to the next term in the list in which the current term resides, and a unique identifying integer, which falls into one of three classes. If this value is 0, the term is a constant or function symbol; if it is negative, it represents a unique Skolem constant or function; and if the number is positive, it represents a unique variable name. These values are always invisible to the user of the system, but they are necessary to ensure consistent substitutions for variables.

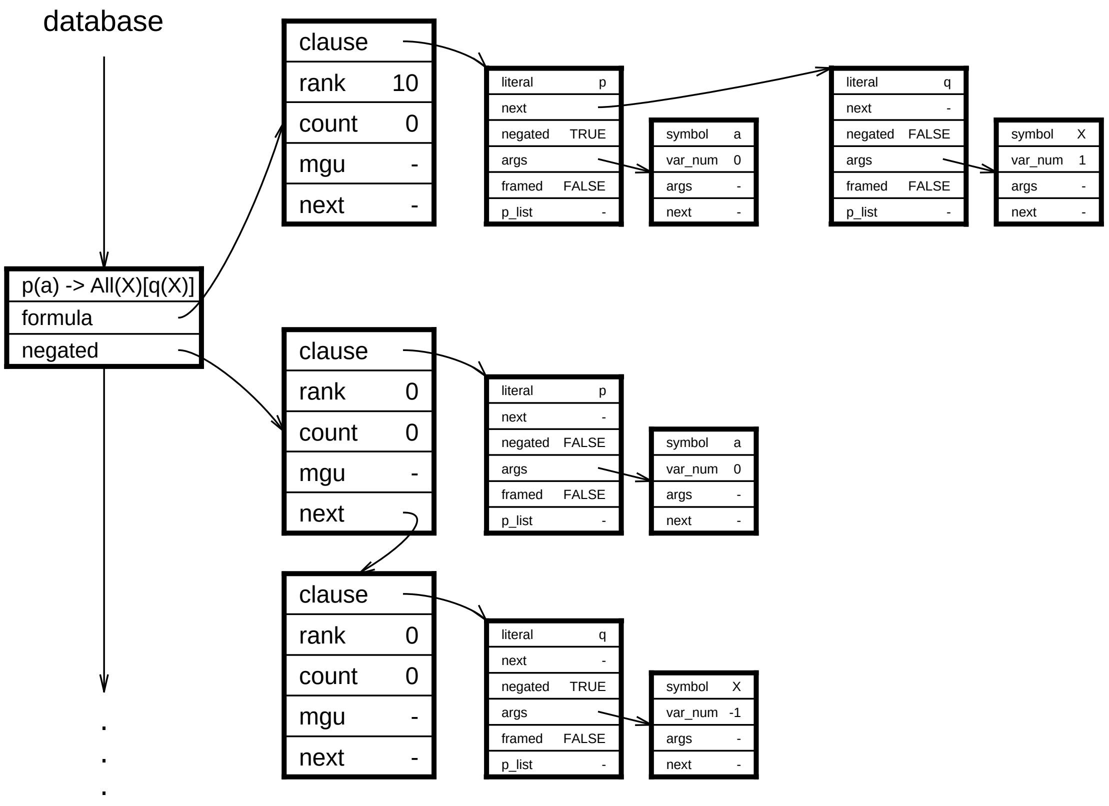


Figure 5.1: The AGM data structure for $p(a) \rightarrow \forall x.q(x)$

The code itself is written in about 4000 lines of standard C; on a MIPS R4000, it compiles into about 100K of object code, and all of the examples in this thesis run in negligible time (less than one second). Efficiency considerations were taken into account in the design of the program, but due to the undecidability of first-order logic, for practical applications a restricted logic should be used, such as Horn clause logic, Datalog, propositional logic, or any tractable subset of first-order logic, implemented with a theorem prover that takes advantage of the restrictions in the logic.

5.2 The First-Order Logic Proof Procedure

It is well known that there is no sound and complete decision procedure for first-order derivability. OL-refutation guarantees soundness and completeness of the inference method, but does not guarantee termination. For the system to be useful, we would like to be able to guarantee not just termination, but termination in a reasonable time for the size of the database, without necessarily losing the correctness of our results. Our system provides two different ways of dealing with this problem. The first option is to allow the search for a refutation to proceed indefinitely. This ensures that all results obtained are correct, but does not ensure that any result ever will be obtained, which, for a practical system, is not a satisfactory situation.

The second approach allows the user to place a limit on the number of times each clause can be used as a side clause in a resolution proof. Once this limit is reached, a warning message is printed and that clause is not used again in the rest of the proof. This guarantees termination at the cost of completeness, but note that the user is always warned when a proof has been cut short in this fashion, and therefore knows which particular results are not totally reliable. The limit on the number of uses of each clause is a parameter which can be adjusted dynamically; used in this way, the theorem prover would then have a depth-first iterative deepening search strategy. This facility has proved to be adequate for stopping infinite recursion in simple problems such as a transitive closure query for an ancestor relation. It could also be used in list processing or arithmetic problems, where the length of the list, or greatest number of interest, respectively, has a fixed upper bound. For more general problems, this approach cannot be used.

In chapter 4, we presented the algorithm for determining the rank of a belief. This algorithm relied on the existence of a procedure called **prove**(Γ, α, r), which returns the value of the highest ranked proof of α from the entrenchment base Γ, ignoring formulae of rank less than *r*. That is, the proofs of α only use formulae in $\overline{\Gamma_r}$ as defined in section 3.2. This procedure is implemented using a modified ordered linear resolution (OL-resolution) theorem prover.

OL-resolution employs a refutation proof procedure which starts with the negation of the clause to be proved as the centre clause, and resolves on the first literal in the clause to produce a resolvent which becomes the new centre clause. This continues until the empty clause is produced. We implement the OL-resolution procedure with a depth-first search strategy. If at any time no resolving clause can be found, the procedure backtracks. This procedure is made as efficient as possible by indexing clauses by the literals they contain, so that no search is needed to find side clauses for resolving with the centre clause. OL-refutation employs the standard techniques of factoring (deleting subsumed literals) and framing literals to create reducible clauses (avoiding the need for ever using previous centre clauses as side clauses later in the proof), and has been shown to be a sound and complete inference method [Chang & Lee 1973].

The following algorithm describes the main features of the OL-resolution procedure:

```
Resolve(Centre_Clause, Database, Min_Rank_Used, Cut_Off_Rank)
    If there exists a factor or reduction CC' of Centre_Clause
        Resolve(CC', Database, Min_Rank_Used, Cut_Off_Rank)
    If there exists no unframed literal in Centre_Clause
        Return Min_Rank_Used
    Otherwise take the first unframed literal $L \in$ Centre_Clause
    For each clause $C \in$ Database
        If rank(C) > Cut_Off_Rank and
          C resolves with Centre_Clause on L, giving CC' and
          C.count < Max_Count
        Then
            Frame(L) in CC'
            If rank(C) < Min_Rank_Used
                Resolve(CC', Database, rank(C), Cut_Off_Rank)
            Else
                Resolve(CC', Database, Min_Rank_Used, Cut_Off_Rank)
```

5.2

The theorem prover finds proofs of α and associates with each proof a rank, defined as the rank of the lowest ranked formula involved in the proof. This is a lower bound for the rank of α. The negation of α, used as the initial centre clause, is assigned rank n, the rank of the logical theorems, so that it does not affect the rank of the proof. Subsequent attempts to prove α are not permitted to use any formulae of equal or lower rank than the previous proof of α, since such a proof cannot provide any further information about the rank of α.

Suppose **prove_1**(Γ, α, r) denotes the rank of the first proof of α found by **prove**(Γ, α, r). Then, if **prove_1**(Γ, α, r) returns r_1, the next value generated by prove is **prove_1**(Γ, α, r_1), and the rank of this proof is assigned to r_2, which is used in the next proof, **prove_1**(Γ, α, r_2). This process continues until the call to **prove_1**(Γ, α, r_i) fails for some i, indicating that all such proofs of α have been found, and the greatest lower bound of the ranks, r_i, is returned as the rank of α. If no proof is found, the rank of α is zero, indicating that α is not a current belief.

Note that our OL-refutation procedure is based on depth-first search and thus is not guaranteed to produce the highest ranked proofs first. In fact, in the worst case, it is possible that it could generate all possible proofs of α. By limiting the proof to formulae in $\overline{\Gamma_r}$, and increasing r as successively higher ranked proofs are discovered, it is hoped that the average case behaviour would be significantly better than the worst case. To date we have no experimental results to confirm or deny this assumption.

An alternative to this depth-first approach would be to use an OL-refutation procedure based on breadth-first search, which is guaranteed to find the highest ranked proof first, but this would be far less space efficient than our method.

5.3 Equality

In order to implement examples of nonmonotonic reasoning (see chapter 7), it is necessary to add an equality predicate to the logical language recognised by the belief revision system. Performing resolution alone on a belief set containing equality relations does not necessarily yield correct results, unless the meaning of the equality symbol is encoded into the system in some way. One way of doing this is by adding extra axioms to the logical database to specify the properties of equality: reflexivity, symmetry, transitivity and substitutivity. The disadvantage of this approach is that many more clauses are needed to represent a belief state, and also many more useless resolvents are generated by the resolution procedure. A better approach is paramodulation, which replaces the symmetric, transitive and substitutive properties of equality with an inference rule which can be used with resolution.

Paramodulation is given the following definition in [Chang & Lee 1973]: Let C_1 and C_2 be two clauses with no variables in common. If C_1 is $L[t] \vee C'_1$ and C_2 is $r{=}s \vee C'_2$, where $L[t]$ is a literal containing the term t and C'_1 and C'_2 are clauses, and if t and r have most general unifier σ, then infer:

$$L\sigma[s\sigma] \cup C'_1\sigma \cup C'_2\sigma,$$

where $L\sigma[s\sigma]$ denotes the result obtained by replacing one single occurrence of $t\sigma$ in $L\sigma$ by $s\sigma$. The above inferred clause is called a binary paramodulant of C_1 and C_2.

We then extended the OL-deduction procedure, in order to allow paramodulation steps in the inference procedure. Note that each paramodulation step can be treated as a sequence of the following two OL-resolution steps. Assuming the notation in the above definition, with C_1 as the centre clause containing the resolving literal $L[r]$, resolve against the theorem $r{\neq}s \vee \neg L[r] \vee L[s]$, to give the new centre clause C_p, in which $L[r]$ is framed. Then C_p is resolved with C_2, on the literal $r{=}s$, to give the same binary paramodulant as shown in the definition above, with literals in the following order, and including the following two framed literals:

- the literals in $C'_2\sigma$, in the same order they appear in C_2
- $r\sigma = s\sigma$ (framed)
- $L\sigma[s\sigma]$
- $L[r]$ (framed)
- the literals in $C'_1\sigma$, in the same order they appear in C_1

It was stated earlier that paramodulation replaces the symmetric, transitive and substitutive properties of equality, but it was not mentioned that it does not account for the reflexivity of equality. To obtain completeness, [Chang & Lee 1973] states that it is sufficient to add to the set of clauses the axiom $x{=}x$, where x is a variable, plus the set of functionally reflexive axioms $\{f(x_1, \cdots, x_n)\}$, for all n-place function symbols f occurring in the set of clauses. But paramodulation was chosen to avoid the need for cumbersome axiomatisations of equality, so this approach is unsatisfactory.

It is often possible to evaluate equality relations directly, using unification and specific assumptions about the naming of objects in the domain. We use the following procedure to implement the reflexivity of equality. If a centre clause C containing an inequality literal is $r \neq s \vee C'$, for some clause C', and r and s are unifiable with most general unifier σ, then C is unsatisfiable if $C'\sigma$ is unsatisfiable, so we can use $C'\sigma$ as a new centre clause for resolution.

It is more difficult to deal with positive equality literals. First we note that there are several ways of naming objects in a domain, which lead to different interpretations of equality. The unique names assumption is discussed in chapter 7, where the equality predicate is used in nonmonotonic reasoning. If the unique names assumption is invoked (via the command-line option "-u") then $a = b$ is unsatisfiable for any distinct constants a and b, otherwise $a = b$ is unsatisfiable only if $a \neq b$ is explicitly derivable from the belief set via resolution.

Likewise, the choice of whether or not to use a domain closure assumption affects the evaluation of quantified equality predicates. The system is built with the assumption that the domain of objects is not closed, as there is no reason to believe that when we revise our belief set with new information, this new information would not mention any new objects. The result of this assumption is that formulae used to express domain closure, such as $\forall x((x=a_1) \vee \cdots \vee (x=a_k))$ are inconsistent in the AGM system. Conversely, formulae such as $\exists x(x \neq a)$ are always satisfiable.

We now complete the description of the equality evaluation procedure. If the centre clause C contains a positive equality literal $r=s$, then unless the terms r and s are identical (so that C is satisfiable), we defer the evaluation of the literal until all further resolution steps have been completed. In this way, we obtain as many instantiations of variables as possible. Then, if the unique names assumption has not been invoked, the remaining clause is assumed to be satisfiable and the refutation fails. Otherwise, an occurs check is performed, and a unification test, to determine whether or not each literal is unsatisfiable. If the clause cannot be shown to be unsatisfiable, the proof fails.

5.3

5.4 Interface to the System

We have yet to design a user-friendly graphical user interface for the AGM belief revision system. The current system provides a text-based interface. It has the following command line options:

-a	For simulation of ATMS (see chapter 6)
-b	Use Theory Base (not AGM) operations (see chapter 4)
-c	Print a count of the number of calls to resolve()
-f	Use the fast_revise algorithm (see chapter 4)
-n	For nonmonotonic reasoning (see chapter 7); implies -u
-p	Print formulae in both internal representations and as input
-r	Print a trace of resolution and paramodulation steps
-s	Silent flag; do not print warning messages
-t	Abort program on successful proof (for stack trace)
-u	Invoke unique names assumption

Input to the program consists of commands to perform AGM operations (expansion, contraction or revision), queries of the epistemic state (belief set membership, relative entrenchment of beliefs, the contents of the entrenchment base, or queries of conditional statements), and a command to set the maximum depth of the theorem proving procedure. Commands which have well-formed formulae as arguments use the following symbols, not necessarily in CNF:

~	negation
|	disjunction
&	conjunction
->	implication
<-	reverse implication
<->	double implication
=	equality (infix)
Exists(*X*)[*f*]	variable *X* is existentially quantified in formula *f*
All(*X*)[*f*]	variable *X* is universally quantified in formula *f*

Variables begin with upper case letters; predicate and function symbols begin with lower case. Parentheses may be used to disambiguate formulae; otherwise the usual precedences apply. Free variables are implicitly universally quantified in database formulae and existentially quantified in queries, as is usual with resolution theorem provers. A comma-separated list of variables may be used instead of a single variable in a quantified formula.

The following syntax is used for commands:

n: *f*	expansion by formula *f* with rank *n*
!*f*	contraction by formula *f*
*n***f*	revision by formula *f* with rank *n*
?*f*	query formula *f*
?*f* => *g*	query conditional $f \Rightarrow g$
?*f* ? *g*	query relative entrenchment of *f* and *g*
??	print current entrenchment base
$*n*	set maximum number of times a clause can be used in a proof to *n* (set *n* = 0 for no limit)

The conditional query "?*f* => *g*" is true with respect to a belief set K if $g \in K^*_f$; K is left unchanged.

5.5 Examples: Expansion, Contraction and Revision

This section illustrates the operation of the expansion, contraction and revision algorithms. We present the input to the system, followed by the system's responses indented with '>>>'.

```
100: All(X) [p(X) & q(X) -> r(X)]
10: r(a)
20: p(a)
30: q(a)
>>> Deleting r(a) from base : derived rank = 20
```

The first three expansions add the new belief at the given rank, with no other changes to the entrenchment base. The expansion by q(a) results in the deletion of r(a) from the database, as it is redundant, since it is derivable from p(a) with rank 20, q(a) with rank 30 and the rule with rank 100. Hence the rank of r(a) is now 20.

```
40: p(a) | r(a)
??
>>> Complete database:
>>> 100 : All(X) [p(X) & q(X) -> r(X)]
>>>  40 : p(a) | r(a)
>>>  30 : q(a)
>>>  20 : p(a)
? r(a)
>>> yes : rank = 30
? r(a) ? r(a) | p(a)
>>> r(a) < r(a) | p(a)
```

We then perform an expansion by p(a) $\vee$ r(a), and then print the whole database, and query the rank of r(a). Note that the addition of p(a) $\vee$ r(a) at rank 40 means that r(a) now has rank 30 because it is derivable from q(a), p(a) $\vee$ r(a) and the rule. The entrenchment relation between r(a) and r(a) $\vee$ p(a) is then queried, giving r(a) $<_E$ r(a) $\vee$ p(a).

```
! r(a)
??
>>> Complete database:
>>> 100 : All(X) [p(X) & q(X) -> r(X)]
>>>  40 : p(a) | r(a)
>>>  30 : (r(a)) -> (q(a))
>>>  20 : p(a)
```

Now the database is contracted by r(a). Because r(a) < r(a) $\vee$ p(a), p(a) remains in the contracted belief set by the (C–) condition, but the belief q(a) with rank 30 (so that r(a) has the same rank as r(a) $\vee$ q(a)) is replaced by r(a) $\rightarrow$ q(a), as required by AGM postulate (K-5). The beliefs which have a rank greater than r(a) (that is, greater than 30) are unaffected by the contraction operation, so they remain in the resulting entrenchment base.

```
40* ˜r(a)
>>> Deleting (r(a)) -> (q(a)) from base : derived rank = 40
>>> Deleting p(a) from base : derived rank = 40
??
>>> Complete database:
>>> 100 : All(X) [p(X) & q(X) -> r(X)]
>>>  40 : p(a) | r(a)
>>>  40 : ˜r(a)
```

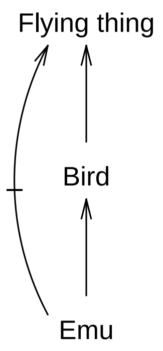
Finally the belief set is revised by ¬r(a) with rank 40. The formula ¬r(a) is added explicitly to the base with rank 40. The rank of r(a)→q(a) is now implicitly increased to 40, since it is derivable from ¬r(a), so this formula is deleted. Similarly, p(a) is also deleted because it is derivable with rank 40 from ¬r(a) and p(a) ∨ r(a).

5.6 Further Examples: Conditional Queries and Inheritance Networks

One standard problem in nonmonotonic reasoning is that of reasoning about inheritance networks [Brewka 1991b], which we use to illustrate one application of conditional queries. In an inheritance network, objects are described by their place in a hierarchy of types, where more specific types inherit information from more general types. The inheritance network must be able to deal with contradictions in the inherited information, and this is generally done by allowing information from more specific types to override the information inherited from a more general type. The classic example is:

Typically birds fly.
Emus are birds.
Typically emus don’t fly.

An inheritance network is often represented by a directed acyclic graph, as shown opposite. Directed links connect subtypes to their supertypes, so an arrow from X to Y represents the fact that generally X’s are Y’s. The crossed arrows represent negative links, so a crossed arrow from X to Y represents the fact that X’s are not generally Y’s. The above example would be represented as:



To encode this example in first-order logic, we write the inheritance rules as universal generalizations, and use the rank of a formula to encode the relative strengths of the rules. If more specific information is to override general information, the more specific rule should have a higher rank than the general rule.

```
10: All(X) [bird(X) -> flying_thing(X)]
20: All(X) [emu(X) -> bird(X)]
20: All(X) [emu(X) -> ˜flying_thing(X)]
? emu(X) => flying_thing(X)
>>> no
? emu(X) => ˜flying_thing(X)
>>> yes
```

We then query the database with: "Are emus flying things?", expressed as a subjunctive conditional query, "If *X* were an emu, would it be a flying thing?" The conditional is evaluated by the Ramsey Test [Ramsey 1931], which can be formulated as:

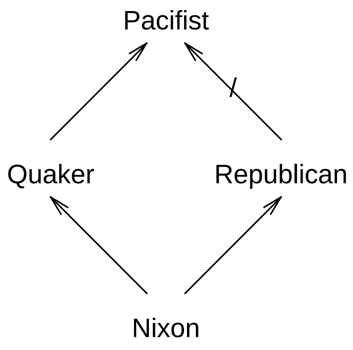
$$\alpha \Rightarrow \beta \text{ iff } \beta \in K^*_\alpha$$

Note that in the unrevised theory, it is inconsistent for any object to be an emu, since both flying_thing and ¬flying_thing would be derivable for that object. Thus we cannot expand the theory with an instance of an emu. If we were to revise the theory with an instance of an emu, then the defeasible rule, All(X) [bird(X) -> flying_thing(X)], would be removed as being the cause of the inconsistency. Obviously we do not want one exception to a rule to invalidate the rule entirely, and so we use a conditional query, which has the property that the base is not affected by the query, whereas the revision operation would remove rules which are inconsistent with current facts. An alternative solution to this problem is the subject of chapter 7.

Returning to the example, we see that the two queries, "Are emus flying things?", and "Are emus not flying things?", are answered correctly, and the database is not affected by the queries.

The next simple example, the Nixon diamond, provides an example where an object is an instance of two types with conflicting properties. The desired conclusion is that the system should be agnostic about both properties.

Republicans are not pacifists.
Quakers are pacifists.
Nixon is a Republican.
Nixon is a Quaker.



We then want to know whether or not Nixon is a pacifist. The example as stated would be encoded as follows:

```
10: All(X) [quaker(X) -> pacifist(X)]
10: All(X) [republican(X) -> ˜pacifist(X)]
? republican(nixon) & quaker(nixon) => pacifist(nixon)
>>> no
? republican(nixon) & quaker(nixon) => ˜pacifist(nixon)
>>> no
```

Note that in the absence of additional information to break the conflict, it is impossible to conclude that Nixon is or is not a pacifist. This time, if we performed a revision by republican(nixon) $\wedge$ quaker(nixon), both of the rules would be removed from the database, as they are inconsistent with the new information, and equally ranked. Now suppose we were given further information, that it is more likely for a Quaker to be a non-pacifist than a Republican to be a pacifist. Then we encode this information by giving the more reliable rule a higher rank than the other, and the example becomes:

5.6

```
10: All(X) [quaker(X) -> pacifist(X)]
20: All(X) [republican(X) -> ˜pacifist(X)]
? republican(nixon) & quaker(nixon) => pacifist(nixon)
>>> no
? republican(nixon) & quaker(nixon) => ˜pacifist(nixon)
>>> yes
```

The system correctly derives that Nixon is not a pacifist.

5.7 Summary

In the last three chapters, we have developed a computational model for AGM belief change operations, presented specific algorithms for implementing the model, and described our implementation of an AGM belief revision system over first-order logic with equality. The computational model is based on a partitioned finite base, which represents both the belief set and the epistemic entrenchment relation over the language, via the construction of the most conservative entrenchment. The representation of the entrenchment and the specific belief change algorithms provide an efficient means of implementing the AGM expansion, contraction and revision operations. The implementation is unique as a system which computes fully rational belief change. The algorithms also provide a unique solution to the iterated revision problem and the more general entrenchment revision problem, by ensuring the belief change operations invoke a minimal change in the epistemic entrenchment relation.

In chapters 6 and 7, we extend this work to utilise the relationships that belief revision has with truth maintenance, foundational reasoning, and nonmonotonic reasoning. We describe an algorithm which extends the system to simulate the behaviour of the most well-known dynamic reasoning system, the ATMS, and show that there is a close relationship between foundational reasoning and most conservative entrenchments. We also use some established relationships with nonmonotonic reasoning to implement a nonmonotonic reasoning system with defaults and exceptions.

6 Belief Revision and Truth Maintenance

There are a number of operational systems which are classed as *truth* (or *reason*) *maintenance systems* [Martins 1990], which are derived from the original Truth Maintenance System (TMS) of [Doyle 1979]. The features which are common to all such systems are that they perform some form of dynamic reasoning, in which a consistent set of beliefs is calculated from a changing set of facts or assumptions, and a (usually static) set of rules or justifications. Truth maintenance systems are the primary operational example of foundational reasoning, as they require that every belief either has a well-founded supporting justification, or else is considered to be self-justifying (foundational). These systems also exhibit temporal nonmonotonicity, that is, previously held beliefs may be given up as the set of supporting facts changes. This chapter focusses on the Assumption-Based Truth Maintenance System (ATMS) [de Kleer 1986], the most popular implementation of a truth maintenance system.

In this chapter, we develop a relationship between belief revision and truth maintenance. In particular, we present two algorithms for calculating and revising an epistemic entrenchment relation to simulate the behaviour of the ATMS using the AGM system. The AGM belief set is used to represent the ATMS context, and the ATMS context switches are performed by AGM expansion and contraction operations.

The first approach appeared in [Dixon & Foo 1992a, 1993], and does not rely on generating the most conservative entrenchment from a given partial entrenchment, but explicitly generates that part of the entrenchment relation which is needed to be tested using the (C–) condition for any admissable contraction operation. The admissable contraction operations are those which retract ATMS assumptions only, which corresponds with the operations allowed by the ATMS itself. This first approach is cumbersome, as the computationally expensive ATMS label update procedure is used in calculating the partial entrenchment.

The second approach shows that, by using the most conservative entrenchment, there is a very natural way to perform truth maintenance and foundational reasoning in general, within the AGM logic. This result is surprising for two reasons: firstly, the AGM logic is supposedly based on the principle of coherence rather than foundationalism; and secondly, it has been shown [Fuhrmann 1991] that foundational reasoning does not in general satisfy the AGM recovery postulate for contraction (K–5), $K \subseteq (K^-_\alpha)^+_\alpha$. The reasons for these surprising results are described in section 6.6.

The chapter is organised as follows: we first describe the ATMS, including a formal specification of its operation, which is used later, in the correctness proofs of the two algorithms. The next two sections describe the algorithms, and we prove that both of these algorithms provide correct implementations of the ATMS. In section 6.4, we describe the automation of the algorithms for use with the AGM implementation. It was necessary to make one change to the expansion algorithm in order to implement the ATMS simulation; the revised expansion algorithm is presented in this section. We then discuss extensions to the ATMS made possible by implementing it within the AGM system, and in the following section we describe the conditions under which the AGM system can be used to perform foundational reasoning. Finally, section 6.7 summarises the lessons learnt from this work.

6.1 The ATMS

The ATMS was developed for use in diagnosis and qualitative physics, where an efficient means of searching a large solution space was required, as well as the ability to make nonmonotonic inferences whilst keeping the database free of inconsistency. It consists of two parts: a problem solver and a truth maintenance system (TMS). The problem solver communicates the justifications for the inferences which it has made to the TMS, which, in turn, determines the current beliefs and passes this information back to the problem solver. Whereas the problem solver is often based on some subset of first-order logic and includes the domain knowledge and inference procedures, the TMS treats all beliefs as atomic, so that they have no logical structure and no logical relationships with other beliefs except as provided by the problem solver in the form of justifications. A justification is communicated by the problem solver as a set of supporting beliefs (the antecedents) and a supported belief (the consequent), interpreted by the TMS as a material implication from the conjunction of the antecedents to the consequent. Thus the justifications are interpreted as propositional Horn clauses, and the TMS has only to perform propositional inference. In this chapter, we shall often use the name ATMS to refer to the TMS part of the system, rather than the whole of the system.

The ATMS implements a foundational approach to modelling belief by insisting that each non-foundational belief be justified by a set of supporting beliefs. Any proposition that does not have such a justification is not accepted as a belief, unless it is one of the foundational beliefs, which require no justification at all. In the ATMS, the foundational beliefs are called *assumptions*, and there is a predetermined set of possible assumptions.

In the foundational theory, a justification is valid only when all of the beliefs which make up the justification are either foundational or are justified themselves, so that chains of justifications are formed. Foundationalism places two restrictions on these chains: firstly they must be acyclic, so that no proposition can form part of its own justification, and secondly the chains must be finite, so that ultimately all beliefs are supported by the foundational beliefs, which require no further justification. The ATMS removes the restriction that justifications be acyclic, since the "ATMS mechanism will never mistakenly use it as a basis for support" [de Kleer 1986] (p.155). The ATMS has no way of representing infinite chains of justifications.

In this work, we place one additional restriction on the ATMS: we do not allow the assumptions to be justified. Surprisingly, this restriction does not reduce the power of the ATMS at all, since [de Kleer 1986] provides a technique which "avoids ever having to justify assumptions" (p.147). Intuitively, too, since the assumptions are supposed to represent foundational beliefs, if they are given a justification, we cannot continue to consider them to be foundational.

6.1.1 ATMS Definitions

In the ATMS, each proposition is represented by a *node*, and is treated as atomic. Any logical relationship between nodes must be provided by the problem solver, in the form of justifications which are passed to the TMS. The justifications represent propositional Horn clauses, with the normal provability relationships, except that inconsistency is avoided by the use of *nogoods,* which have a separate data structure. Nogoods are sets of assumptions which cannot be held simultaneously, and they are used to reduce the size of the search space, improving the efficiency of the ATMS. A nogood can be seen as a justification for the propositional constant false, which is logically equivalent to the negation of the conjunction of the set of assumptions it contains.

The system operates with a set of assumptions called the *environment*, and the set of facts derivable from this environment is called the *context*. An atomic proposition p is believed in an environment E if and only if it is a member of the current context. In this thesis, ATMS derivability is denoted $E \vdash_{ATMS} p$, where we assume a fixed set of justifications and nogoods.

For a proposition to be believed in a particular context, it must be an assumption or else have a well-founded supporting justification, that is, a collection of justified justifications starting from sets of assumptions, where each of these assumptions is a member of the current environment. The only other condition under which a proposition is believed is when the current environment is inconsistent, in which case all propositions are believed.

The implementation of the ATMS achieves a high level of efficiency by creating a static data structure in place of a theorem prover to calculate revisions of belief sets. The justifications and nogoods are "pre-compiled" into this data structure initially, and after this they are not consulted again. The ATMS creates a node representing each proposition supplied by the problem solver, and also associates a *label* with it, which stores the justificational information. The label represents the set of environments in which the proposition is believed. That is, the label is a set of support sets, each of which are sets of assumptions. The label must be *consistent, sound, complete* and *minimal*. A label is *consistent* when each of its members is a consistent environment. That is, a label must not contain an environment which is a superset of any nogood set. The label for p is *sound* when p is derivable from each environment of the label. If every environment from which p can be derived is a superset of some environment of its label, then the label is *complete*. A label is *minimal* if no environment of the label is a superset of any other.

Some attempts have been made to formalise the logic of the ATMS, for example [Reiter & de Kleer 1987; Selman & Levesque 1990; Kean & Tsiknis 1993]. We do not attempt to formalise the logic of the ATMS, but we have formalised the functional behaviour of the ATMS in order to prove the correctness of the simulation algorithm.

6.1.2 An Example

Suppose we are given the following set of assumptions, justifications and nogood environments (written in their logical form, for clarity):

Assumptions: $w,\ x,\ y,\ z$

Justifications: $w \wedge x \rightarrow a$

$x \wedge z \rightarrow b$

$y \rightarrow b$

$a \wedge b \rightarrow c$

Nogoods: $\neg(x \wedge y)$

Then the ATMS labels are:

Labels: $a:\ \{\{w,\ x\}\}$

$b:\ \{\{x,\ z\},\ \{y\}\}$

$c:\ \{\{w,\ x,\ z\}\}$

$nogood: \quad \{\{x,\ y\}\}$

The ATMS contexts can then be calculated by simple inclusion tests: a node is in a context if and only if one of the sets in its label is a subset of the environment. The environment is inconsistent if the context contains a member of the label for *nogood*. For example, the node b is in the context of the consistent environments $\{x,\ z\}$, $\{y\}$, $\{w,\ y\}$, $\{y,\ z\}$, $\{w,\ x,\ z\}$ and $\{w,\ y,\ z\}$, but it is not in the contexts of $\{w,\ x\}$, $\{w,\ z\}$, $\{\}$, $\{w\}$, $\{x\}$ or $\{z\}$. The environments $\{x,\ y\}$, $\{w,\ x,\ y\}$, $\{x,\ y,\ z\}$ and $\{w,\ x,\ y,\ z\}$ are inconsistent.

6.1.3 ATMS Functional Specification

Let the set of all atoms be denoted Σ. Let E^* represent the set of all allowable assumptions. As described above, no member of E^* may appear as the consequent of any justification. Let x and y represent atoms, and A a set of atoms. The ATMS operates with a fixed set of justifications J, which are ordered pairs (A, y) with $A \subseteq \Sigma$, $y \in \Sigma - E^*$, representing the logical formula $(\bigwedge_{x \in A} x) \rightarrow y$. There is also a fixed set of nogood environments N, where for each $A \in N$, we have $A \subseteq E^*$, representing the logical formula $\neg \bigwedge_{x \in A} x$. The environment $E \subseteq E^*$ contains the assumptions which are currently believed.

Then for $p \in \Sigma$, we define $E \vdash_{ATMS} p$ if and only if:

1. $p \in E$, (p is an assumption), or
2. $\exists\, C \in N$ such that $C \subseteq E$, (E is inconsistent), or
3. $\exists\, (A, p) \in J$ such that $\forall\, x \in A$, $E \vdash_{ATMS} x$, (p has a well-founded justification).

Condition 3 may be reformulated in terms of the ATMS label:

3A. $\exists\, L \in Label(p)$ such that $L \subseteq E$.

6.1.4 ATMS Labels

Using the functional specification of the ATMS, it is possible to work backwards to a definition of the label of a node. Each environment in the label of a node p is a support set for p, so the label is a set of support sets, which may be defined as follows.

$A \in Label(p)$ if and only if:

1. $A \subseteq E^*$
2. $A \vdash_{ATMS} p$
3. $(\forall\, A' \subset A)\ \ [A' \nvdash_{ATMS} p]$
4. $(\neg \exists\, C \in N)\ \ [C \subseteq A]$

The first condition requires that the label contain assumptions only, that is, the foundational beliefs of the node. The second condition ensures soundness, the third minimality, and the fourth consistency. Completeness is achieved when the label contains all sets fulfilling the four conditions.

The above definition uses the ATMS provability relation to extract the internal data structure from the ATMS. However, an explicit definition is necessary to implement the algorithm in section 6.2. When the justifications are initially loaded into the ATMS, for each justification $(A, p) \in J$, the algorithm **Update_Label**(p) is called. The label update algorithm is performed as follows:

```
Update_Label($p$)
    For each justification $(B,p) \in J$
        If the label of any member of $B$ is empty
            Continue with next justification
        For all choices of one environment from each member of $B$
            Let $L$ be the union of these environments
            If $L$ subsumes any nogood environment
                Continue with next choice
            Else if $L$ is subsumed by any environment in $Label(p)$
                Continue with next choice
            Else if $L$ subsumes any environments in $Label(p)$
                Remove those environments from $Label(p)$
            Add $L$ to $Label(p)$
    If $Label(p)$ has changed
        For each justification $(B,q) \in J$ such that $p \in B$
            Update_Label($q$)
```

The labels of all assumptions are initialised to the singleton support set containing the singleton assumption itself. That is, if p is an assumption, $Label(p) = \{ \{p\} \}$. The labels of all other nodes are initially empty.

The foundational beliefs of an atom p in the environment E, denoted $FB(p, E)$, is the set of support sets of p which are believed in the environment E. If no such set exists, then the atom has no well-founded justification. Formally:

$$FB(p, E) = \{ A \in Label(p) : A \subseteq E \}$$

6.1

The essential support set, $ES(p,\ E)$, is defined:

$$ES(p,\ E)\ =\ \cap FB(p,\ E)$$

From the functional specification of the ATMS, a proposition is believed if and only if it is an assumption, or the environment is inconsistent, or it has a well-founded justification. We can now express this specification in terms of the above definitions.

For $p \in \Sigma$, $E \vdash_{ATMS} p$ if and only if:

(1) $p \in E$, or
(2) for some $C \in N$, $C \subseteq E$, or
(3) $FB(p,\ E) \neq \emptyset$

Using these definitions, we can now describe the algorithms which calculate the entrenchment relations for the AGM logic to simulate ATMS behaviour.

6.2 ATMS Algorithm 1: Explicit Entrenchment Generation

The algorithms in this and the next section reconstruct the dynamic reasoning of the ATMS using the AGM system. The entrenchment base Γ contains the logical form of the ATMS justifications and nogoods, as well as the environment and some disjunctions of assumptions which encode the independence of the assumptions. The belief set K derived from this base contains the ATMS context (and all of its logical consequences).

6.2.1 Constructing an Entrenchment Relation

In previous chapters, the entrenchment relation was generated from a finite entrenchment base by the most conservative entrenchment construction. In this section, we do not make use of the most conservative entrenchment, but instead we specify a partial ordering which contains sufficient information to define uniquely any admissable contraction function. In doing this, we are not defining an alternative to the AGM entrenchment postulates, but instead we are allowing many of the relative entrenchments to remain unknown, since we will never need to consult these values to implement any AGM contraction operation. That is, any extension of the partial entrenchment $\leq_E$ to an entrenchment relation $\leq'_E \supseteq \leq_E$ which is consistent with the AGM entrenchment postulates, will not change the behaviour of the algorithm. The most conservative entrenchment is one such consistent extension.

Looking from a different perspective, we can consider the partial entrenchment to define the class of all entrenchments which contain the partial entrenchment and are consistent with the AGM postulates. Then all members of this class behave identically with respect to the simulation algorithm.

As in previous chapters, the partial entrenchment relation will be specified in terms of the ranks of various formulae, which, for convenience, are represented by natural numbers. Since we are modelling ATMS environment changes, the only beliefs to be added or contracted are ATMS assumptions, which are atomic. We show in section 6.6 that this restriction is crucial to the success of both algorithms. In fact, our algorithm contracts one atom at a time and, for the comparison with the behaviour of the ATMS, the only beliefs we need to check for membership in K are also atomic (ATMS nodes). From the definition of contraction, we only need to know entrenchment relations between α and $\alpha\vee\beta$, for each $\alpha \in E^*$ and $\beta \in \Sigma$. For any atoms α and β, (EE2) constrains the relation between these atoms to be $\mathbf{rank}(\Gamma, \alpha) \leq \mathbf{rank}(\Gamma, \alpha \vee \beta)$, so there are only two possibilities: either $\mathbf{rank}(\Gamma, \alpha) < \mathbf{rank}(\Gamma, \alpha \vee \beta)$ or $\mathbf{rank}(\Gamma, \alpha) = \mathbf{rank}(\Gamma, \alpha \vee \beta)$.

For this algorithm, it is sufficient to use only 5 distinct values for the ranks of beliefs, say, 0, 10, 20, 30 and 40. The structure of the entrenchment relation is as follows. By (EE4), non-beliefs are given rank 0, and by (EE5), $\mathbf{rank}(\Gamma, \alpha) = 40$ only if α is a theorem. Since ATMS justifications and nogoods cannot be altered once they are provided by the problem solver, they are entrenched at the next highest rank, 30. Finally, the atoms in the current environment have rank 10, and the disjunctions of an assumption and another atom are given a default rank of 20.

To avoid ambiguity, the entrenchment base corresponding to the ATMS environment *E*, is denoted Γ_E, where necessary, and so the the rank of the proposition α in Γ_E is denoted $\mathbf{rank}(\Gamma_E, \alpha)$.

6.2.2 The Explicit Rank Generation Algorithm

The algorithm calculates a partial entrenchment relation which defines an AGM contraction operation to simulate the ATMS environment changes. The belief set *K*, represented by the entrenchment base Γ, contains all of the current beliefs, and corresponds to the ATMS context. Obviously, since *K* is logically closed, it is a much larger set than the corresponding ATMS context, but in the correctness proof, we show that for all atoms $\alpha \in \Sigma$, $E \vdash_{ATMS} \alpha$ if and only if $\alpha \in K$.

ATMS_Algorithm_1

```
Γ := ∅
For each $(A,p) \in J$                          { Enter justifications into Γ }
    $\Gamma :=$ expand$(\Gamma, (\bigwedge_{x \in A} x) \rightarrow p, 30)$
For each $C \in N$                              { Enter nogoods into Γ }
    $\Gamma :=$ expand$(\Gamma, \neg \bigwedge_{x \in C} x, 30)$
$E_{Old} := \emptyset$
For each $E_{New}$                              { Main Loop }
    $\forall x \in (E_{Old} - E_{New})$                   { Remove ex-assumptions }
        $\Gamma :=$ contract$(\Gamma, x)$                            { 1 }
        $\forall y \in (E_{Old} - \{x\})$                        { 2 }
            $\Gamma :=$ expand$(\Gamma, x \vee y, 0)$
        $\forall c$ such that $x \in \cup FB(c, E_{Old})$             { 3 }
            $\Gamma :=$ expand$(\Gamma, x \vee c, 0)$                    { 3a }
            $\forall y \in (ES(c, E_{New}) - ES(c, E_{Old}))$           { 3b }
                $\Gamma :=$ expand$(\Gamma, y \vee c, 10)$
            $\forall y \in (\cup FB(c, E_{Old}) - \cup FB(c, E_{New}))$     { 3c }
                $\Gamma :=$ expand$(\Gamma, y \vee c, 0)$
    $\forall x \in (E_{New} - E_{Old})$                   { Add new assumptions }
        $\Gamma :=$ expand$(\Gamma, x, 10)$                           { 4 }
        $\forall y \in (E_{New} - \{x\})$                        { 5 }
            $\Gamma :=$ expand$(\Gamma, x \vee y, 20)$   { Default rank }
        $\forall c$ such that $x \in \cup FB(c, E_{New})$             { 6 }
            $\Gamma :=$ expand$(\Gamma, x \vee c, 20)$   { Default rank }     { 6a }
            $\forall y \in (ES(c, E_{New}) - ES(c, E_{Old}))$           { 6b }
                $\Gamma :=$ expand$(\Gamma, y \vee c, 10)$
            $\forall y \in (ES(c, E_{Old}) - ES(c, E_{New}))$           { 6c }
                $\Gamma :=$ expand$(\Gamma, y \vee c, 20)$
    $E_{Old} := E_{New}$
```

The algorithm encodes the support relationships in the entrenchment relation. That is, for some α and β, if α is an essential support for β, then the removal of α must force the removal of β, and this, according to the (C–) condition and (EE2), implies $\mathbf{rank}(\Gamma, \alpha) = \mathbf{rank}(\Gamma, \alpha\vee\beta)$. Since $\mathbf{rank}(\Gamma, \alpha) = 10$, this is achieved by ensuring $\mathbf{rank}(\Gamma, \alpha\vee\beta) = 10$. Alternatively, if β is not dependent on α, then we have $\mathbf{rank}(\Gamma, \alpha) < \mathbf{rank}(\Gamma, \alpha\vee\beta)$, by ensuring $\mathbf{rank}(\Gamma, \alpha\vee\beta) = 20$.

For each environment change, the algorithm computes the corresponding changes in the entrenchment relation from the changes in the essential support sets. For each new assumption α which is an essential support of the proposition β, the rank of $\alpha \vee \beta$ is set to 10, so that $\mathbf{rank}(\Gamma, \alpha) = \mathbf{rank}(\Gamma, \alpha\vee\beta)$. Conversely, for any α which was a member of the essential support set of β but is no longer an essential support, we let $\mathbf{rank}(\Gamma, \alpha\vee\beta) = 20$, so that $\mathbf{rank}(\Gamma, \alpha) < \mathbf{rank}(\Gamma, \alpha\vee\beta)$.

The algorithm works as follows. Initially, the logical forms of the justifications and nogoods are placed in Γ at rank 30. Then for each environment, the algorithm has two parts: removing the assumptions which were in the previous environment but are not in the current environment (steps 1 to 3), and adding new assumptions which are in the current environment but were not in the previous environment (steps 4 to 6). The removals are performed before the additions, to avoid passing through any inconsistent intermediate states.

To remove an assumption, the algorithm uses the AGM contraction operation (step 1), and then updates the entrenchment relation to reflect the changes in essential supports (steps 2 and 3). First, all disjunctions of the removed assumption and another assumption are deleted from the database (step 2), by adjusting their rank to 0. Then the changes in the essential support sets are reflected in the entrenchment relation by changing the ranks of the disjunctions of each pair of propositions whose support relationship has changed (step 3). This involves deleting each disjunction of the removed assumption and the beliefs it was supporting (step 3a), and then for each new essential support y of c, the disjunction $y \vee c$ is given a rank of 10 (step 3b), and for each c which was supported by some y, but no longer is supported by that y, the formula $y \vee c$ is deleted from the database (step 3c).

The algorithm then adds each new assumption using the AGM expansion operation (step 4), followed by a series of expansions which add the disjunctions of the new assumption and each other atom in the new environment to the database at the default rank of 20 (step 5). The essential support sets are then updated (step 6), by a series of expansions for the propositions supported by the new assumption (step 6a), with the default that the support is not essential. Any exceptions to this default are corrected by the following steps (6b and 6c), which update the ranks representing the essential support relationships in a manner similar to steps 3b and 3c. Finally, the current environment becomes the previous environment for the next iteration of the algorithm.

Note that the default rank of 20 for a disjunction of two atoms induces the relation $\mathbf{rank}(\Gamma, \alpha) < \mathbf{rank}(\Gamma, \alpha \vee \beta)$, for atomic α and β, since $\mathbf{rank}(\Gamma, \alpha) = 10$ for all assumptions α in the current environment. This encodes the fact that α and β are independent, since if α is removed from K, β is unaffected. It is the job of the problem solver, not the TMS, to point out the logical relationships between the various atoms, so we may assume all nodes are independent of each other, unless we are explicitly told otherwise, via a justification or nogood.

Finally we comment on the correctness of the algorithm, relative to the functional specification of the ATMS given in section 6.1. We show in appendix E that **ATMS_Algorithm_1** maintains a consistent belief set which correctly simulates the behaviour of the ATMS at all times. Thus we have implemented a foundational reasoning system using the AGM system, despite the results of [Fuhrmann 1991], who notes that foundational reasoning in general does not satisfy the AGM recovery postulate. The ATMS, the most well-known implementation of foundational reasoning, is an exception, in that its operations do satisfy the recovery postulate, as we have shown by implementing it using the AGM system.

6.2.3 Example

We now repeat the example from subsection 6.1.2 to demonstrate the operation of **ATMS_Algorithm_1**. The output of the algorithm is a series of AGM expansion and contraction operations. Note that apart from the expansions by atomic formulae (at rank 10), all other expansions are entrenchment revision commands, which can be executed as database updates since we do not assume the most conservative entrenchment construction. The first step performed by the algorithm is to generate expansion operations to add each of the justifications and nogoods to the belief set:

```
30: w & x -> a
30: x & z -> b
30: y -> b
30: a & b -> c
30: ~(x & y)
```

Now suppose we want to change environment from $E_1 = \{\}$ to $E_2 = \{w,\ x\}$ and then to $E_3 = \{w,\ y\}$. The change to E_2 invokes the following expansion operations:

```
10: x
20: x | w
10: w
20: w | x
20: w | a
10: x | a
10: w | a
```

The first four expansions update the environment, and encode the independence of the two assumptions (steps 4 and 5). Then since a is now supported by the new environment, $w \vee a$ is given the default entrenchment of 20 (step 6a). The last two lines encode the fact that both w and x are now essential supports for a, and so the rank of $w \vee a$ is immediately reduced from its default value (step 6b). When these operations are performed on the AGM system, the belief set contains the context of E_2, and any atomic queries to the system return the same answers as the ATMS.

The context switch to environment E_3 is performed as follows. First, the assumptions which are no longer believed $(E_2 - E_3)$ are removed. This is done by performing the contraction K_x^- (step 1), then deleting the disjunctions $x \vee w$ and $x \vee a$ (step 2), and then revising the entrenchment base to reflect the new supports (step 3). In this intermediate state, a has no support, so the disjunctions $x \vee a$ and $w \vee a$, which encoded the support relationships for a, are deleted (step 3a). Note the notation we use for a database deletion is the same as an expansion with rank 0. In the original AGM system, this was not a legal expansion operation; in section 6.4 we explain the changes to the expansion algorithm to allow rank-reducing expansions. Note that the algorithm is not optimal, as it generates redundant operations (such as the repeated deletion of $x \vee a$), but this is allowed to keep the algorithm as simple as possible.

6.2

```
! x
0: x | w
0: x | a
0: x | a
0: w | a
```

Then the new assumption (*y*) is added to the base with rank 10 (step 4), and the disjunctions $y \vee w$ and $y \vee b$ are given the default rank of 20 (steps 5 and 6a). Finally the new essential support relationship (*y* supports *b*) is encoded into the entrenchment relation by reducing the rank of $y \vee b$ to 10 (step 6b).

```
10: y
20: y | w
20: y | b
10: y | b
```

At this point we note that the belief state representing any ATMS context is always finitely representable, and there is a unique most conservative entrenchment corresponding to the explicitly generated entrenchments. For the present example, the base for the most conservative entrenchment representing the context of environment E_3 is:

```
30 : (x & w) -> a
30 : (z & x) -> b
30 : y -> b
30 : (b & a) -> c
30 : ˜y | ˜x
20 : y | w
10 : w
10 : y
```

The entrenchment revision process for this algorithm is trivial, as the expansion and contraction operations never affect explicit formulae apart from those mentioned in the expansions and contractions themselves. Therefore, the expansion operations can be implemented as database updates, so that either the formula and its rank are added to the database if it was not in the database originally, or else the rank of the existing database record is adjusted to the new rank mentioned in the expansion operation.

6.3 ATMS Algorithm 2: Conservative Entrenchment Generation

A more natural way to simulate ATMS behaviour is via the most conservative entrenchment. In chapter 3, one justification given for the most conservative entrenchment was that the rank of a formula is derived directly from the proofs (justifications) of the formula. Thus if we take the beliefs in the entrenchment base to be the foundational beliefs, then the derived beliefs are those which have a well-founded support in the base. We now give an example of how the most conservative entrenchment can be used to implement one type of foundational reasoning, via **ATMS_Algorithm_2**.

6.3.1 The Conservative Rank Generation Algorithm

The basic structure of the algorithm is similar to **ATMS_Algorithm_1**. The algorithm maintains an AGM belief state representing the ATMS justifications, nogoods and current environment. Rather than explicitly encoding the support relationships, we encode only the independence of the foundational beliefs. Recall that the ATMS assumes foundational beliefs (assumptions) to be independent unless explicitly told otherwise. Then the most conservative entrenchment automatically generates the correct ranking of formulae to simulate ATMS behaviour.

Note that this algorithm does not need to generate the ATMS labels, but can compute the entrenchment base directly from the current environment, making it considerably simpler and more efficient than the previous algorithm. In fact the only entrenchment information which must be explicitly supplied by the algorithm is that which encodes the pairwise independence of the assumptions. That is, for each pair of assumptions in Γ, the algorithm ensures that the rank of their disjunction is strictly greater than the rank of the assumptions themselves, so that the removal of one assumption will never force the removal of any other assumption. This property is re-established after each of the expansion and contraction operations.

The structure of the entrenchment base generated by the algorithm is as follows: the logical theorems are given a rank of 40, the ATMS justifications and nogoods are given a rank of 30, the independence disjunctions have a rank of 20, assumptions have rank 10, and non-beliefs must be given the rank 0.

ATMS_Algorithm_2

```
    $\Gamma := \emptyset$
    For each $(A,p) \in J$                        { Enter justifications into $\Gamma$ }
        $\Gamma :=$ **expand**$(\Gamma,\ (\bigwedge_{x\in A} x) \rightarrow p,\ 30)$
    For each $C \in N$                            { Enter nogoods into $\Gamma$ }
        $\Gamma :=$ **expand**$(\Gamma,\ \neg \bigwedge_{x\in C} x,\ 30)$
    $E_{Old} := \emptyset$
    For each $E_{New}$                            { Main Loop }
        $\forall x \in (E_{Old} - E_{New})$               { Remove ex-assumptions }
            $\Gamma :=$ **contract**$(\Gamma, x)$
            $\forall y \in (E_{Old} - \{x\})$                 { Remove disjunctions }
                $\Gamma :=$ **expand**$(\Gamma,\ x \vee y,\ 0)$
        $\forall x \in (E_{New} - E_{Old})$               { Add new assumptions }
            $\Gamma :=$ **expand**$(\Gamma,\ x,\ 10)$
            $\forall y \in (E_{New} - \{x\})$                 { Encode independence }
                $\Gamma :=$ **expand**$(\Gamma,\ x \vee y,\ 20)$
        $E_{Old} := E_{New}$
```

The initialisation procedure is the same as that of **ATMS_Algorithm_1**; the justifications and nogoods are added to the base with rank 30. Then for each environment change, the assumptions α which were in the previous environment but are not in the new environment are removed from the base via the contraction operation **contract**(Γ, α). Then the pairwise disjunctions with each other assumption β are removed, since there is no longer any reason for believing the disjunction $\alpha\vee\beta$ apart from the truth of the remaining disjunct β. The correct rank of $\alpha\vee\beta$ is then the same as that of β, which has a rank of 10, unless β has also been removed from the base; this value is computed using the most conservative entrenchment after the disjunction is deleted from the entrenchment base.

Then the new assumptions α are added to the entrenchment base, via the expansion operation **expand**$(\Gamma, \alpha, 10)$. To encode the fact that the belief in α is independent of the belief in each other assumption β, the rank of $\alpha\vee\beta$ is increased to 20, again using an expansion operation. This operation could be achieved by a database update, since it is always a consistent operation ($\alpha\vee\beta$ is only added to the base when both α and β

are already members of the base), and it does not cause any redundancy in the base. This is because the foundational beliefs are not allowed to be justified (see section 6.1), and thus the only clauses containing the assumptions as positive literals are the unit clauses representing the assumptions themselves, at rank 10. Thus any derivation of a clause containing only positive literals which are all assumptions must have rank no greater than 10, so no disjunction which is added by the algorithm can ever be made redundant. Likewise, the assumptions themselves cannot be made redundant, since they can never be derived from any subset of the base not containing the assumption itself, unless the environment is inconsistent.

Comparing this algorithm with the first ATMS simulation algorithm, we note that the converse of this is also true – all disjunctions added by **ATMS_Algorithm_1** which are not added by **ATMS_Algorithm_2** are redundant. This is expected, since both algorithms exhibit the same behaviour, and therefore they should also give rise to the same most conservative entrenchment.

We show in appendix F that **ATMS_Algorithm_2** maintains a consistent belief set which correctly represents the ATMS context at all times, and represents the same most conservative entrenchment corresponding to the base generated by **ATMS_Algorithm_1**.

6.3.2 Example

We now repeat the example from subsection 6.1.2 again to demonstrate the operation of **ATMS_Algorithm_2**, and highlight its simplicity compared to the previous algorithm. Note that apart from the expansions by atomic formulae (at rank 10), all other expansions are entrenchment revision commands, which can be executed as database updates. The algorithm begins by generating expansion operations to add each of the justifications and nogoods to the belief set:

```
30: w & x -> a
30: x & z -> b
30: y -> b
30: a & b -> c
30: ˜(x & y)
```

Once more we want to change environment from $E_1 = \{\}$ to $E_2 = \{w, x\}$ and then to $E_3 = \{w, y\}$. The change to E_2 invokes the following expansion operations:

```
10: x
20: x | w
10: w
20: w | x
```

The expansions update the environment, and encode the independence of the two assumptions *x* and *w*. When these operations are performed on the AGM system, the belief set will contain the context of E_2, and any atomic queries to the system will return the same answers as the ATMS.

The context switch to environment E_3 is performed as follows. First, the assumptions which are no longer believed $(E_2 - E_3)$ are removed. This is done by performing the contraction K_x^-, and deleting the associated disjunction ($x \vee w$).

```
! x
0: x | w
```

Then the new assumption (*y*) is added to the base with rank 10, and the disjunction $y \vee w$ is given rank 20 to encode the independence of the two assumptions.

```
10: y
20: y | w
```

When the algorithm was implemented, the operations described above produced the following final entrenchment base:

```
30 : (x & w) -> a
30 : (z & x) -> b
30 : y -> b
30 : (b & a) -> c
30 : ~y | ~x
20 : y | w
10 : w
10 : y
```

6.4 Implementing the Algorithms

The algorithms described in the previous sections have been implemented in conjunction with the AGM system. **ATMS_Algorithm_1** was incorporated into the ATMS system developed in [Dixon 1989] to produce the ATMS labels and then derive the AGM expansion, contraction and database deletion operations to simulate the context switches. The output of the system was then piped straight into the input of the AGM implementation. This enabled easy testing of the algorithm and the underlying AGM system.

It was found that one minor change had to be made to the expansion algorithm of the AGM system, to obtain the correct entrenchment revision policy. In the original system, the rank of a formula could not be decreased by an expansion operation. For example, if a formula α is in Γ with rank 20, and we perform an expansion of Γ by α with rank 10, the database is returned unchanged. This was justified by considering the rank as the degree of evidence for a formula, so that when we are given evidence that is less compelling than the current evidence for a formula, our beliefs do not change.

For the ATMS implementation, we need to be able to reduce the rank of formulae in the database, without removing the formula entirely. In the general case, reducing the rank of a formula is problematic. Suppose a belief α is derived from formulae in the base with ranks 20 and 30, so that the rank of α is 20. Now if we require the expansion operation **expand**$(\Gamma, \alpha, 10)$ to reduce the rank of α from 20 to 10, there are several possible solutions to choose from, which alter the entrenchment relation successfully, without changing the belief set. We could decrease the rank of all formulae involved in any proof of α to 10, so that the rank of α is also decreased to 10. One might argue that this is not a minimal change, and so a better solution would be to reduce the rank of the lowest ranked formula in each proof of α. If there is more than one lowest ranked formula, we could either reduce all of them or, if that is objectionable because of the minimal change principle, we could choose one formula arbitrarily and reduce its rank. Having to make an arbitrary choice among formulae is exactly the problem which we initially tried to avoid, by placing a preference ordering on all formulae, so we do not consider this to be a viable solution.

There is a simpler and much better approach which we adopt for the ATMS simulation. Returning to the interpretation of the entrenchment base as a set of formulae accompanied by a measure of the evidence for each formula, we argue that a request to reduce the rank of a formula, without entirely removing the formula, can only be interpreted as discrediting a particular piece of evidence itself and not a derived belief. When there is evidence for the negation of a belief, the belief ought to be removed completely, but otherwise, the lowering of rank means that we do not trust the evidence to be as reliable as we previously thought. Thus the lowering of rank should be applied to the evidence directly, and not to the beliefs which are derived from the evidence.

Therefore, we only allow explicit formulae in the base to have their rank reduced, and then, the rank must not be reduced so much that the entrenchment base becomes inconsistent with condition (R1). That is, if an explicit formula is also derivable implicitly from other formulae in the base, its rank cannot be reduced to a level lower than the minimally ranked formula in the derivation.

```
expand_2($\Gamma$, $\alpha$, newrank)
    if rank($\Gamma$, $\neg\alpha$) > 0   { $\alpha$ is inconsistent with $\Gamma$ }
        return($\Gamma$)
    else
        oldrank := rank($\Gamma$, $\alpha$)
        if newrank $\leq$ oldrank and newrank $\leq$ rank(delete($\Gamma$, $\alpha$), $\alpha$)
            return(delete($\Gamma$, $\alpha$))
        else
            $\Delta$ := update($\Gamma$, $\alpha$, newrank)
            for each $\beta \in \overline{\Gamma_1}$ with oldrank $\leq$ select($\Gamma$, $\beta$, rank) $\leq$ newrank
                if prove($\Delta - \{\beta\}$, $\beta$, select($\Gamma$, $\beta$, rank)) > 0
                    $\Delta$ := delete($\Delta$, $\beta$)
            return($\Delta$)
```

So a rank-reducing expansion may be achieved by a simple update operation, as long as the formula is not derivable from other formulae in the base with a higher rank than the given new rank of the formula. The modified expansion algorithm checks the validity of the update by deleting the formula from the database and then calculating its rank in the new database. A warning message is printed if the rank in the new database is still greater than the given rank, and hence the update is not successful. Note that if the formula is not already in the database with a greater rank than the requested new rank, then the new expansion operation performs identically with the previous algorithm.

A special case of this expansion algorithm is if α is given the rank 0. Then the algorithm is equivalent to the database operation **delete**(Γ, α), leaving the remaining entrenchment base intact. This explains the choice of notation in the examples in this chapter, where the deletions were written as expansions with rank 0.

6.5 Extensions to the ATMS

Clearly there is no computational advantage in implementing an ATMS via the AGM logic and **ATMS_Algorithm_1**. The original purpose of the ATMS was to avoid the computational cost of theorem proving, especially the repetitive work involved in computing derivations of the same formula in different environments. **ATMS_Algorithm_1** uses both the expensive label update procedure and a theorem prover, making it computationally much worse than the ATMS. The point of developing the algorithm was to show that some forms of foundational reasoning, such as that exhibited by the ATMS, can be performed within the AGM system.

The second algorithm provides much more insight into the use of the AGM system. The algorithm itself is very efficient; the only real loss is in the complexity of query answering, which is a simple subset test for the ATMS, but requires a satisfiability computation for the AGM system. This satisfiability test is still tractable, as we are dealing only with propositional Horn clause logic.

One advantage of the AGM approach is that it generalises the ATMS to allow more complex formulae involving, for example, disjunction, negation, variables and quantifiers; in short, it allows us to give a logical structure to nodes, and to use the added expressive power of first-order logic. At the same time, we can still perform the same style of reasoning as the ATMS, by explicitly encoding the independence of foundational beliefs.

Another advantage of this approach is that it enables a mixture of foundational and non-foundational styles of reasoning, which is not possible with the ATMS. The need for a reasoning system to be able to perform non-foundational reasoning is addressed in [Dixon & Foo 1992b]; one important class of examples is in reasoning about action, where events may occur whose effects persist beyond the duration of the event, and thus our belief in the effect should not be given up just because the event which caused it no longer occurs.

Thirdly, the AGM approach to truth maintenance allows us to put a structure on the justifications, so that rules may be revised during belief change operations. In the first-order case, this allows a form of nonmonotonic reasoning using defaults with exceptions, which shall be discussed at length in chapter 7.

Apart from being an "existence proof" for the implementation of foundational reasoning or truth maintenance using a belief revision system, a comparison of the algorithms yields another interesting result. If we consider the second algorithm using the most conservative entrenchment as an implementation of the first, we see that the most conservative entrenchment automatically generates the support relationships between the beliefs. That is, if α is an essential support for β, then we shall have $\mathbf{rank}(\Gamma, \alpha) = \mathbf{rank}(\Gamma, \alpha\vee\beta)$. Similarly, if β is supported but α is not an essential part of the support, then we will have $\mathbf{rank}(\Gamma, \alpha) < \mathbf{rank}(\Gamma, \alpha\vee\beta)$.

6.6 Foundational Reasoning

The large number of entrenchment revision operations required by **ATMS_Algorithm_1** in order to simulate the ATMS context switches suggests that the style of dynamic reasoning performed by the ATMS is not well suited to the AGM logic. On the surface we can explain this by the two fundamentally different approaches to reasoning that the systems have.

The ATMS is strictly foundational in nature; every belief must have a valid justification which can be traced back to the foundational beliefs (assumptions). On the other hand, the AGM logic does not impose any requirements on the members of its belief sets, except logical consistency, as it is based on a weak coherence principle.

Foundational reasoning is independent of history. Hence, the ATMS context depends only on the current environment, and is not affected by any previous environment, whereas the AGM logic relies on a principle of minimal change when moving from one theory to the next. In order to achieve foundational behaviour using the AGM system, it is not sufficient to apply a principle of minimal change to the entrenchment relation, as non-minimal revisions of the entrenchment relation may be necessary to keep the system "effectively independent" of its previous state. Although this seems to be a complex process for the first algorithm, based on calculating support sets for each environment, we have shown with the second algorithm that most of this work is unnecessary if we use a most conservative entrenchment.

Thus we have reconstructed one style of foundational reasoning, using the AGM belief change operations with the assumptions of a most conservative entrenchment and the independence of explicit beliefs. The possibility of such a construction was first postulated in [Nebel 1989], where reason (truth) maintenance, the primary example of foundational reasoning, was described as a "side-effect" of choosing the "right" contraction operation, which in our model is equivalent to choosing the "right" entrenchment relation.

ATMS_Algorithm_2 can be generalised to implement a less restricted style of foundational reasoning than the ATMS allows. The basic function of the algorithm would be unchanged: to maintain a database of foundational beliefs, where the disjunction of each pair of foundational beliefs is entrenched higher than the foundational beliefs themselves. This encodes the independence of the foundational beliefs, which, by definition, are the beliefs which are held independently of any justification. The idea of explicitly encoding the disjunctions of explicit beliefs into the entrenchment relation appeared in [Williams 1992] under the name of a *maximal ensconcement*, but the application to foundational reasoning was not discussed in that work.

The one requirement which ensures that such a system behaves correctly is that the expansion and contraction operations only use formulae which are atomic foundational beliefs. It is because the ATMS satisfies this restriction that it also satisfies the AGM recovery postulate, and therefore is not an example of the general incompatibility of foundational and AGM reasoning suggested by [Fuhrmann 1991]. As soon as a contraction of a more complex formula or a non-foundational (derived) belief is performed, the foundational behaviour breaks down. For example, if the belief state generated by the ATMS simulation algorithms contains the two assumptions α and β, and we perform a contraction by $\alpha \vee \beta$, it is easy to show that under the most conservative entrenchment, all assumptions and all explicit disjunctions of assumptions would be removed from the base, leaving only the justifications and rules untouched. Alternatively, suppose a situation in which the foundational belief α is ranked lower than the justification $\alpha \rightarrow \beta$. Then a contraction by the derived belief β would force the removal of α, and in order that a subsequent expansion by β would restore α to the belief set (to satisfy the recovery postulate), the AGM contraction operation would add the formula $\beta \rightarrow \alpha$ to the belief set, contrary to foundationalist intuitions.

Thus we conclude that the AGM belief change postulates do not necessarily induce coherentist or foundationalist behaviour, as both types of behaviour can be achieved by the choice of a suitable entrenchment relation. We also note that the information in the entrenchment relation can be interpreted as encoding the dependence or independence of formulae in the belief set. This interpretation leads us to the conclusion that dependence and independence are the primitive notions from which the entrenchment relation may be derived.

6.7 Summary

We have shown that the ATMS can be simulated in the AGM logic by using a suitable epistemic entrenchment relation to encode the independence of the assumptions, and using contraction and expansion operations to perform ATMS context switches. The first ATMS algorithm does not compute a complete entrenchment relation, but defines the class of entrenchments for which the behaviour of the AGM system, with respect to atoms, is equivalent to the ATMS. The second algorithm maintains an entrenchment base which represents a unique entrenchment relation via the most conservative entrenchment construction.

These implementations of the ATMS illustrate one advantage of the AGM system over the purely foundational systems: it is possible to express different types of justificational information, using epistemic entrenchment to implement foundational and coherence style reasoning. It also demonstrates that the AGM system is not necessarily derived from coherentist principles, as foundational behaviour can be achieved by the use of a most conservative entrenchment with an explicit encoding of independence assumptions. Further, we conclude that since the most conservative entrenchment is suitable for implementing foundational reasoning, conservatism is also a foundational principle.

An interesting extension of this work is to allow rule revision, which is not allowed by the ATMS, but can be implemented in the AGM logic by decreasing the entrenchment of defeasible rules relative to the fixed rules. An application of rule revision is in modelling default rules with exceptions, a central issue in the field of nonmonotonic reasoning. In the following chapter, we discuss one way in which this may be achieved.

6.7

7 Belief Revision and Nonmonotonic Reasoning

Although belief revision can be regarded as providing a theoretical foundation for nonmonotonic reasoning, there have been few investigations of the practical use of belief revision for implementing nonmonotonic reasoning systems. In this chapter, we describe the design and implementation of a nonmonotonic reasoning system based on the AGM belief revision system described in previous chapters. The system uses the language of first-order logic with equality, and operates under the assumption of uniqueness of names. The entrenchment base is interpreted as a modified conservative entrenchment, motivated by the intuitions underlying default reasoning such as the independence of default instances.

The chapter is organized as follows. Firstly, we give a brief background to the area of nonmonotonic reasoning, showing its relationship to belief revision. Secondly, we summarize our approach to nonmonotonic reasoning based on the AGM belief revision system. We then discuss the necessary modifications to the belief revision algorithms described in chapter 4, in order to implement the nonmonotonic reasoning system. Following this, we present some examples of the system's behaviour on standard problems in nonmonotonic reasoning. Finally, we compare our work with other approaches to nonmonotonic reasoning.

7.1 Theoretical Connections

Although the formal structure of belief revision and nonmonotonic reasoning have many similarities, the motivations behind the two fields are quite different. Belief revision is concerned with how a state of belief is updated as a result of receiving new information, while nonmonotonic reasoning deals with how we may jump to conclusions that do not follow from our belief set by classical inference. Nonmonotonic reasoning often uses default rules or generalisations to define which nonclassical inferences may be performed.

This style of reasoning is nonmonotonic in the sense that increasing the set of data from which we reason may lead to the loss of some conclusions which were drawn from the smaller data set. That is, if $\mid\!\sim$ represents a nonmonotonic inference relation, then we may have, for some α, β, and γ, that $\alpha \mid\!\sim \gamma$ but $\alpha \wedge \beta \not\mid\!\sim \gamma$.

[Gärdenfors 1990a] shows that belief revision is also nonmonotonic, in the following three senses. Firstly, it is possible to have two belief sets A and B such that for some formula α we have $A \subseteq B$ but $A^*_\alpha \not\subseteq B^*_\alpha$. Secondly, for a belief set K and formula α, it is generally the case that $K \not\subseteq K^*_\alpha$. Finally, for a belief set K and formulae α and β, it may be the case that $\alpha \vdash \beta$ and yet $K^*_\alpha \not\vdash K^*_\beta$.

The formal connection between nonmonotonic reasoning and belief revision described in [Makinson & Gärdenfors 1991] is based on the following translations. For a fixed theory K equipped with a revision operation $*$, we may define a nonmonotonic inference relation $\mid\!\sim$ by $\alpha \mid\!\sim \beta$ if and only if $\beta \in K^*_\alpha$, for any formulae α and β. Conversely a belief revision function on some fixed background theory K may be defined by $\beta \in K^*_\alpha$ if and only if $\alpha \mid\!\sim \beta$.

Using this translation, Makinson and Gärdenfors compare the AGM belief revision postulates with conditions on various nonmonotonic inference relations, by translating postulates from one formalism to the other. [Gärdenfors 1990a] comments that all of the AGM postulates translate into conditions on $\mid\!\sim$ which are valid for some nonmonotonic inference relations in the literature and, conversely, that every postulate on $\mid\!\sim$ in the literature translates into a condition that is a consequence of the AGM revision postulates (K*1) – (K*8). Thus there is a very strong correspondence between the two areas.

7.2 A Belief Revision Approach to Nonmonotonic Reasoning

We now describe our nonmonotonic reasoning system, AGM-NMR, which uses a language of defaults expressed as restricted first-order sentences with equality, and exceptions expressed as propositional formulae. Each default represents the collection of all ground instances of the formula, entrenched such that the belief in each instance of the default is independent of the belief in all other instances of the default. That is, the discovery of one instance of an exception to a default does not affect our belief in any other instance of the default.

7.2

We define a default theory Δ to be a finite, nonredundant, consistent collection of nontrivial defaults of the form $\forall x\, \delta(x)$ where $\delta(x)$ is a clause, each of whose literals is either ground or of the form $[\neg]p(x)$ for some predicate symbol p and variable x or of the form $x = t$ for some ground term t. Note that the variable x must be the same in all literals in the clause, so that we may have an efficient algorithm for revising defaults.

Thus the language of defaults is highly restrictive, yet adequate for representing many problems in nonmonotonic reasoning. For example, we allow simple defaults such as $\forall x\, (bird(x) \to feathers(x))$, as well as more complex defaults with exceptions such as $\forall x\, ((x \neq tweety) \wedge bird(x) \to fly(x))$.

If the initial collection of beliefs is to be consistent, there can be no "universal" defaults such as $\forall x\, (bird(x) \to fly(x))$ if the belief set also contains ground facts which contradict the universal statement, such as $bird(tweety) \wedge \neg fly(tweety)$, an instance of a bird which does not fly. Under the belief revision approach to nonmonotonic reasoning, a generic default like "birds fly" is represented as the formula $\forall x\, (bird(x) \to fly(x))$, which represents a collection of independent beliefs $bird(t) \to fly(t)$, one for each term t in the language. For any exceptional bird, such as the non-flying *tweety*, the universal default must be weakened to $\forall x\, ((x \neq tweety) \wedge bird(x) \to fly(x))$ to ensure the consistency of the entrenchment base.

In addition, we assume that the default theory comes with a total pre-order $\leq$ on the defaults satisfying (EE1)–(EE3), which extends in a natural way to a canonical entrenchment based on the most conservative entrenchment compatible with $\leq$ modified by the assumption of independence of default instances. Given such a set of beliefs, the set of conclusions derivable nonmonotonically from a formula α are those formulae which occur in the revised belief set K^*_α, in agreement with the translation of [Makinson & Gärdenfors 1991].

From the point of view of nonmonotonic reasoning, most conservative entrenchments give intuitively acceptable conclusions only with propositional default theories, in other words, if a default theory is a consistent collection of ground instances of defaults. The AGM-NMR system uses first-order formulae to represent defaults, so the most conservative entrenchment generated from the ordering on defaults must be modified. The unacceptable behaviour resulting from the use of most conservative entrenchments can be illustrated by the Nixon diamond, formalized as the default theory consisting of two equally ranked formulae $\forall x\,(quaker(x) \rightarrow pacifist(x))$ and $\forall x\,(republican(x) \rightarrow \neg pacifist(x))$. To find out whether or not Nixon is a pacifist, we revise this theory by the formula $\alpha \equiv republican(nixon) \wedge quaker(nixon)$. According to the most conservative entrenchment, the resulting entrenchment base contains $republican(nixon) \wedge quaker(nixon)$ as expected, but nothing else! Each rule β is removed because in the most conservative entrenchment, $\neg\alpha$ and $\alpha \rightarrow \beta$ are equally entrenched.

We make use of two additional assumptions to overcome this problem: uniqueness of names and independence of default instances. First, uniqueness of names states that any two distinct terms in the logical language denote distinct objects in the domains. This essentially means restricting our attention to Herbrand models of the belief language. Uniqueness of names has been suggested by [Lifschitz 1989] as being an intuitively desirable property of default reasoning. Given that the instances of a default rule all apply to different objects in the domain by uniqueness of names, the second assumption states the truth of all these beliefs are independent. That is, there is no epistemic relationship of justification between two instances of the same default, so that when one instance of a default is removed from a theory by a revision, all other instances of the default remain in the theory.

Uniqueness of names can be formalized by adopting the standard axioms of equality and inequality into our logical language, as described in section 5.3. Independence of default instances can be formalized as follows. The idea is that the most conservative entrenchment correctly determines the entrenchment of the ground instances of the defaults, but must be generated from a modified initial ordering on defaults to incorporate this extra assumption. The modification is to assume that for each pair of defaults δ, $\delta' \in \Delta$, and for every ground term t, we have the entrenchment relation $\delta(t) <_E \delta(t) \vee \forall x\,((x \neq t) \rightarrow \delta'(x))$, where $\delta(t)$ denotes the instance of δ formed by replacing all occurrences of the variable in δ with the term t. By the (C$*$) condition for revision, this ensures that the weakened defaults $\forall x\,((x \neq t) \rightarrow \delta'(x))$, for all $\delta' \in \Delta$, remain in the theory after a revision by $\neg\delta(t)$ for any default $\delta \in \Delta$. Thus an exceptional instance $\neg\delta(t)$ of a default $\delta(x)$ does not invalidate any other instance of δ which held in the unrevised belief set.

7.2

We must also show that this modified conservative entrenchment is consistent with the entrenchment postulates (EE1) to (EE3), and also agrees with the original ordering on the entrenchments of the ground instances of the defaults. First we note that the weakened defaults are implicitly in the unrevised theory, since for all $\delta' \in \Delta$, $\forall x\,(\delta'(x)) \vdash \forall x\,((x \neq t) \rightarrow \delta'(x))$. Thus the modified conservative entrenchment does not introduce any new beliefs into the belief set, so the belief set is consistent whenever Δ is consistent. Also, the modified conservative entrenchment does not contradict the entrenchment of any default δ. We show this by considering all proofs of an instance $\delta(t)$ of the default δ.

Suppose there exists a refutation of $\neg\delta(t)$ using formulae ranked greater than δ. We divide these formulae into three sets, Δ_1, Δ_2 and Δ_3, where Δ_1 contains the initial defaults ranked higher than δ, Δ_2 contains the weakened defaults $\delta(t) \vee \forall x\,((x \neq t) \rightarrow \delta'(x))$ which contain $\delta(t)$, and Δ_3 contains any other weakened defaults $\overline{\delta}(t') \vee \forall x\,((x \neq t') \rightarrow \delta'(x))$, where $\overline{\delta} \neq \delta$. Let $\Delta' = \Delta_1 \cup \Delta_2 \cup \Delta_3$.

Now $\Delta \vdash \Delta'$, and $\Delta \vdash \delta'(t')$ for all $\delta' \in \Delta$ and any term t', so since Δ is consistent, we have $\Delta' \nvdash \neg\delta'(t')$, for all $\delta' \in \Delta$ and any term t'. Thus any refutation of $\neg\delta(t)$ cannot use clauses from Δ_3, as these can never resolve to the empty clause.

Now suppose a clause from Δ_2 is used in the refutation. Then after resolving with $\neg\delta(t)$, there are no other literals $\neg\delta'(t')$ to resolve against, so again, it cannot resolve to the empty clause. Thus no clause in Δ_2 can be used in the refutation.

Thus the refutation must use clauses solely from Δ_1, the initial defaults. But this contradicts the initial ordering on the defaults Δ, which, by (EE2) and (EE3), implied that $\Delta_1 \nvdash \delta(t)$. Hence we have shown that no such proof exists, and thus the modified conservative entrenchment agrees with the original ordering on the defaults.

7.3 A Revision Operation for Nonmonotonic Reasoning

We now outline the design of a system implementing these principles. We consider only two types of belief change: expansion and revision. The expansion operation is exactly the same as the one developed in chapter 4, but the revision operation has been modified to implement the independence of default instances.

An initial default theory is a collection of formulae over first-order logic with equality, and comes with an ordering respecting (EE1)–(EE3). The system needs to treat this ordering as standing for the most conservative entrenchment compatible with this ordering under the assumptions of uniqueness of names and independence of default instances, which is unique supposing the default theory to be consistent. It is assumed that the formula to be accepted in a revision is either a new, consistent, default or else a ground exception to a default, which by our assumptions is independent of all the other instances of that default.

We now present the algorithm **nmr_revise**(Γ, α, *newrank*) for revising a belief base Γ by a formula α with rank *newrank*. Assume *n* is the rank of the theorems.

```
nmr_revise($\Gamma$, $\alpha$, newrank)
    if rank($\Gamma$, $\alpha$) = $n$              { $\alpha$ is a theorem }
        return($\Gamma$)
    else if rank($\Gamma$, $\neg\alpha$) = $n$        { $\alpha$ is a contradiction }
        return($\Gamma$)
    else
        $\Delta$ := $\Gamma$
        oldrank := rank($\Gamma$, $\neg\alpha$)
        for $r$ := oldrank downto 1 do
            $\Xi$ := {}            { weakened defaults added at rank $r$ }
            for each $\beta \in \Gamma$ such that rank($\Delta$, $\beta$) $= r$ do
                let $\beta'(X) \equiv (X \neq t) \rightarrow \delta(X)$
                    where $\beta \equiv \delta(X)$ and $\alpha \equiv \neg\delta'(t)$
                if prove($\Delta - \{\beta\} \cup \{\beta'\}$, $\beta$, $r$) $=$ prove($\Delta$, $\beta'$, $r+1$) $= 0$
                { $\beta'$ is weaker than $\beta$ and $\beta'$ is not in $\Delta$ }
                    $\Xi$ := $\Xi \cup \{\beta'\}$
            for each $\beta \in \Gamma$ such that rank($\Delta$, $\beta$) $= r$ do
                if prove($\Delta$, $\alpha \rightarrow \beta$, oldrank$+1$) $= 0$    { (C*) condition }
                    $\Delta$ := delete($\Delta$, $\beta$)
            for each $\beta' \in \Xi$ do
                $\Delta$ := update($\Delta$, $\beta'$, $r$)
        $\Delta$ := expand($\Delta$, $\alpha$, newrank)
        return($\Delta$)
```

7.3

If α is a theorem, the revision is trivial and no change is made to the base. If α is inconsistent, the operation is disallowed and no change is made to the base. Otherwise we remove from the base any formula β such that $\neg\alpha$ is of rank equal to $\alpha \rightarrow \beta$ in the original base, as required by the (C*) condition. In addition, formulae β of the form $\delta(X)$ ranked at most the old rank of α are weakened to $(X \neq t) \rightarrow \delta(X)$ if the new formula really is weaker than β and it is not already contained in the base with a higher rank. It suffices to consider only those β ranked less than or equal to $\neg\alpha$ because if $\neg\alpha < \beta$ then $\neg\alpha < \alpha \rightarrow \beta$, so β remains in the revised set. Finally, the set of formulae remaining in the base is expanded by α.

7.4 Examples: Benchmark Problems in Nonmonotonic Reasoning

The following examples show the operation of the revision algorithm on some benchmark problems taken from [Lifschitz 1989]. The problems are divided into the categories of default reasoning, inheritance, uniqueness of names, reasoning about action and autoepistemic reasoning. We do not consider the problems involving uniqueness of names, since it is an assumption which is built into our system, nor those of autoepistemic reasoning, which have trivial solutions if we define an agent's knowledge to be the contents of its belief set. We now discuss how the system handles the three other types of problems.

7.4.1 Default Reasoning

The first example covers basic default reasoning, default reasoning with irrelevant information and default reasoning in an open domain (Lifschitz's examples A1, A2 and A5). The problem is stated informally as:

Assumptions:	Blocks *A* and *B* are heavy.
	Heavy blocks are normally located on the table.
	A is not on the table.
	B is red.
Conclusions:	*B* is on the table.
	All heavy blocks other than *A* are on the table.

For this example, specific facts are given a higher rank than defaults because specific information about objects is presumed to be more reliable than general defaults. An exception δ of a default is input to the system by making a revision to accept δ.

We now show the operation of the system on the three examples.

```
# Problem A1, Basic Default Reasoning
100: heavy(a)                          # Facts
100: heavy(b)

50: All(X) [heavy(X) -> table(X)]      # Default

100* ˜table(a)                         # Exception

? table(b)                             # Query
>>> yes : rank = 50


# Problem A2, irrelevant information
100: red(b)                            # Fact
? table(b)                             # Query
>>> yes : rank = 50


# Problem A5, open domain
? All(X)[ (X!=a & heavy(X)) -> table(X)]     # Query
>>> yes : rank = 50
??
>>> Complete database:
>>> 100 : heavy(a)
>>> 100 : heavy(b)
>>> 100 : ˜table(a)
>>> 100 : red(b)
>>>  50 : All(X) [(X!=a) -> (heavy(X) -> table(X))]
```

Note that when a revision is made to accept an exceptional instance of a default, the default is weakened to cover all but the exceptional case.

The following example involving several defaults (Lifschitz's problem A3) cannot be handled by the belief revision approach to nonmonotonic reasoning because the ordering on defaults which extends to an entrenchment is assumed to be a total pre–order.

7.4

Assumptions: Blocks *A* and *B* are heavy.
Heavy blocks are normally located on the table.
Heavy blocks are normally red.
A is not on the table.
B is not red.
Conclusions: *B* is on the table.
A is red.

The belief revision approach interprets any exception to a default at rank *r* as an exception to all defaults at rank less than or equal to *r*, unless the contrary is explicitly stated, for example by placing an instantiation of a default at a higher rank than the default itself. So if the default that heavy blocks are on the table is placed at a higher rank than the default that blocks are red, it is concluded that *B* is on the table, but not that *A* is red. If the ranks of the rules are swapped, the opposite results are obtained. The third possibility is that the defaults are given equal rank, intuitively the most reasonable choice, yet this yields neither of the desired conclusions, as *A* and *B* become exceptions to both rules.

The last example in this section is Lifschitz's problem A8 on reasoning about unknown exceptions.

Assumptions: Block *A* is heavy.
Heavy blocks are normally located on the table.
At least one heavy block is not on the table.
Conclusion: *A* is on the table.

This example is outside the scope of our theory, because the theory base contains $heavy(c) \wedge \neg table(c)$ where c is a Skolem constant. The question is whether c is different from *A*. Since the system does not assume that uniqueness of names applies to Skolem constants, it does not generate this conclusion. This can be justified on the grounds that there is no evidence that the unnamed block is not block *A*.

7.4.2 Inheritance

Reasoning about inheritance networks is a standard problem in nonmonotonic reasoning, and it is relatively straightforward to solve all of Lifschitz's problems B1–B4. We present our solutions to problem set B2, tree-structured inheritance.

Assumptions:	Animals normally don't fly.
	Birds are animals.
	Birds normally fly.
	Bats are animals.
	Bats normally fly.
	Emus are birds.
	Emus don't fly.
Conclusions:	Animals other than birds and bats do not fly.
	Birds other than emus fly.
	Bats fly.
	Emus don't fly.

In this example, we use the rank of a formula to encode the relative strengths of the defaults. The defaults which have no exceptions are given the highest rank. Then the default rules are ordered such that the more specific defaults (that is, exceptions to other defaults) override the more general defaults.

```
100: bird(X) -> animal(X)          # Facts
100: bat(X) -> animal(X)
100: emu(X) -> bird(X)

70: emu(X) -> ˜fly(X)              # Defaults
60: bird(X) -> fly(X)
60: bat(X) -> fly(X)
50: animal(X) -> ˜fly(X)

? All(X) [animal(X) & ˜bird(X) & ˜bat(X) -> ˜fly(X)]
>>> yes : rank = 50

? All(X) [bird(X) & ˜emu(X) -> fly(X)]
>>> yes : rank = 60

? All(X) [bat(X) -> fly(X)]
>>> yes : rank = 60

? All(X) [emu(X) -> ˜fly(X)]
>>> yes : rank = 70
```

7.4

For this example and for all of the inheritance problems proposed by Lifschitz, the answers generated by the system agree with the desired conclusions.

7.4.3 Reasoning about Action

The final type of problem considered in this section concerns reasoning about action. The approach to modelling these problems is based on the theory of actions developed in [Peppas *et al.* 1991], which uses belief revision functions from complete theories to complete theories to model events, by identifying an action with a revision to accept its postcondition. In a similar manner, the AGM-NMR system is used to keep track of the effects of multiple actions by performing successive revisions of the entrenchment base. Note that for these examples we do not assume that the world is modelled by complete theories.

The following example is Lifschitz's problem D3.

Assumptions:	After an action is performed, things normally remain as they were.
	A block is on the table if and only if it is not on the floor.
	When a robot grasps a block, the block will normally be in the hand.
	When the robot moves a block onto the table, the block will normally be on the table.
	Moving a block that is not in the hand is an exception to this rule.
	Initially block *A* is not in the hand.
	Initially block *A* is on the floor.
Conclusion:	After the robot grasps block *A*, waits, and then moves it to the table, the block will not be on the floor.

For this example, there are two kinds of facts: those which may change over time (for example, block *A* is on the floor), and those which are time invariant (for example, a block cannot be on the table *and* on the floor). In contrast to the examples concerning nonmonotonic reasoning in a static world, the defaults, which are time invariant, are given a higher rank than the specific facts, because a change in the world is presumed to affect the facts rather than override a default. The domain constraints, which are both time invariant and also not subject to exceptions, have the highest rank of all.

```
100: table(X) <-> ˜floor(X)                    # Domain Constraint

70: All(X)[˜holding(X) & move(X) -> ˜table(X)]     # Defaults
50: All(X)[grasp(X) -> holding(X)]
50: All(X)[move(X) -> table(X)]

10: ˜holding(a)                                 # Facts
10: floor(a)

10* grasp(a)                                    # Actions
10* move(a)

? ˜floor(a)                                     # Query
>>> yes : rank = 10
```

Note that the condition giving an exception to a temporal default "moving a block that is not in the hand is an exception to the previous rule" cannot be expressed in our language because it concerns an exception to a normal revision. This must be captured by a default stating explicitly that the outcome of the exceptional action is the negation of the expected outcome – we cannot capture chronological ignorance, only denial. Note also that the waiting action is presumed to have the postcondition *true* and so is modelled by a trivial revision.

Because the revision system can be used only in a "forwards" direction, it cannot be used directly for temporal explanation or reasoning about the unknown order of actions.

7.5 Discussion and Comparison with Related Work

A number of authors have demonstrated that there is a close relationship between belief revision and nonmonotonic reasoning. The connection between belief revision and nonmonotonic consequence operations was explored in [Gärdenfors 1990a] and [Makinson & Gärdenfors 1991], and extended in [Gärdenfors 1991], where the entrenchment relation was generalised to allow an ordering on non-beliefs, called an expectation ordering. These authors showed how belief revision relates to the default reasoning of [Poole 1988], which is itself a syntactic variant of the default logic of [Reiter 1980]. Poole’s skeptical inference was shown to be equivalent to full meet revision in the case where the default set is logically closed. A generalisation of Poole’s default reasoning corresponding to partial meet revision was also presented.

[Nebel 1991] also proposed the use of theory change operations for default reasoning, showing that prioritized base revision corresponds to both the skeptical inference of [Poole 1988] and the level default theories of [Brewka 1989], in the case of finitary propositional logic. [Brewka 1991a] developed a belief revision framework from an approach to default reasoning based on preferred subtheories generated from an ordering on defaults. This approach does not satisfy the basic AGM rationality postulates, as it does not even ensure the success of a belief change.

The connection of belief revision to conditional logic approaches to nonmonotonic reasoning was developed in [Wobcke 1992a,b], where a correspondence was shown between the extensions of a default theory as expressed in conditional logic and the class of most conservative entrenchments compatible with the ordering on defaults.

Our approach to nonmonotonic reasoning is closely related to those using orderings of defaults. Similar to epistemic entrenchment, priority orderings on formulae in a logical language have been proposed independently as a basis for implementing nonmonotonic reasoning systems. System Z, [Pearl 1990], uses a total ordering on a finite collection of defaults to define 0–entailment, the "core" of nonmonotonic inference, and 1–entailment, an extension of 0–entailment which has close connections to the rational nonmonotonic consequence operations of [Lehmann & Magidor 1992]. With a propositional database, the AGM-NMR system corresponds to the 1-entailment of [Pearl 1990] and the rational consequence of [Lehmann & Magidor 1992].

One of the weaknesses of this approach is that an exception to a default is propagated to all other defaults. For example, suppose we have a system with two defaults, which state that typically birds can fly, and typically birds have feathers. Then an exceptional bird, such as one that cannot fly, becomes exceptional in all other default properties of birds, such as having feathers. This was the difficulty encountered in section 7.3.1, where we attempted to solve Lifschitz’s problem A3, but were unable to specify the fact that the defaults were unrelated. To state that the two defaults are unrelated would require a partial order on defaults along the lines proposed by [Geffner & Pearl 1992] or [Brewka 1989, 1991a], but this cannot be expressed within the AGM framework, which requires a total pre-order for epistemic entrenchment. [Goldszmidt *et al.* 1990] also addresses some of the weaknesses of 1-entailment with an approach based on "maximum entropy" total orderings of defaults.

In [Pearl 1990] and its extensions, the ordering on the defaults is generated *automatically* from a given collection of defaults. This represents a major difference between approaches based on orderings and those based on belief revision because the entrenchment or expectation used by a belief revision system is assumed to be given. Clearly these approaches are less expressive than the AGM-NMR system, as they only allow the one arbitrary ordering.

7.6 Summary

We have developed a nonmonotonic reasoning system based on belief revision over the language of first-order logic with equality. The system accords with the AGM postulates for any finitely representable belief state, and interprets a partially specified entrenchment base as standing for a single most conservative entrenchment, modified to incorporate the assumptions of uniqueness of names and independence of default instances, which are motivated by the intuitions underlying nonmonotonic reasoning.

8 Conclusion

In this thesis, a rational and computational approach to belief revision has been presented. Firstly, we described the place of belief revision within the field of artificial intelligence, showing the need for belief change operations in order to reason about the real world. We then summarised the various approaches to belief revision, including both the constructive and nonconstructive modellings, and discussing their theoretical and practical limitations from the point of view of computational belief revision. The rationality constraints, defined by the AGM postulates, were supplemented in chapters 3 and 4 by a minimal change entrenchment revision policy, and the computational analysis was based on standard considerations of finiteness and tractability.

We then presented a new computational approach to AGM belief revision, beginning with a finite and efficient representation for a belief state, called an entrenchment base, which can be extended to an AGM epistemic entrenchment relation using the *most conservative entrenchment* construction. This construction was motivated both on the grounds of evidence, and by the close connection of belief revision with truth maintenance and nonmonotonic reasoning. Entrenchment revision policies were then discussed, and a conservative policy based on the coherentist minimal change principle was adopted.

The algorithms for computing the various belief change operations (expansion, contraction and revision) were then described. These algorithms were shown to be correct relative to the AGM postulates and a set of formal criteria defining minimal change in the epistemic state. The efficiency of the algorithms was discussed, concluding that the algorithms were all tractable if implemented over a logic with a tractable decision procedure for derivability. It was also shown that for any finitely representable belief state, the application of a consistent AGM operation would successfully end in another finitely representable belief state. Two alternative algorithms were also presented for comparison with the AGM algorithms.

We also described an implementation of these algorithms over first-order logic. This is the first such system which satisfies the AGM rationality postulates. The system uses a resolution theorem prover together with our algorithms for the efficient computation of expansion, contraction and revision operations. The theorem prover provides a method of ensuring termination, whilst sacrificing the completeness of the inference method, but not without alerting the user to the cases in which completeness may have been lost. The chapter concluded with a series of examples which demonstrate the operation of the system performing the standard belief change operations, as well as some nonstandard operations such as conditional queries.

Despite the fact that the AGM approach to belief revision is based on a coherentist view of justification, we showed that by using the most conservative entrenchment and encoding the independence of foundational beliefs, it is possible to perform foundational reasoning using the AGM operations. Having chosen the ATMS as a typical example of truth maintenance and foundational reasoning, we presented two simulation algorithms for achieving the behaviour of the ATMS within the AGM system. The first algorithm computes ranks of beliefs explicitly, so that no closure operation such as the most conservative entrenchment construction is needed. The second algorithm takes advantage of the most conservative entrenchment, and demonstrates that there is a very natural way of expressing foundational information in the AGM paradigm. Both algorithms were proved correct relative to the formal specification of the ATMS, and both were implemented in conjunction with the AGM system.

Finally, we described an approach to nonmonotonic reasoning based on belief revision, including an implementation which is built upon the AGM system. The system reasons with defaults and exceptions, using a modified version of the revision algorithm in chapter 4. The original algorithm was modified to incorporate the assumption of independence of default instances, and the theorem prover was extended to include equality in the logic, under the unique names assumption. The system was tested upon a number of standard problems in nonmonotonic reasoning, and the strengths and weaknesses of the system were discussed at length, and compared with other approaches to nonmonotonic reasoning.

8.1

8.1 Further Work

The implementation of the AGM belief revision system opens a door to the experimental evaluation of belief revision and nonmonotonic reasoning operations. There are three obvious directions for further work in this area: firstly, the AGM algorithms could be implemented with a different theorem prover such as for a restricted logic; secondly, the existing framework could be used to implement variations of the AGM belief change operations, such as the theory base operations described in chapter 2; and thirdly, we could investigate alternative entrenchment revision policies.

The first suggestion is motivated by the fact that the algorithms in chapter 4 are defined independently of the underlying logic, so no alteration to the algorithms is required. For the practical use of the AGM system, the logic would have to be restricted to a language with a tractable decision procedure for derivability.

The algorithms presented in chapter 4 are simple to understand and implement, so that the AGM system can be easily modified to incorporate alternative belief change operations, built upon the existing program. This would allow an automated experimental comparison of the various operations instead of employing the error-prone procedure of testing out new ideas on the standard examples by hand.

The minimal change entrenchment revision policy advocated within this thesis is motivated by the desire to preserve as much as possible of an epistemic state during belief change operations. Very little work has been done in this area to demonstrate whether or not this policy is always the most rational approach to generating a new entrenchment relation. The implementation provides an opportunity to generate and test new ideas in entrenchment revision.

Apart from extensions to the belief revision system itself, the work with truth maintenance and nonmonotonic reasoning suggests that the system could be used to implement other styles of reasoning. We described several possible extensions to the ATMS in chapter 6, to allow for non-foundational reasoning, and also for a more general form of foundational reasoning using a more expressive language, and including the possibility of revising justifications. Theoretical work linking belief revision with some other reasoning formalisms is outlined in chapter 7; further work could put some of these relationships into use as the basis for other implementations.

The obvious application of belief revision to database update has yet to be realised in a practical way; this is another interesting avenue of further research, which could have far-reaching effects outside the AI community.

APPENDICES

A Proof of Theorem 3.1

We prove that the relation computed from an entrenchment base satisfying conditions (R1) and (R2), via the most conservative entrenchment construction, also satisfies conditions (EE1) to (EE5) for an epistemic entrenchment.

Assume we are given an entrenchment base $\Gamma = (\Gamma_1, \Gamma_2, \cdots, \Gamma_n)$ satisfying:

$$\text{(R1)} \quad \forall\, i, \ \ \forall\, \beta \in \Gamma_i, \ \overline{\Gamma_{i+1}} \nvdash \beta$$
$$\text{(R2)} \quad \alpha \in \Gamma_n \ \text{ if and only if } \ \vdash \alpha$$

a function **rank** satisfying (Def_Rank):

$$\mathbf{rank}(\Gamma, \alpha) \ = \begin{cases} \max(\{i : \overline{\Gamma_i} \vdash \alpha\}) & \text{if } \overline{\Gamma_1} \vdash \alpha \\ 0 & \text{if } \overline{\Gamma_1} \nvdash \alpha \end{cases}$$

and a relation $\leq_E$ given by (Def_MCE):

$$\alpha \leq_E \beta \quad \text{iff} \quad \mathbf{rank}(\Gamma, \alpha) \leq \mathbf{rank}(\Gamma, \beta)$$

The relation $\leq_E$ is called the most conservative entrenchment generated from the entrenchment base Γ. Define $\alpha =_E \beta$ if and only if $\alpha \leq_E \beta$ and $\beta \leq_E \alpha$; and $\alpha <_E \beta$ if and only if $\alpha \leq_E \beta$ and not $\beta \leq_E \alpha$.

(EE1) If $\alpha \leq_E \beta$ and $\beta \leq_E \gamma$ then $\alpha \leq_E \gamma$

Proof: Choose any α, β, γ such that $\alpha \leq_E \beta$ and $\beta \leq_E \gamma$.
Then by (Def_MCE), $\mathbf{rank}(\Gamma, \alpha) \leq \mathbf{rank}(\Gamma, \beta)$, and also $\mathbf{rank}(\Gamma, \beta) \leq \mathbf{rank}(\Gamma, \gamma)$.
Since the ranks are natural numbers, we conclude $\mathbf{rank}(\Gamma, \alpha) \leq \mathbf{rank}(\Gamma, \gamma)$.
So, by (Def_MCE), $\alpha \leq_E \gamma$.

(EE2) If $\alpha \vdash \beta$ then $\alpha \leq_E \beta$

Proof: Choose any α, β, such that $\alpha \vdash \beta$.

Suppose $\alpha >_E \beta$.

Then $\mathbf{rank}(\Gamma, \alpha) > \mathbf{rank}(\Gamma, \beta)$.

Let $i = \mathbf{rank}(\Gamma, \alpha)$.

From (Def_Rank), $\overline{\Gamma_i} \vdash \alpha$, and since $\alpha \vdash \beta$, we conclude that $\overline{\Gamma_i} \vdash \beta$.

Now $\overline{\Gamma_i} \subseteq \{\gamma \in \Gamma : \mathbf{rank}(\Gamma, \beta) < \mathbf{rank}(\Gamma, \gamma)\}$.

Hence $\{\gamma \in \Gamma : \mathbf{rank}(\Gamma, \beta) < \mathbf{rank}(\Gamma, \gamma)\} \vdash \beta$, contradicting (R1).

Thus $\alpha \leq_E \beta$.

(EE3) For any α and β, $\alpha \leq_E \alpha\wedge\beta$ or $\beta \leq_E \alpha\wedge\beta$

Proof: Suppose that there exists α, β such that $\alpha >_E \alpha\wedge\beta$ and $\beta >_E \alpha\wedge\beta$.

Without loss of generality, assume $\alpha \geq_E \beta$.

Let $i = \mathbf{rank}(\Gamma, \alpha)$ and $j = \mathbf{rank}(\Gamma, \beta)$.

Then $i \geq j$, and $\overline{\Gamma_i} \subseteq \overline{\Gamma_j}$.

By definition $\overline{\Gamma_i} \vdash \alpha$, so we have $\overline{\Gamma_j} \vdash \alpha$.

Combining this with $\overline{\Gamma_j} \vdash \beta$ gives $\overline{\Gamma_j} \vdash \alpha\wedge\beta$.

Since $\mathbf{rank}(\Gamma, \alpha\wedge\beta) < j$, we have $\overline{\Gamma_j} \subseteq \{\gamma \in \Gamma : \mathbf{rank}(\Gamma, \alpha\wedge\beta) < \mathbf{rank}(\Gamma, \gamma)\}$.

Thus $\{\gamma \in \Gamma : \mathbf{rank}(\Gamma, \alpha\wedge\beta) < \mathbf{rank}(\Gamma, \gamma)\} \vdash \alpha\wedge\beta$, contradicting (R1).

Hence no such α and β exist.

(EE4) When $K \neq K_\perp$, $\alpha \notin K$ iff $\alpha \leq_E \beta$ for all β

Proof: Assume $K \neq K_\perp$.

Suppose $\alpha \in K$.

Then $\overline{\Gamma_1} \vdash K$, and hence $\mathbf{rank}(\Gamma, \alpha) = \max(\{i : \overline{\Gamma_i} \vdash \alpha\}) \geq 1$.

Also $\neg\alpha \notin K$, since $K \neq K_\perp$, so by definition, $\mathbf{rank}(\Gamma, \neg\alpha) = 0$.

Then there exists β such that $\mathbf{rank}(\Gamma, \alpha) > \mathbf{rank}(\Gamma, \beta)$, that is $\alpha >_E \beta$.

So it is not the case that $\alpha \leq_E \beta$ for all β.

Alternatively, suppose $\alpha \notin K$.

Then $\mathbf{rank}(\Gamma, \alpha) = 0$.

Clearly, from (Def_Rank), $\mathbf{rank}(\Gamma, \beta) \geq 0$ for all β.

Thus for all β, $\mathbf{rank}(\Gamma, \alpha) \leq \mathbf{rank}(\Gamma, \beta)$, which gives $\alpha \leq_E \beta$ for all β.

(EE5) If $\beta \leq_E \alpha$ for all β then $\vdash \alpha$

Proof: Choose α such that $\beta \leq_E \alpha$ for all β.

Then $\mathbf{rank}(\Gamma, \beta) \leq \mathbf{rank}(\Gamma, \alpha)$ for all β.

A

When $\vdash \beta$, **rank**$(\Gamma, \beta) = n$ by (R2), so **rank**$(\Gamma, \alpha) \geq n$.
If $\overline{\Gamma_1} \nvdash \alpha$ then $\alpha \notin K$, and **rank**$(\Gamma, \alpha) = 0$, and we have a contradiction.
Otherwise $\overline{\Gamma_1} \vdash \alpha$.
Now if $\vdash \alpha$, we are done, so suppose $\nvdash \alpha$.
Then $\overline{\Gamma_n} \nvdash \alpha$, since $\vdash \overline{\Gamma_n}$, by (R2).
Then there exists a greatest i such that $\overline{\Gamma_i} \vdash \alpha$ and $i < n$.
Hence **rank**$(\Gamma, \alpha) < n$, and we have a contradiction.

Thus we have shown that the relation generated from a ranked finite base satisfying conditions (R1) and (R2) is an epistemic entrenchment ordering, which we name the most conservative entrenchment.

B The Correctness of the Expansion Algorithm

B.1 Lemma

Let $\Delta = \mathbf{expand}(\Gamma, \alpha, r)$, for any entrenchment base Γ, formula α and rank *r*. Then for all formulae β, $\mathbf{rank}(\Delta, \beta) \geq \mathbf{rank}(\Gamma, \beta)$.

Proof: Let $i = \mathbf{rank}(\Gamma, \beta)$.
Consider any proof $\Phi \vdash \beta$, where $\Phi \subseteq \overline{\Gamma_i}$ and for all $\Phi' \subset \Phi$, $\Phi' \nvdash \beta$.
For each $\phi \in \Phi$, $\phi \in \Gamma_j$, for some $j \geq i$.
Also, by inspection of the expansion algorithm, we have either $\phi \in \Delta_j$ or $\overline{\Delta_j} \vdash \phi$.
i.e., ϕ is only deleted if it is derivable with at least the same rank as it had previously.
Since $i \leq j$, $\overline{\Delta_i} \vdash \phi$.
Hence $\overline{\Delta_i} \vdash \Phi$, and since $\Phi \vdash \beta$, $\overline{\Delta_i} \vdash \beta$.
So $\mathbf{rank}(\Delta, \beta) \geq i$; that is, $\mathbf{rank}(\Delta, \beta) \geq \mathbf{rank}(\Gamma, \beta)$.

B.2 Proof of (R+) Condition

We now prove that any expansion operation $\Delta = \mathbf{expand}(\Gamma, \alpha, r)$ satisfies the (R+) condition,

$$\mathbf{rank}(\Delta, \beta) = \begin{cases} \min(\mathbf{rank}(\Gamma, \alpha\rightarrow\beta), r) & \text{if } \overline{\Gamma_1} \nvdash \neg\alpha \text{ and} \\ & \quad \mathbf{rank}(\Gamma, \beta) < \min(\mathbf{rank}(\Gamma, \alpha\rightarrow\beta), r) \\ \mathbf{rank}(\Gamma, \beta) & \text{otherwise} \end{cases}$$

Let β be any formula.

Case (1): $\overline{\Gamma_1} \vdash \neg\alpha$.
Then $\Delta = \Gamma$, and hence $\mathbf{rank}(\Delta, \beta) = \mathbf{rank}(\Gamma, \beta)$, as required.
So we may assume for the remaining cases that $\overline{\Gamma_1} \nvdash \neg\alpha$.

Case (2): $\textbf{rank}(\Gamma, \alpha \to \beta) \leq \textbf{rank}(\Gamma, \beta)$.

Consider any proof $\Phi \vdash \beta$ where $\Phi \subseteq \overline{\Delta_j}$ for some j, and for all $\Phi' \subset \Phi$, $\Phi' \nvdash \beta$.

Now suppose $j > \textbf{rank}(\Gamma, \beta)$.

Let $\Psi = \Phi - \{\alpha\}$.

Then by the definition of expansion, $\Psi \subseteq \overline{\Gamma_j}$.

Therefore $\overline{\Gamma_j} \cup \{\alpha\} \vdash \beta$, and by the deduction theorem $\overline{\Gamma_j} \vdash \alpha \to \beta$.

Thus $\textbf{rank}(\Gamma, \alpha \to \beta) \geq j$.

Since $j > \textbf{rank}(\Gamma, \beta)$, we deduce $\textbf{rank}(\Gamma, \alpha \to \beta) > \textbf{rank}(\Gamma, \beta)$.

This contradicts our initial condition.

Thus no such j exists, so we conclude that $\textbf{rank}(\Delta, \beta) \leq \textbf{rank}(\Gamma, \beta)$.

Combining this with Lemma B.1 gives $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Gamma, \beta)$, as required.

Case (3): $r \leq \textbf{rank}(\Gamma, \beta)$.

Consider any proof $\Phi \vdash \beta$, where $\Phi \subseteq \overline{\Delta_j}$ for some j, and for all $\Phi' \subset \Phi$, $\Phi' \nvdash \beta$.

Now suppose $j > \textbf{rank}(\Gamma, \beta)$.

(a) Suppose $\alpha \in \Phi$.

Then since $j > r$, by the definition of expansion, $\textbf{rank}(\Gamma, \alpha) > r$, and hence $\Delta = \Gamma$.

Therefore $\Phi \subseteq \overline{\Gamma_j}$, and hence $\overline{\Gamma_j} \vdash \beta$.

Thus $\textbf{rank}(\Gamma, \beta) \geq j$, contradicting the choice of j.

(b) Assume $\alpha \notin \Phi$.

Then $\Phi \subseteq \overline{\Gamma_j}$, and hence $\overline{\Gamma_j} \vdash \beta$.

Thus $\textbf{rank}(\Gamma, \beta) \geq j$, contradicting the choice of j.

We conclude that no such $j > \textbf{rank}(\Gamma, \beta)$ exists.

Hence $\textbf{rank}(\Delta, \beta) \leq \textbf{rank}(\Gamma, \beta)$.

Combining this with Lemma B.1 gives $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Gamma, \beta)$, as required.

Case (4): Otherwise $\textbf{rank}(\Gamma, \alpha \to \beta) > \textbf{rank}(\Gamma, \beta)$ and $r > \textbf{rank}(\Gamma, \beta)$.

Let $j = \min(\textbf{rank}(\Gamma, \alpha \to \beta), r)$.

Then $j > \textbf{rank}(\Gamma, \beta)$.

Suppose $\textbf{rank}(\Gamma, \alpha) \geq r$.

Then since $j \leq r$, $\overline{\Gamma_j} \vdash \alpha$.

By the choice of j, we also have $\overline{\Gamma_j} \vdash \alpha \to \beta$.

By modus ponens, $\overline{\Gamma_j} \vdash \beta$, and hence $\textbf{rank}(\Gamma, \beta) \geq j$, contradicting the choice of j.

Therefore, $\textbf{rank}(\Gamma, \alpha) < r$.

Then, by the definition of expansion, $\alpha \in \Delta_r$, and hence $\overline{\Delta_j} \vdash \alpha$.

Also by Lemma B.1, $\mathbf{rank}(\Delta, \alpha\to\beta) \geq \mathbf{rank}(\Gamma, \alpha\to\beta)$, so $\overline{\Delta_j} \vdash \alpha\to\beta$.

By modus ponens, $\overline{\Delta_j} \vdash \beta$, and hence $\mathbf{rank}(\Delta, \beta) \geq j$.

Now suppose $\mathbf{rank}(\Delta, \beta) > j$.

Then there exists some minimal set $\Phi \subseteq \overline{\Delta_k}$ such that $\Phi \vdash \beta$, for some $k > j$.

(a) Suppose $\alpha \in \Phi$

Then $r \geq k > j$, since we have already shown that $\mathbf{rank}(\Gamma, \alpha) < r$.

Let $\Psi = \Phi - \{\alpha\}$.

$\Psi \subseteq \overline{\Gamma_k}$, so $\overline{\Gamma_k} \cup \{\alpha\} \vdash \beta$, and by the deduction theorem $\overline{\Gamma_k} \vdash \alpha\to\beta$.

So $\mathbf{rank}(\Gamma, \alpha\to\beta) \geq k$, and hence $\mathbf{rank}(\Gamma, \alpha\to\beta) > j$.

But this gives $\min(\mathbf{rank}(\Gamma, \alpha\to\beta), r) > j$, contradicting the choice of j.

(b) Otherwise $\alpha \notin \Phi$.

Then $\Phi \subseteq \overline{\Gamma_k}$, and hence $\overline{\Gamma_k} \vdash \beta$.

Thus $\mathbf{rank}(\Gamma, \beta) \geq k$, and hence $\mathbf{rank}(\Gamma, \beta) > j$, contradicting the choice of j.

Therefore, no such Φ exists, and we may conclude that $\mathbf{rank}(\Delta, \beta) = j$.

That is, $\mathbf{rank}(\Delta, \beta) = \min(\mathbf{rank}(\Gamma, \alpha\to\beta), r)$, as required.

B.3 Satisfaction of AGM Postulates

Finally, we show that any expansion function satisfying the (R+) condition also satisfies the AGM rationality postulates for consistent expansion. Consider an expansion $\Delta = \mathbf{expand}(\Gamma, \alpha, r)$, for any consistent entrenchment base Γ, formula α and integer $r > 0$, such that $\overline{\Gamma_1} \nvdash \neg\alpha$. The AGM postulates are satisfied if and only if:

$$Cn(\overline{\Delta_1}) = Cn(\overline{\Gamma_1} \cup \{\alpha\}).$$

That is, for any β:

$$\overline{\Delta_1} \vdash \beta \quad \text{iff} \quad \overline{\Gamma_1} \cup \{\alpha\} \vdash \beta$$

By the deduction theorem, this is equivalent to:

$$\overline{\Delta_1} \vdash \beta \quad \text{iff} \quad \overline{\Gamma_1} \vdash \alpha\to\beta$$

Expressing this in ranks:

$$\mathbf{rank}(\Delta, \beta) > 0 \quad \text{iff} \quad \mathbf{rank}(\Gamma, \alpha\to\beta) > 0.$$

Proof:

Case (1): **rank**(Γ, β) $<$ min(**rank**(Γ, α→β), r).

Then **rank**(Γ, α→β) $>$ **rank**(Γ, β) ≥ 0, so **rank**(Γ, α→β) > 0.

Also, by the (R+) condition, **rank**(Δ, β) $=$ min(**rank**(Γ, α→β), r) > 0.

Case (2): Otherwise, suppose that **rank**(Γ, β) > 0.

Then by (EE2), **rank**(Γ, α→β) > 0.

Also, by the (R+) condition, **rank**(Δ, β) $=$ **rank**(Γ, β) > 0, as required.

Case (3): Otherwise **rank**(Γ, β) $= 0$.

Now since $r > 0$, and Γ did not satisfy Case (1), we must have **rank**(Γ, α→β) $= 0$.

So by the (R+) condition, **rank**(Δ, β) $=$ **rank**(Γ, β) $= 0$, and the proof is complete.

C The Correctness of the Contraction Algorithm

C.1 Lemma

Let $\Delta = \textbf{contract}(\Gamma, \alpha)$, for any entrenchment base Γ, and formula α. Then for all formulae β, $\textbf{rank}(\Delta, \beta) \le \textbf{rank}(\Gamma, \beta)$.

Proof: Let $i = \textbf{rank}(\Delta, \beta)$.
Consider any proof $\Phi \vdash \beta$ such that $\Phi \subseteq \overline{\Delta_i}$, for some i.
For each $\phi \in \Phi$, $\phi \in \Delta_j$, for some $j \ge i$.
Then either $\phi \in \Gamma_j$ or $\phi' \in \Gamma_j$, where $\phi \equiv \alpha \to \phi'$.
Now $\phi' \vdash \alpha \to \phi'$, for all α and ϕ', so $\Gamma_j \vdash \phi$ in both cases.
Since $i \le j$, $\overline{\Gamma_i} \vdash \phi$, for all $\phi \in \Phi$.
Hence $\overline{\Gamma_i} \vdash \Phi$, and since $\Phi \vdash \beta$, $\overline{\Gamma_i} \vdash \beta$.
So $\textbf{rank}(\Gamma, \beta) \ge i$; that is, $\textbf{rank}(\Delta, \beta) \le \textbf{rank}(\Gamma, \beta)$.

C.2 Lemma

Let $\Delta = \textbf{contract}(\Gamma, \alpha)$, for any entrenchment base Γ, and formula α. Then for all formulae β, if $\textbf{rank}(\Gamma, \beta) > \textbf{rank}(\Gamma, \alpha)$ then $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Gamma, \beta)$.

Proof: Suppose $\textbf{rank}(\Gamma, \beta) > \textbf{rank}(\Gamma, \alpha)$.
Let $i = \textbf{rank}(\Gamma, \beta)$, and let $\phi \in \overline{\Gamma_i}$.
Then $\textbf{rank}(\Gamma, \alpha \vee \phi) \ge \textbf{rank}(\Gamma, \phi)$, by (EE2).
Therefore $\textbf{rank}(\Gamma, \alpha \vee \phi) > \textbf{rank}(\Gamma, \alpha)$.
Then $\phi \in \overline{\Delta_i}$, by the definition of contraction.
Hence $\overline{\Gamma_i} = \overline{\Delta_i}$, so $\overline{\Delta_i} \vdash \beta$, and $\textbf{rank}(\Delta, \beta) \ge i$.
That is, $\textbf{rank}(\Delta, \beta) \ge \textbf{rank}(\Gamma, \beta)$.
Combining this result with Lemma C.1 gives $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Gamma, \beta)$.

C.3 Lemma

Let $\Delta = \mathbf{contract}(\Gamma, \alpha)$, for any entrenchment base Γ, and formula α. Then for any formula β, if $\beta \in \Gamma_i$ and $\beta \notin \Delta_i$ and $\alpha \to \beta \notin \Delta_i$ then $\mathbf{rank}(\Delta, \alpha \to \beta) > i$.

Proof: Note that $\mathbf{rank}(\Gamma, \beta) = i$.
By the contrapositive of Lemma C.2, $\mathbf{rank}(\Gamma, \beta) \leq \mathbf{rank}(\Gamma, \alpha)$.
And by the definition of contraction, $\mathbf{rank}(\Gamma, \beta) \geq \mathbf{rank}(\Gamma, \alpha)$.
Thus $\mathbf{rank}(\Gamma, \beta) = \mathbf{rank}(\Gamma, \alpha)$.
Also from the definition of contraction, $\mathbf{rank}(\Gamma, \alpha \to \beta) \neq \mathbf{rank}(\Gamma, \alpha)$.
Combining with the previous result implies: $\mathbf{rank}(\Gamma, \alpha \to \beta) > \mathbf{rank}(\Gamma, \alpha)$.
By Lemma C.2 again, $\mathbf{rank}(\Delta, \alpha \to \beta) = \mathbf{rank}(\Gamma, \alpha \to \beta)$.
Thus $\mathbf{rank}(\Delta, \alpha \to \beta) > \mathbf{rank}(\Gamma, \alpha)$.
That is, $\mathbf{rank}(\Delta, \alpha \to \beta) > \mathbf{rank}(\Gamma, \beta)$.
Thus we have shown $\mathbf{rank}(\Delta, \alpha \to \beta) > i$.

C.4 Lemma

Let $\Delta = \mathbf{contract}(\Gamma, \alpha)$, for any entrenchment base Γ, and formula α. Then for any formula β, $\mathbf{rank}(\Delta, \alpha \to \beta) \geq \mathbf{rank}(\Gamma, \beta)$.

Proof: Let $i = \mathbf{rank}(\Gamma, \beta)$.
Now if $i = 0$ we are done, so suppose $i > 0$.
Consider any proof $\Phi \vdash \beta$ such that $\Phi \subseteq \overline{\Gamma_i}$.
For each $\phi \in \Phi$, $\phi \in \Gamma_j$, for some $j \geq i$.
Then if $\phi \in \Delta_j$, $\mathbf{rank}(\Delta, \alpha \to \phi) \geq j$ since $\phi \vdash \alpha \to \phi$.
Also, if $\alpha \to \phi \in \Delta_j$, then $\mathbf{rank}(\Delta, \alpha \to \phi) \geq j$.
Otherwise, Lemma C.3 applies, and we have $\mathbf{rank}(\Delta, \alpha \to \phi) \geq j$.
Since $i \leq j$, for any $\phi \in \Phi$, $\overline{\Delta_i} \vdash \alpha \to \phi$.
By the deduction theorem, $\overline{\Delta_i} \cup \{\alpha\} \vdash \phi$.
Thus $\overline{\Delta_i} \cup \{\alpha\} \vdash \Phi$, and since $\Phi \vdash \beta$, $\overline{\Delta_i} \cup \{\alpha\} \vdash \beta$.
Hence by the deduction theorem again, $\overline{\Delta_i} \vdash \alpha \to \beta$.
Thus $\mathbf{rank}(\Delta, \alpha \to \beta) \geq i$; that is, $\mathbf{rank}(\Delta, \alpha \to \beta) \geq \mathbf{rank}(\Gamma, \beta)$.

C.5 Proof of (R–) Condition

Let $\Delta = \textbf{contract}(\Gamma, \alpha)$, for any entrenchment base Γ, and formula α, and let β be any formula. We now prove that the contraction algorithm satisfies the (R–) condition,

$$\textbf{rank}(\Delta, \beta) = \begin{cases} \textbf{rank}(\Gamma, \beta) & \text{if } \vdash\alpha \text{ or } \textbf{rank}(\Gamma, \alpha) < \textbf{rank}(\Gamma, \alpha\vee\beta) \\ 0 & \text{otherwise} \end{cases}$$

Case (1): $\vdash \alpha$.

Then $\Delta = \Gamma$, and therefore $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Gamma, \beta)$, as required.

Case (2): $\nvdash \alpha$ and $\textbf{rank}(\Gamma, \alpha) = \textbf{rank}(\Gamma, \alpha\vee\beta)$.

Consider any proof $\Phi \vdash \alpha\vee\beta$ such that $\Phi \subseteq \overline{\Gamma_1}$.

Suppose $\Phi \subseteq \overline{\Delta_1}$.

Then for all $\phi \in \Phi$, $\textbf{rank}(\Gamma, \alpha) < \textbf{rank}(\Gamma, \alpha\vee\phi)$.

Let $j = \min(\{\textbf{rank}(\Gamma, \alpha\vee\phi) : \phi \in \Phi\})$.

Then $j > \textbf{rank}(\Gamma, \alpha)$.

Also $\overline{\Gamma_j} \vdash \alpha\vee\phi$, for all $\phi \in \Phi$.

By the deduction theorem, $\overline{\Gamma_j} \cup \{\neg\alpha\} \vdash \phi$, for all $\phi \in \Phi$.

Thus $\overline{\Gamma_j} \cup \{\neg\alpha\} \vdash \Phi$, and since $\Phi \vdash \alpha\vee\beta$, we have $\overline{\Gamma_j} \cup \{\neg\alpha\} \vdash \alpha\vee\beta$.

Thus $\overline{\Gamma_j} \vdash \alpha\vee(\alpha\vee\beta)$, by the deduction theorem.

That is, $\overline{\Gamma_j} \vdash \alpha\vee\beta$.

So $\textbf{rank}(\Gamma, \alpha\vee\beta) \geq j$.

Hence $\textbf{rank}(\Gamma, \alpha\vee\beta) > \textbf{rank}(\Gamma, \alpha)$, contradicting the initial condition.

Therefore, for any subset $\Phi \subseteq \overline{\Gamma_1}$ such that $\Phi \vdash \alpha\vee\beta$, $\Phi \not\subseteq \overline{\Delta_1}$.

Thus $\overline{\Delta_1} \nvdash \beta$, and hence $\textbf{rank}(\Delta, \beta) = 0$, as required.

Case (3): $\nvdash \alpha$ and $\textbf{rank}(\Gamma, \alpha) < \textbf{rank}(\Gamma, \alpha\vee\beta)$.

If $\textbf{rank}(\Gamma, \beta) > \textbf{rank}(\Gamma, \alpha)$, by Lemma C.2, $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Gamma, \beta)$, as required.

Otherwise $\textbf{rank}(\Gamma, \beta) \leq \textbf{rank}(\Gamma, \alpha)$.

Then by Lemma C.2, $\textbf{rank}(\Delta, \alpha\vee\beta) > \textbf{rank}(\Gamma, \alpha)$.

Thus $\textbf{rank}(\Delta, \alpha\vee\beta) > \textbf{rank}(\Gamma, \beta)$.

Also, by Lemma C.4, $\textbf{rank}(\Delta, \alpha\rightarrow\beta) \geq \textbf{rank}(\Gamma, \beta)$.

Let $j = \textbf{rank}(\Gamma, \beta)$.

Then $\overline{\Delta_j} \vdash \alpha\vee\beta$ and $\overline{\Delta_j} \vdash \alpha\rightarrow\beta$.

Therefore, $\overline{\Delta_j} \vdash \beta$.
Thus **rank**$(\Delta, \beta) \geq j$.
That is, **rank**$(\Delta, \beta) \geq$ **rank**(Γ, β).
Combining this with Lemma C.1 gives **rank**$(\Delta, \beta) =$ **rank**(Γ, β), as required.

C.6 Satisfaction of AGM Postulates

We now show that any contraction function satisfying the (R–) condition also satisfies the AGM rationality postulates for contraction. Consider a contraction $\Delta =$ **contract**(Γ, α), for any consistent entrenchment base Γ and formula α. The AGM postulates are satisfied if and only if the (C–) condition holds:

$$\beta \in K^-_\alpha \text{ iff } \beta \in K \text{ and either } \vdash \alpha \text{ or } \alpha <_E \alpha \vee \beta$$

Expressing this in terms of ranks:

$$\textbf{rank}(\Delta, \beta) > 0 \quad \text{iff} \quad \textbf{rank}(\Gamma, \beta) > 0 \text{ and either } \vdash \alpha \text{ or } \textbf{rank}(\Gamma, \alpha) < \textbf{rank}(\Gamma, \alpha \vee \beta)$$

Proof:
Case (1): $\vdash \alpha$.
Then by the (R–) condition, **rank**$(\Delta, \beta) =$ **rank**(Γ, β).
Hence **rank**$(\Delta, \beta) > 0$ iff **rank**$(\Gamma, \beta) > 0$ as required.

Case (2): **rank**$(\Gamma, \alpha) <$ **rank**$(\Gamma, \alpha \vee \beta)$.
Then by the (R–) condition, **rank**$(\Delta, \beta) =$ **rank**(Γ, β).
Again, **rank**$(\Delta, \beta) > 0$ iff **rank**$(\Gamma, \beta) > 0$ as required.

Case (3): Otherwise, $\nvdash \alpha$ and **rank**$(\Gamma, \alpha) =$ **rank**$(\Gamma, \alpha \vee \beta)$.
Then by the (R–) condition, **rank**$(\Delta, \beta) = 0$, as required.
Hence the contraction algorithm satisfies the AGM postulates.

D The Correctness of the Revision Algorithm

D.1 Proof of (R∗) Condition

Let $\Delta = \textbf{revise}(\Gamma, \alpha, r)$, for any entrenchment base Γ, formula α, and rank r, and let β be any formula. We now prove that revision algorithm satisfies the (R∗) condition,

$$\textbf{rank}(\Delta, \beta) = \begin{cases} 0 & \text{if } \textbf{rank}(\Gamma, \neg\alpha) \geq \textbf{rank}(\Gamma, \alpha\rightarrow\beta) \\ \textbf{rank}(\Gamma, \beta) & \text{if } \textbf{rank}(\Gamma, \neg\alpha) < \textbf{rank}(\Gamma, \alpha\rightarrow\beta) \\ & \quad\text{and } (\textbf{rank}(\Gamma, \alpha\rightarrow\beta) \leq \textbf{rank}(\Gamma, \beta) \\ & \qquad\text{or } r \leq \textbf{rank}(\Gamma, \beta)) \\ \min(\textbf{rank}(\Gamma, \alpha\rightarrow\beta), r) & \text{otherwise} \end{cases}$$

Let $\Delta' = \textbf{contract}(\Gamma, \neg\alpha)$, so that $\Delta = \textbf{expand}(\Delta', \alpha, r)$.

Case (1): $\vdash \alpha$.
Then $\textbf{rank}(\Gamma, \neg\alpha) = 0$.
If $\textbf{rank}(\Gamma, \beta) > 0$ then by Lemma C.2, $\textbf{rank}(\Delta', \beta) = \textbf{rank}(\Gamma, \beta)$.
Otherwise by Lemma C.1 $\textbf{rank}(\Delta', \beta) = 0$, and therefore $\textbf{rank}(\Delta', \beta) = \textbf{rank}(\Gamma, \beta)$.
Then $\textbf{rank}(\Delta', \beta) = \textbf{rank}(\Delta', \alpha\rightarrow\beta)$, by (R1) and since $\vdash \alpha$.
Then by the (R+) condition, $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Delta', \beta)$.
Hence $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Gamma, \beta)$.

Case (2): $\vdash \neg\alpha$.
Then $\Delta' = \Gamma$, and $\Delta = \Delta'$, by the definitions of contraction and expansion.
Thus $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Gamma, \beta)$.

We assume for the remaining cases that $\not\vdash \alpha$ and $\not\vdash \neg\alpha$.

Case (3): $\textbf{rank}(\Gamma, \neg\alpha) \geq \textbf{rank}(\Gamma, \alpha\rightarrow\beta)$.
By the (R–) condition, $\textbf{rank}(\Delta', \beta) = 0$.
Also by the (R–) condition, $\textbf{rank}(\Delta', \alpha\rightarrow\beta) = 0$.
So by the (R+) condition, $\textbf{rank}(\Delta, \beta) = \textbf{rank}(\Delta', \beta)$.

D.1

That is $\mathbf{rank}(\Delta, \beta) = 0$, as required.

Case (4): $\mathbf{rank}(\Gamma, \neg\alpha) < \mathbf{rank}(\Gamma, \alpha\to\beta)$ and $\mathbf{rank}(\Gamma, \alpha\to\beta) \le \mathbf{rank}(\Gamma, \beta)$.
By (EE2), $\mathbf{rank}(\Gamma, \beta) \le \mathbf{rank}(\Gamma, \alpha\to\beta)$, so $\mathbf{rank}(\Gamma, \alpha\to\beta) = \mathbf{rank}(\Gamma, \beta)$.
By the (R–) condition, $\mathbf{rank}(\Delta', \beta) = \mathbf{rank}(\Gamma, \beta)$.
Also by the (R–) condition, $\mathbf{rank}(\Delta', \alpha\to\beta) = \mathbf{rank}(\Gamma, \alpha\to\beta)$.
Therefore $\mathbf{rank}(\Delta', \beta) = \mathbf{rank}(\Delta', \alpha\to\beta)$.
So by the (R+) condition, $\mathbf{rank}(\Delta, \beta) = \mathbf{rank}(\Delta', \beta)$.
That is $\mathbf{rank}(\Delta, \beta) = \mathbf{rank}(\Gamma, \beta)$, as required.

Case (5): $\mathbf{rank}(\Gamma, \neg\alpha) < \mathbf{rank}(\Gamma, \alpha\to\beta)$ and $r \le \mathbf{rank}(\Gamma, \beta)$.
Again, by (R–), $\mathbf{rank}(\Delta', \beta) = \mathbf{rank}(\Gamma, \beta)$ and $\mathbf{rank}(\Delta', \alpha\to\beta) = \mathbf{rank}(\Gamma, \alpha\to\beta)$.
Then since $r \le \mathbf{rank}(\Delta', \beta)$, $\mathbf{rank}(\Delta, \beta) = \mathbf{rank}(\Delta', \beta)$, by the (R+) condition.
Thus $\mathbf{rank}(\Delta, \beta) = \mathbf{rank}(\Gamma, \beta)$, as required.

Case (6): Otherwise.
$\mathbf{rank}(\Gamma, \neg\alpha) < \mathbf{rank}(\Gamma, \alpha\to\beta)$, $\mathbf{rank}(\Gamma, \alpha\to\beta) > \mathbf{rank}(\Gamma, \beta)$ and $r > \mathbf{rank}(\Gamma, \beta)$.
Again, by (R–), $\mathbf{rank}(\Delta', \beta) = \mathbf{rank}(\Gamma, \beta)$ and $\mathbf{rank}(\Delta', \alpha\to\beta) = \mathbf{rank}(\Gamma, \alpha\to\beta)$.
So $\mathbf{rank}(\Delta', \beta) < \mathbf{rank}(\Delta', \alpha\to\beta)$ and $\mathbf{rank}(\Delta', \beta) < r$.
Hence by (R+), $\mathbf{rank}(\Delta, \beta) = \min(\mathbf{rank}(\Delta', \alpha\to\beta), r)$.
That is $\mathbf{rank}(\Delta, \beta) = \min(\mathbf{rank}(\Gamma, \alpha\to\beta), r)$, as required.

D.2 Satisfaction of AGM Postulates

Since we have implemented the revision operation as the composition of a contraction and expansion operation, both of which satisfy the AGM postulates, then by the Levi identity, the resulting revision operation satisfies the AGM postulates for revision.

E The Correctness of ATMS_Algorithm_1

Before proving that the algorithm correctly simulates the behaviour of the ATMS, it is necessary to prove two lemmas which are used in proving the correctness of **ATMS_Algorithm_1**. The first lemma establishes the relationship between entrenchment and the notion of essential support, and the second lemma relates essential support to ATMS provability.

E.1 Lemma

In any environment E, for all $a \in E$, and for all b such that $E \vdash_{ATMS} b$:

$$\mathbf{rank}(\Gamma_E, a) = \mathbf{rank}(\Gamma_E, a \vee b) \quad \text{iff} \quad a \in ES(b,E)$$

Proof:
This lemma will be proved by induction on the number k of expansion and contraction operations performed by the algorithm. Let E represent the current environment (E_{Old} partially updated k times), and E' the same environment updated $k+1$ times.

Initial case: Initially, $E = \varnothing$, and hence the lemma is trivially satisfied.

Inductive Hypothesis: Assume that after the first k expansions and contractions, we have $\mathbf{rank}(\Gamma_E, a) = \mathbf{rank}(\Gamma_E, a \vee b)$ iff $a \in ES(b,E)$. Then let the $(k+1)$st operation expand or contract E by x, giving the new environment E'.

Case (1): Expansion

Part (i): Suppose $a \in ES(b,E)$.
Then from the inductive hypothesis we have $\mathbf{rank}(\Gamma_E, a) = \mathbf{rank}(\Gamma_E, a \vee b)$.

Suppose $x \notin \cup Label(b)$.
Then $FB(b,E) = FB(b,E')$, so $ES(b,E) = ES(b,E')$, and $a \in ES(b,E')$.

E.1

Also, the ranks of a and $a \vee b$ are unchanged.

Hence $\mathbf{rank}(\Gamma_{E'}, a) = \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Otherwise, $x \in \cup Label(b)$.

Then we must consider two possibilities:

Firstly, if $a \in ES(b,E')$ then $a \notin (ES(b,E) - ES(b,E'))$.

Also, the ranks of a and $a \vee b$ are unchanged.

Hence, from the inductive hypothesis, $\mathbf{rank}(\Gamma_{E'}, a) = \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Secondly, if $a \notin ES(b,E')$, then $a \in (ES(b,E) - ES(b,E'))$.

Then the rank of $a \vee b$ is changed to 20, giving $\mathbf{rank}(\Gamma_{E'}, a) < \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Part (ii): Now consider the case when $a \notin ES(b,E)$.

Then from the inductive hypothesis we have $\mathbf{rank}(\Gamma_{E}, a) < \mathbf{rank}(\Gamma_{E}, a \vee b)$.

Suppose $x \notin \cup Label(b)$.

Then $FB(b,E) = FB(b,E')$, and so $ES(b,E) = ES(b,E')$, and $a \notin ES(b,E')$.

As above, the ranks of a and $a \vee b$ are unchanged.

Hence $\mathbf{rank}(\Gamma_{E'}, a) < \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Otherwise, $x \in \cup Label(b)$.

Then once more we must consider two possibilities:

Firstly, if $a \in ES(b,E')$ then $a \in (ES(b,E') - ES(b,E))$.

Then the rank of $a \vee b$ becomes 10, so $\mathbf{rank}(\Gamma_{E'}, a) = \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Secondly, if $a \notin ES(b,E')$, then $a \notin (ES(b,E) - ES(b,E'))$.

Therefore the ranks of a and $a \vee b$ are unchanged.

Hence, from the inductive hypothesis, $\mathbf{rank}(\Gamma_{E'}, a) < \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Case (2): Contraction

Part (i): Suppose $a \in ES(b,E)$.

Then from the inductive hypothesis we have $\mathbf{rank}(\Gamma_{E}, a) = \mathbf{rank}(\Gamma_{E}, a \vee b)$.

Suppose $x \notin \cup FB(b, E)$.

Then $FB(b,E) = FB(b,E')$, and so $ES(b,E) = ES(b,E')$, and $a \in ES(b,E')$.

Also, the ranks of a and $a \vee b$ are unchanged.

Hence $\mathbf{rank}(\Gamma_{E'}, a) = \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Otherwise, $x \in \bigcup FB(b, E)$.

Then we must consider two possibilities:

Firstly, if $a \in ES(b,E')$ then $a \notin (ES(b,E) - ES(b,E'))$.

Then the ranks of a and $a \vee b$ are unchanged.

Hence, from the inductive hypothesis, $\mathbf{rank}(\Gamma_{E'}, a) = \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Secondly, if $a \notin ES(b,E')$, then $a \in (ES(b,E) - ES(b,E'))$.

Then $a \vee b$ is given rank 20, so that $\mathbf{rank}(\Gamma_{E'}, a) < \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Part (ii): Now consider the case when $a \notin ES(b,E)$.

Then from the inductive hypothesis we have $\mathbf{rank}(\Gamma_E, a) < \mathbf{rank}(\Gamma_E, a \vee b)$.

Suppose $x \notin \bigcup FB(b, E)$.

Then $FB(b,E) = FB(b,E')$, and so $ES(b,E) = ES(b,E')$, and $a \notin ES(b,E')$.

As above, the ranks of a and $a \vee b$ are unchanged.

Hence $\mathbf{rank}(\Gamma_{E'}, a) < \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Otherwise, $x \in \bigcup FB(b, E)$.

Then we must consider two possibilities:

Firstly, if $a \in ES(b,E')$ then $a \in (ES(b,E') - ES(b,E))$

Then the rank of $a \vee b$ is set to 10, giving $\mathbf{rank}(\Gamma_{E'}, a) = \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

Secondly, if $a \notin ES(b,E')$, then $a \notin (ES(b,E) - ES(b,E'))$.

Once again, the ranks of a and $a \vee b$ are unchanged.

Hence, from the inductive hypothesis, $\mathbf{rank}(\Gamma_{E'}, a) < \mathbf{rank}(\Gamma_{E'}, a \vee b)$.

We have shown that $\mathbf{rank}(\Gamma_{E'}, a) = \mathbf{rank}(\Gamma_{E'}, a \vee b)$ if and only if $a \in ES(b,E')$, for all cases, and hence by induction, Lemma E.1 holds for all consistent assumption sets E.

E.2 Lemma

If E is a consistent environment and $a \neq b$ then:

$$a \in ES(b,E) \quad \text{iff} \quad (E \vdash_{ATMS} b) \wedge ((E - \{a\}) \nvdash_{ATMS} b)$$

Proof: Let $E' = E - \{a\}$

Case (1): $a \in ES(b,E)$

Then $FB(b,E) \neq \varnothing$, and hence $E \vdash_{ATMS} b$.

Also, for all $A \in FB(b,E)$, $a \in A$.
Hence for all $A \in FB(b,E)$, $A \notin FB(b,E')$.
Therefore $FB(b,E') = \emptyset$, and $E' \not\models_{ATMS} b$.

Case (2): $a \notin ES(b,E)$
Suppose $E \models_{ATMS} b$.
Then either $b \in E$ or $FB(b,E) \neq \emptyset$.
If $b \in E$ then $E' \models_{ATMS} b$, since $a \neq b$.
Otherwise, choose $A \in FB(b,E)$ such that $a \notin A$, since $a \notin \cap FB(b,E)$.
Then $A \in FB(b,E')$, so $FB(b,E') \neq \emptyset$ and hence $E' \models_{ATMS} b$.

We have shown that if $a \notin ES(b,E)$, then either $E' \models_{ATMS} b$, or else $E \not\models_{ATMS} b$.
Combining with case (1) completes the result.

E.3 Correctness of ATMS_Algorithm_1

To prove the correctness of the algorithm, we must show that the behaviour of the AGM system using the algorithm is equivalent to the behaviour of the ATMS. This will occur if and only if for all consistent environments E, and all $\alpha \in \Sigma$:

$$E \models_{ATMS} \alpha \text{ iff } \overline{\Gamma_1} \vdash \alpha$$

Proof: by induction on the number m of expansion and contraction operations.

Initial case:
Before any operations are performed, $E = \emptyset$.
In this case, $E \models_{ATMS} a$ iff there exists $(B,a) \in J$ such that $E \models_{ATMS} b$, $\forall b \in B$.
Since $E = \emptyset$, the chains of justifications must end in formulae of the form $(\emptyset,x)$.

Case (1): Suppose $E \models_{ATMS} a$.
We show by induction on the length n of the chain of justifications that $\overline{\Gamma_1} \vdash a$.

Initial case: When $n = 1$, we have $(\emptyset,a) \in J$.
Then $a \in \Gamma_{30}$, and hence $\overline{\Gamma_1} \vdash a$.

Inductive step: assume that if $E \models_{ATMS} a$ then $\overline{\Gamma_1} \vdash a$, for all $n \leq k$.
Consider a proof of a with maximum length $k+1$ justifications.
Then $\exists\ (B,a) \in J$ such that $\forall\ b \in B$, $E \models_{ATMS} b$ in at most k steps.

Then by the inductive hypothesis, for each b, $\overline{\Gamma_1} \vdash b$.

Therefore, $\overline{\Gamma_1} \vdash (\bigwedge_{x \in B} x)$.

Also $((\bigwedge_{x \in B} x) \to a) \in \Gamma_{30}$, since $(B,a) \in J$.

Therefore, by modus ponens, $\overline{\Gamma_1} \vdash a$.

We conclude by induction that $\overline{\Gamma_1} \vdash a$, for all n.

Case (2): Conversely, suppose $\overline{\Gamma_1} \vdash a$.
Initially, $\overline{\Gamma_1} = \Gamma_{30} = J \cup N$.
Consider a sequence of resolution steps using these clauses which produces a.
No clause in $J \cup N$ has more than one positive literal, so, in order to resolve to a clause containing a positive literal, each pair of resolving clauses must contain a positive literal.
That is, no clauses from N are used in the proof.

We now show by induction on the number n of resolution steps that $E \models_{ATMS} a$.

Initial case: when n=0, then $(\emptyset,a) \in J$. Hence $E \models_{ATMS} a$.

Inductive step: assume for all $n \leq k$ that $E \models_{ATMS} a$.
Consider a sequence of $k + 1$ resolution steps ending with the clause a.
Then there exists in this sequence a clause containing the positive literal a.
This clause corresponds to some justification $(B,a) \in J$.
Let this clause be $\neg b_1 \vee \cdots \vee \neg b_p \vee a$ (i.e. $B = \{b_1, \ldots, b_p\}$).
Then for $1 \leq i \leq p$, $\overline{\Gamma_1} \vdash b_i$, and the resolution proof for each b_i takes $\leq k$ steps.
So by the inductive hypothesis $E \models_{ATMS} b_i$, for $1 \leq i \leq p$.
But $(\{b_1, \ldots, b_p\},a) \in J$, so by the definition of the ATMS, $E \models_{ATMS} a$.

By induction, we have shown that for all n, if $\overline{\Gamma_1} \vdash a$ then $E \models_{ATMS} a$.
This completes the initial case of the correctness proof.

Inductive step:
Assume that after m expansions and contractions, $E \models_{ATMS} \alpha$ iff $\overline{\Gamma_1} \vdash \alpha$, for all $\alpha \in \Sigma$.
Consider the $(m+1)$st operation.

Case (1): Expansion by a
Let $E' = E \cup \{a\}$.

Let $\Gamma' = \textbf{expand}(\Gamma, a, 10)$.

Part (i): Suppose $E' \models_{ATMS} b$.

Then one of the following 4 cases must hold:

(a) $E \models_{ATMS} b$

(b) $E \not\models_{ATMS} b$ and $\exists\ C \in N$ such that $C \subseteq E'$

(c) $E \not\models_{ATMS} b$ and $a = b$

(d) $E \not\models_{ATMS} b$ and $\exists\ (A,b) \in J$ such that $\forall\ x \in A,\ E' \models_{ATMS} x$

Case (a): By the inductive hypothesis, $\overline{\Gamma_1} \vdash b$.

Using AGM postulate (K^+3), $K \subseteq K_a^+$, so we have $\overline{\Gamma'_1} \vdash b$.

Case (b): $\overline{\Gamma_1} \vdash E$, from the inductive hypothesis, since $\forall\ x \in E,\ E \models_{ATMS} x$.

Also $\overline{\Gamma'_1} \vdash a$, by AGM postulate (K^+1).

Since $K \subseteq K_a^+$ (AGM postulate (K^+3)) and $C \subseteq E'$, we have $\overline{\Gamma'_1} \vdash C$.

But since $C \in N$, $\overline{\Gamma'_1} \vdash (\neg \bigwedge_{x \in C} x)$.

Thus $\textbf{expand}(\Gamma, \alpha, 10)$ is inconsistent, so we can say that b is in the resulting belief state, for any b.

Case (c): By AGM postulate (K^+1), $\overline{\Gamma'_1} \vdash b$.

Case (d): We show that $\overline{\Gamma'_1} \vdash b$ by induction on the number n of justifications used in deriving $E' \models_{ATMS} b$.

Initial case: When $n = 0$, by cases (a) – (c), we have $\overline{\Gamma'_1} \vdash b$.

Inductive step: Assume that for all $n \leq k$ we have $\overline{\Gamma'_1} \vdash b$.

Consider a proof using $k + 1$ justifications, including some $(A,b) \in J$.

Then for all $x \in A,\ E' \models_{ATMS} x$.

Now if case (a), (b) or (c) applies, we have shown $\overline{\Gamma'_1} \vdash x$.

Alternatively, if case (d) applies, the proof of x contains $\leq k$ justifications.

Hence by the inductive hypothesis $\overline{\Gamma'_1} \vdash x$.

Also, since $(A,b) \in J$, $((\bigwedge_{x \in A} x) \to b) \in \overline{\Gamma'_{30}}$.

Therefore we have $\overline{\Gamma'_1} \vdash b$.

Hence, by induction $\overline{\Gamma'_1} \vdash b$, for all n.

Part (ii): Now suppose that $\overline{\Gamma'_1} \vdash b$.

That is, $\overline{\Gamma_1} \cup \{a\} \vdash b$.

We show by induction on the number n of resolution steps used that $E' \models_{ATMS} b$.

Initial case: When $n = 0$, either $\overline{\Gamma_1} \vdash b$ or $b = a$.

If $\overline{\Gamma_1} \vdash b$, then by the inductive hypothesis of the main proof, $E \models_{ATMS} b$.

Also since $E \subseteq E'$, we have $E' \models_{ATMS} b$.

Otherwise, if $b = a$, then $b \in E'$ and thus $E' \models_{ATMS} b$.

Inductive step: Assume for all $n \leq k$ that $E' \models_{ATMS} b$.

Now consider a resolution proof of b requiring $k + 1$ steps.

This proof contains a clause $\neg r_1 \vee \cdots \vee \neg r_p \vee b$, where $(\{r_1, \ldots, r_p\}, b) \in J$.

Also, $\overline{\Gamma'_1} \vdash r_i$, $1 \leq i \leq p$, where each resolution proof requires $\leq k$ steps.

Hence by the inductive hypothesis $E' \models_{ATMS} r_i$, $1 \leq i \leq p$.

Also $(\{r_1, \ldots, r_p\}, b) \in J$, so by the definition of the ATMS, $E' \models_{ATMS} b$.

Hence for all n, $E' \models_{ATMS} b$, and thus we have shown that $E' \models_{ATMS} b$ iff $\overline{\Gamma'_1} \vdash b$.

Case (2): Contraction by a

Let $E' = E - \{a\}$.

Let $\Gamma' = \textbf{contract}(\Gamma, a)$.

Suppose $E' \models_{ATMS} b$.

Then $E \models_{ATMS} b$, since $E' \subseteq E$, and hence by the inductive hypothesis, $\overline{\Gamma_1} \vdash b$.

Then by Lemma E.2, $a \notin ES(b,E)$.

From (EE2) we have $\textbf{rank}(\Gamma, a) \leq \textbf{rank}(\Gamma, a \vee b)$.

Using Lemma E.1, this result strengthens to $\textbf{rank}(\Gamma, a) < \textbf{rank}(\Gamma, a \vee b)$.

Finally, from the definition of contraction, we have $\overline{\Gamma'_1} \vdash b$.

Conversely, suppose that $\overline{\Gamma'_1} \vdash b$.

Then by the definition of contraction, $\textbf{rank}(\Gamma, a) < \textbf{rank}(\Gamma, a \vee b)$.

By Lemma E.1, $a \notin ES(b,E)$.

We also know that $\overline{\Gamma_1} \vdash b$, since $\overline{\Gamma_1} \vdash \overline{\Gamma'_1}$.

So by the inductive hypothesis, $E \models_{ATMS} b$.

Then by Lemma E.2 we conclude that $E' \models_{ATMS} b$.

E.3

Thus after the $(m+1)$st operation, $\forall\alpha \in \Sigma$, we have $E \models_{ATMS} \alpha$ if and only if $\overline{\Gamma_1} \vdash \alpha$.

Conclusion:

Therefore, we have shown by induction that $E \models_{ATMS} \alpha$ iff $\overline{\Gamma_1} \vdash \alpha$, $\forall\alpha \in \Sigma$, and the algorithm correctly maintains the entrenchment relation so that the behaviour of the AGM system is equivalent to that of the ATMS.

F The Correctness of ATMS_Algorithm_2

We show that this algorithm is correct, by demonstrating that the most conservative entrenchment generated from the operations produced by the algorithm is identical to the most conservative entrenchment relation generated by **ATMS_Algorithm_1**. Several lemmas will be proved first, from which the most conservative entrenchment can be computed.

The notation used for these lemmas will be the same for each. Let E represent the ATMS environment, J the justifications, N the nogoods, and D the disjunctions defined by: $D = \{\alpha \vee \beta : \alpha, \beta \in E \text{ and } \alpha \neq \beta\}$. In this section we use the notation for E, J and N very loosely; they refer to both the set-theoretic definitions of the ATMS environments, justifications and nogoods, as well as the logical form of these sets. The correct interpretation will be obvious from the context.

F.1 Lemma

For any $\alpha \in E^*$, $E \cup N \vdash \alpha$ if and only if either $\alpha \in E$ or $E \cup N \vdash \perp$.

Proof:
Suppose $E \cup N \vdash \alpha$.
Suppose also that $\alpha \notin E$ and $E \cup N \nvdash \perp$.
Then the positive literal α does not occur in $E \cup N$, so $E \cup N \cup \{\neg\alpha\}$ is satisfiable, contradicting the first assumption.
Therefore either $\alpha \in E$ or $E \cup N \vdash \perp$.

Alternatively, suppose that either $\alpha \in E$ or $E \cup N \vdash \perp$.
Trivially for either case $E \cup N \vdash \alpha$.

F.2 Lemma

For any $\alpha \in E^*$, if $E \cup N \cup J \cup D \vdash \alpha$ and $E \cup N \nvdash \bot$, then $E \cup J \vdash \alpha$.

Proof:

Firstly, note that since $E \vdash D$ then $E \cup N \cup J \cup D \vdash \alpha$ if and only if $E \cup N \cup J \vdash \alpha$.

Then by the definition of ATMS, if E is consistent ($E \cup N \nvdash \bot$) then $E \cup N \cup J \vdash \alpha$ if and only if $E \cup J \vdash \alpha$.

F.3 Lemma

For any $\alpha, \beta \in \Sigma$, $E \cup N \cup J \vdash \alpha \vee \beta$ if and only if either $E \cup N \cup J \vdash \alpha$ or $E \cup N \cup J \vdash \beta$.

Proof:

Suppose $E \cup N \cup J \vdash \alpha \vee \beta$.

Suppose also that $E \cup N \cup J \nvdash \alpha$ and $E \cup N \cup J \nvdash \beta$.

Let $N' = N \cup \{\neg\alpha\}$.

Then $E \cup N' \cup J \vdash \beta$ and $E \cup N' \cup J \nvdash \bot$.

Thus $E \cup N' \nvdash \bot$, so by Lemma F.2 we have $E \cup J \vdash \beta$.

Hence $E \cup N \cup J \vdash \beta$, contradicting the second supposition.

Therefore either $E \cup N \cup J \vdash \alpha$ or $E \cup N \cup J \vdash \beta$.

Alternatively, if either $E \cup N \cup J \vdash \alpha$ or $E \cup N \cup J \vdash \beta$, then trivially $E \cup N \cup J \vdash \alpha \vee \beta$.

F.4 Lemma

For any $\alpha \in E^*$, $E \cup N \vdash \neg\alpha$ if and only if $\exists C \in N$ such that $C \subseteq E \cup \{\alpha\}$.

Proof:

Suppose $E \cup N \vdash \neg\alpha$.

Then since E consists entirely of positive literals, and all of the clauses in N contain entirely negative literals, there exists a clause $C = \neg l_1 \vee \neg l_2 \vee \cdots \vee \neg l_n \in N$ such that $l_i = \alpha$ for some i.

For all $j \neq i$, $\neg l_j$ must resolve against a literal in E, that is $l_j \in E$.

Thus $C \subset E \cup \{\alpha\}$.

Conversely, suppose for some $C \in N$, $C \subseteq E \cup \{\alpha\}$.
Resolving the clause C against the corresponding literals in E, we are left with either the empty clause, in which case $E \cup N \vdash \perp$ and trivially $E \cup N \vdash \neg\alpha$; otherwise $\neg\alpha$ appears in the clause, and then $\neg\alpha$ is the only literal remaining after resolution, and thus we have shown that $E \cup N \vdash \neg\alpha$.

F.5 Lemma

For any $\alpha \in E^*$, $E \cup N \cup J \cup D \vdash \neg\alpha$ if and only if $E \cup N \vdash \neg\alpha$.

Proof:
Suppose that $E \cup N \cup J \cup D \vdash \neg\alpha$.
Once again, since $E \vdash D$, then D is irrelevant to the proof.
Choose any minimal subset of $J' \subseteq J$ such that $E \cup N \cup J' \vdash \neg\alpha$.
Suppose J' has at least one element, $\Phi_0 \rightarrow r_0$.
Let $J_0 = J' - \{\Phi_0 \rightarrow r_0\}$.
Then $E \cup N \cup J_0 \nvdash \neg\alpha$.
Also $E \cup N \cup J_0 \cup \{\Phi_0 \rightarrow r_0\} \vdash \neg\alpha$.
By the deduction theorem, $E \cup N \cup J_0 \vdash (\Phi_0 \rightarrow r_0) \rightarrow \neg\alpha$.
Therefore $E \cup N \cup J_0 \vdash \alpha \rightarrow \neg(\Phi_0 \rightarrow r_0)$.
Thus $E \cup N \cup J_0 \cup \{\alpha\} \vdash \neg(\Phi_0 \rightarrow r_0)$.
That is, $E \cup N \cup J_0 \cup \{\alpha\} \vdash \neg r_0$.
Let $E_1 = E \cup \{\alpha\}$.
Repeating the above steps, we have: $E_1 \cup N \cup J_1 \cup \{r_0\} \vdash \neg r_1$, for some $J_1 \subset J_0$.
After a finite number of steps n (since J is finite), $J_n = \emptyset$.
Then $E_{n+1} \cup N \vdash \neg r_n$.
By Lemma F.4, there exists some $C \in N$ such that $C \subseteq E_{n+1} \cup \{r_n\}$.
Thus $E_{n+1} \cup \{r_n\} \subseteq E^*$.
But if $(\Phi_i \rightarrow r_i) \in J$ then by definition $r_i \notin E^*$.
Hence we have a contradiction, so $J' = \emptyset$.
Therefore $E \cup N \vdash \neg\alpha$, as required.

F.6 Lemma

After any number n of consistent expansion and contraction operations generated by **ATMS_Algorithm_2**, the entrenchment base Γ is given by: $\Gamma = \{\Gamma_{10}, \Gamma_{20}, \Gamma_{30}\}$, where $\Gamma_{10} = E$, $\Gamma_{20} = D$ and $\Gamma_{30} = J \cup N$.

Proof:

This proof will be completed by induction on n.

When $n = 0$, $E = \emptyset$, $\Gamma_{30} = J \cup N$, $\Gamma_{20} = \emptyset$ and $\Gamma_{10} = \emptyset$ as required.

Now assume that after m operations, $\Gamma_{10} = E$, $\Gamma_{20} = D$ and $\Gamma_{30} = J \cup N$.

Suppose the $(m+1)$ st operation is the expansion $\Gamma' = \mathbf{expand}(\Gamma, \alpha, 10)$.

Then let $E' = E \cup \{\alpha\}$.

Now $\alpha \notin E$, and α cannot be justified (since $\alpha \in E^*$), so we have $\mathbf{rank}(\Gamma, \alpha) = 0$.

Now if $\mathbf{rank}(\Gamma, \neg\alpha) > 0$ then, by Lemma F.5, E' is inconsistent, and Γ' will also be inconsistent.

Otherwise $\Gamma'_{10} = \Gamma_{10} \cup \{\alpha\}$.

Then $\overline{\Gamma'_{10}}$ is tested for any redundancy.

No formula with rank 10 is redundant, since assumptions cannot be justified.

Also, formulae with ranks greater than 10 are unaffected.

The following step performs the expansions by $\alpha\vee\phi$, where $\phi \in E$.

By Lemma F.4, $E \cup J \cup N \vdash \alpha\vee\phi$ iff $E \cup J \cup N \vdash \alpha$ or $E \cup J \cup N \vdash \phi$.

By Lemma F.2, this gives $E \cup J \vdash \alpha$ or $E \cup J \vdash \phi$.

But assumptions cannot be justified, so we have $\phi \in E$, since we know $\alpha \notin E$.

Also $\alpha\vee\phi \notin D$, so $\mathbf{rank}(\Gamma, \alpha\vee\phi) = 10$.

So the expansion operations conclude with:

$$\begin{aligned}\Gamma'_{20} &= \Gamma_{20} \cup \{\alpha\vee\phi : \phi \in E\}\\ &= \{\phi\vee\psi : \phi, \psi \in E' \text{ and } \phi \neq \psi\}\\ &= D'\end{aligned}$$

Also, $\Gamma'_{30} = \Gamma_{30}$.

Therefore Γ' represents the correct entrenchment relation after $m+1$ operations.

Suppose the $(m+1)$ st operation is the contraction $\Gamma' = \mathbf{contract}(\Gamma, \alpha)$.

Then let $E' = E - \{\alpha\}$.

Since $\mathbf{rank}(\Gamma, \alpha) = 10$, and by the definition of D, $\mathbf{rank}(\Gamma, \alpha\vee\beta) = 20$ for all $\beta \in \Gamma_{10}$, we have $\Gamma'_{10} = \Gamma_{10} - \{\alpha\} = E'$.

Also $\Gamma'_{20} = \Gamma_{20} - \{\alpha\vee\phi : \phi \in E'\}$

$= \{\phi\vee\psi : \phi,\ \psi \in E' \text{ and } \phi \neq \psi\}$

$= D'$

Also, $\Gamma'_{30} = \Gamma_{30}$.

Therefore Γ' represents the correct entrenchment relation after $m+1$ operations.

Hence the induction is complete, and the result holds for all n.

F.7 Correctness of the Algorithm

Now since we have shown the correctness of **ATMS_Algorithm_1**, it is easiest to prove the correctness of the second algorithm by showing it is equivalent to the first. Since for all $\alpha \in E$, we have **rank**$(\Gamma, \alpha) = 10$, then the algorithms are equivalent if and only if the following condition is satisfied (compare with Lemma E.1):

For any environment E, for all $\alpha \in E$, and for all β such that $E \models_{ATMS} \beta$:

$$\mathbf{rank}(\Gamma_E,\ \alpha\vee\beta) = \begin{cases} 10, & \text{if } \alpha \in ES(\beta, E) \\ x \geq 20, & \text{otherwise} \end{cases}$$

Proof:

Suppose $\alpha \in ES(\beta, E)$.

Then $\overline{\Gamma_{10}} - \{\alpha\} \nvdash \beta$.

Also $\overline{\Gamma_{10}} \vdash \beta$.

Therefore $\overline{\Gamma_{10}} - \{\alpha\} \cup \{\alpha\} \vdash \beta$.

By the deduction theorem, $\overline{\Gamma_{10}} - \{\alpha\} \vdash \alpha \rightarrow \beta$.

Suppose $\overline{\Gamma_{10}} - \{\alpha\} \vdash \alpha \vee \beta$.

Resolving with the previous line gives $\overline{\Gamma_{10}} - \{\alpha\} \vdash \beta$, a contradiction.

Hence $\overline{\Gamma_{10}} - \{\alpha\} \nvdash \alpha \vee \beta$.

Thus **rank**$(\Gamma, \alpha\vee\beta) = 10$.

Conversely, suppose that $\alpha \notin ES(\beta, E)$.

Then $\overline{\Gamma_{10}} - \{\alpha\} \vdash \beta$.

Also there exists some $\Phi \in FB(\beta, E)$ such that $\alpha \notin \Phi$.

Then $\Phi \subseteq E$, so $\alpha\vee\phi \in D$ for all $\phi \in \Phi$.

Therefore $D \cup \{\neg\alpha\} \vdash \Phi$.

Since $\Phi \cup J \vdash \beta$, we have $D \cup \{\neg\alpha\} \cup J \vdash \beta$.
By the deduction theorem, $D \cup J \vdash \alpha\vee\beta$.
Thus **rank**$(\Gamma, \alpha\vee\beta) \geq 20$.

Thus we have shown that the generated entrenchment satisfies Lemma E.1, and hence by Correctness Proof E.3, we conclude that **ATMS_Algorithm_2** is also correct under the same conditions.

G References

[Alchourrón *et al.* 1985] Alchourrón, C.E., Gärdenfors, P. & Makinson, D. (1985) On the Logic of Theory Change: Partial Meet Contraction and Revision Functions, *Journal of Symbolic Logic*, **50**, pp. 510-530.

[Alchourrón & Makinson 1982] Alchourrón, C.E. & Makinson, D. (1982) The Logic of Theory Change: Contraction Functions and Their Associated Revision Functions, *Theoria*, **48**, pp. 14-37.

[Alchourrón & Makinson 1985] Alchourrón, C.E. & Makinson, D. (1985) On the Logic of Theory Change: Safe Contraction, *Studia Logica*, **44**, pp. 405-422.

[Boutillier 1993] Boutillier, C. (1993) Revision Sequences and Nested Conditionals, *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pp. 519-525.

[Brewka 1989] Brewka, G. (1989) Preferred Subtheories – An Extended Logical Framework for Default Reasoning, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 1043-1048, Morgan Kaufmann.

[Brewka 1991a] Brewka, G. (1991) Belief Revision in a Framework for Default Reasoning, in *The Logic of Theory Change*, Fuhrmann, A. & Morreau, M. (Eds), pp. 206-222, Springer-Verlag, Berlin.

[Brewka 1991b] Brewka, G. (1991) *Nonmonotonic Reasoning: Foundations of Common Sense*, Cambridge University Press, Cambridge.

[Chang & Lee 1973] Chang, C.L. & Lee, R.C.T. (1973) *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, New York, NY.

[Cornman 1978] Cornman, J.W. (1978) Foundational versus Nonfoundational Theories of Empirical Justification, in *Essays on Knowledge and Justification*, Pappas, G.S. & Swain, M. (Eds), pp. 229-278, Cornell University Press, Ithica, NY.

[Dalal 1988] Dalal, M. (1988) Investigations into a Theory of Knowledge Base Revision: Preliminary Report, *Proceedings of the Seventh National Conference on Artificial Intelligence*, pp. 475-479, Morgan Kaufmann.

[de Kleer 1986] de Kleer, J. (1986) An Assumption-Based TMS, *Artificial Intelligence*, **28**, pp. 127-162.

[Dixon 1989] Dixon, S.E. (1989) Implementation and Comparison of ATMS and AGM, Computer Science Honours Thesis, University of Sydney, NSW 2006, Australia.

[Dixon 1993] Dixon, S.E. (1993) A Finite Base Belief Revision System, *Proceedings of the Sixteenth Australian Computer Science Conference*, pp. 445-451.

[Dixon & Foo 1992a] Dixon, S.E. & Foo, N.Y. (1992) Encoding the ATMS in AGM Logic (Revised), Computer Science Technical Report 441, University of Sydney, NSW 2006, Australia.

[Dixon & Foo 1992b] Dixon, S.E. & Foo, N.Y. (1992) Beyond Foundational Reasoning, *Proceedings of the Fifth Australian Joint Conference on Artificial Intelligence*, pp. 247-252, World Scientific.

[Dixon & Foo 1993] Dixon, S.E. & Foo, N.Y. (1993) Connections Between the ATMS and AGM Belief Revision, *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pp. 534-539.

[Dixon & Wobcke 1993] Dixon, S.E. & Wobcke, W.R. (1993) The Implementation of a First-Order Logic AGM Belief Revision System, *Proceedings of the Fifth IEEE International Conference on Tools with AI*, pp. 40-47.

[Dixon & Wobcke 1994] Dixon, S.E. & Wobcke, W.R. (1993) A Nonmonotonic Reasoning Belief Revision System, to appear.

[Doyle 1979] Doyle, J. (1979) A Truth Maintenance System, *Artificial Intelligence*, **12**, pp. 231-272.

[Fagin *et al.* 1983] Fagin, R., Ullman, J.D. & Vardi, M.Y. (1983) On the Semantics of Updates in Databases, *Proceedings of the Second ACM SIGACT-SIGMOD*

Symposium on Principles of Database Systems, pp. 352-365.

[Fagin *et al.* 1986] Fagin, R., Kuper, G.M., Ullman, J.D. & Vardi, M.Y. (1986) Updating Logical Databases, *Advances in Computing Research*, **3**, pp. 1-18, JAI Press.

[Fuhrmann 1991] Fuhrmann, A. (1991) Theory Contraction Through Base Contraction, *Journal of Philosophical Logic*, **20**, pp. 175-203.

[Gärdenfors 1988] Gärdenfors, P. (1988) *Knowledge in Flux: Modeling the Dynamics of Epistemic States*, Bradford Books, MIT Press, Cambridge, MA.

[Gärdenfors 1990a] Gärdenfors, P. (1990) Belief Revision and Nonmonotonic Logic: Two Sides of the Same Coin?, *Proceedings of the Ninth European Conference on Artificial Intelligence*, pp. 768-773.

[Gärdenfors 1990b] Gärdenfors, P. (1990) The Dynamics of Belief Systems: Foundations vs Coherence Theories, *Revue Internationale de Philosophie*, **1**, 172, pp. 24-46.

[Gärdenfors 1991] Gärdenfors, P. (1991) Nonmonotonic Inferences Based on Expectations: A Preliminary Report, *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, pp. 585-590.

[Gärdenfors & Makinson 1988] Gärdenfors, P. & Makinson, D. (1988) Revisions of Knowledge Systems Using Epistemic Entrenchment, *Proceedings of the Second Conference on Theoretical Aspects of Reasoning About Knowledge*, pp. 83-95.

[Geffner & Pearl 1992] Geffner H. & Pearl, J. (1992) Conditional Entailment: Bridging Two Approaches to Default Reasoning, *Artificial Intelligence*, **53**, pp. 209-244.

[Genesereth & Nilsson 1987] Genesereth, M.R. & Nilsson, N.J. (1987) *Logical Foundations of Artificial Intelligence*, Morgan Kaufmann.

[Goldszmidt *et al.* 1990] Goldszmidt, M., Morris, P. & Pearl, J. (1990) A Maximum Entropy Approach to Nonmonotonic Reasoning, *Proceedings of the Eighth National Conference on Artificial Intelligence*, pp. 646-652.

[Grove 1988] Grove, A. (1988) Two Modellings for Theory Change, *Journal of Philosophical Logic*, **17**, pp. 157-170.

[Hansson 1989] Hansson, S.O. (1989) New Operators For Theory Change, *Theoria*, **50**, pp. 114-132.

[Hansson 1991] Hansson, S.O. (1991) Belief Contraction Without Recovery, *Studia Logica*, **50**, pp. 251-260.

[Harman 1986] Harman, G. (1986) *Change In View*, Bradford Books, MIT Press, Cambridge, MA.

[Katsuno & Mendelzon 1989] Katsuno, H. & Mendelzon, A.O. (1989) A Unified View of Propositional Knowledge Base Updates, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 1413-1419, Morgan Kaufmann.

[Kean & Tsiknis 1993] Kean, A. & Tsiknis, G. (1993) Clause Management Systems (CMS), *Computational Intelligence*, **9**, 1, pp. 11-40.

[Lehmann & Magidor 1992] Lehmann, D. & Magidor, M. (1992) What Does a Conditional Knowledge Base Entail?, *Artificial Intelligence*, **55**, pp. 1-60.

[Levesque & Brachman 1985] Levesque, H.J. & Brachman, R.J. (1985) A Fundamental Tradeoff in Knowledge Representation and Reasoning, in *Readings in Knowledge Representation*, pp. 41-70, Morgan Kaufmann, San Mateo, CA.

[Lewis 1973] Lewis, D.K. (1973) *Counterfactuals*, Blackwell, Oxford.

[Lifschitz 1989] Lifschitz, V. (1989) Benchmark Problems for Formal Nonmonotonic Reasoning (Version 2.00), in *Non-Monotonic Reasoning*, Reinfrank, M., de Kleer, J., Ginsberg, M.L. & Sandewall, E. (Eds), pp. 202-219, Springer-Verlag, Berlin.

[Makinson 1985] Makinson, D. (1985) How To Give It Up: A Survey of Some Formal Aspects of the Logic of Theory Change, *Synthese*, **62**, pp. 347-363.

[Makinson 1987] Makinson, D. (1987) On the Status of the Postulate of Recovery in the Logic of Theory Change, *Journal of Philosophical Logic*, **16**, pp. 383-394.

[Makinson & Gärdenfors 1991] Makinson, D. & Gärdenfors, P. (1991) Relations Between the Logic of Theory Change and Nonmonotonic Logic, in *The Logic of Theory Change*, Fuhrmann, A. & Morreau, M. (Eds), pp. 185-205, Springer-Verlag, Berlin.

[Martins 1990] Martins, J.P. (1990) The Truth, the Whole Truth, and Nothing But the Truth: An Indexed Bibliography to the Literature of Truth Maintenance Systems, *AI Magazine (Special Issue)*, pp. 7-25.

[Nebel 1989] Nebel, B. (1989) A Knowledge Level Analysis of Belief Revision, *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pp. 301-311, Morgan Kaufmann, San Mateo, CA.

[Nebel 1990] Nebel, B. (1990) *Reasoning and Revision in Hybrid Representation Systems*, Springer-Verlag, Berlin.

[Nebel 1991] Nebel, B. (1991) Belief Revision and Default Reasoning: Syntax-Based Approaches, *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, pp. 417-428, Morgan Kaufmann, San Mateo, CA.

[Niederée 1991] Niederée, R. (1991) Multiple Contraction: A Further Case Against Gärdenfors' Principle of Recovery, in *The Logic of Theory Change*, Fuhrmann, A. & Morreau, M. (Eds), Springer-Verlag, Berlin.

[Pappas & Swain 1978] *Essays on Knowledge and Justification*, Pappas, G.S. & Swain, M. (Eds), pp. 30-35, 184-308, Cornell University Press, Ithica, NY.

[Pastin 1978] Pastin, M. (1978) Modest Foundationalism and Self-Warrant, in *Essays on Knowledge and Justification*, Pappas, G.S. & Swain, M. (Eds), pp. 279-288, Cornell University Press, Ithica, NY.

[Pearl 1990] Pearl, J. (1990) System Z: A Natural Ordering of Defaults with Tractable Applications to Nonmonotonic Reasoning, *Proceedings of the Third Conference on Theoretical Aspects of Reasoning About Knowledge*, pp. 121-135.

[Peppas *et al.* 1991] Peppas, P., Foo, N.Y. & Wobcke, W.R. (1991) Events as Theory

Operators, *Proceedings of the First World Conference on the Fundamentals of Artificial Intelligence*, pp. 413-426.

[Peppas & Williams 1992] Peppas, P. & Williams, M. (1992) A Unified View of Constructive Modellings for Revision, Presented at: *Second International Symposium on Artificial Intelligence and Mathematics*.

[Poole 1988] Poole, D. (1988) A Logical Framework for Default Reasoning, *Artificial Intelligence*, **36**, pp. 27-47.

[Ramsey 1931] Ramsey, F.P. (1931) General Propositions and Causality, in *Foundations of Mathematics and Other Logical Essays*, pp. 237-257, Routledge and Kegan Paul, New York, NY.

[Reiter 1980] Reiter, R. (1980) A Logic For Default Reasoning, *Artificial Intelligence*, **13**, pp. 81-132.

[Reiter & de Kleer 1987] Reiter, R. & de Kleer, J. (1987) Foundations of Assumption-Based Truth Maintenance Systems: Preliminary Report, *Proceedings of the Fifth National Conference on Artificial Intelligence*, pp. 183-188, Morgan Kaufmann.

[Ross & Anderson 1982] Ross, L. & Anderson, C.A. (1982) Shortcomings in the Attribution Process: On the Origins and Maintenance of Erroneous Social Assessments, in *Judgement Under Certainty: Heuristics and Bias*, Kahneman, D., Slovic, P. & Tversky, A. (Eds), pp. 129-152, Cambridge University Press.

[Rott 1991] Rott, H. (1991) A Nonmonotonic Conditional Logic For Belief Revision I, in *The Logic of Theory Change*, Fuhrmann, A. & Morreau, M. (Eds), Springer-Verlag, Berlin.

[Rott 1992] Rott, H. (1992) On the Logic of Theory Change: More Maps Between Different Kinds of Contraction Functions, in *Belief Revision*, Gärdenfors, P. (Ed.), pp. 122-141, Cambridge University Press.

[Selman & Levesque 1990] Selman, B. & Levesque, H.J. (1990) Abductive and Default Reasoning: A Computational Core, *Proceedings of the Eighth National Conference on Artificial Intelligence*, MIT Press.

[Sosa 1980] Sosa, E. (1980) The Raft and the Pyramid: Coherence versus Foundations in the Theory of Knowledge, *Midwest Studies in Philosophy*, **5**, pp. 3-25.

[Spohn 1988] Spohn, W. (1988) Ordinal Conditional Functions: A Dynamic Theory of Epistemic States, in *Causation in Decision, Belief Change, and Statistics, II*, Harper, W.L. & Skyrms, B. (Eds), Kluwer, Dordrecht.

[Thagard 1989] Thagard, P. (1989) Explanatory Coherence, *Behavioural and Brain Sciences*, **12**, pp. 435-502.

[Ullman 1988] Ullman, J.D. (1988) *Principles of Database and Knowledge-Base Systems*, **1**, Computer Science Press, Rockville, MD.

[Williams 1992] Williams, M. (1992) Two Operators for Theory Base Change, *Proceedings of the Fifth Australian Joint Conference on Artificial Intelligence*, pp. 259-265.

[Williams 1993] Williams, M. (1993) On The Logic of Theory Base Change, Computer Science Technical Report 459, University of Sydney, NSW 2006, Australia.

[Winston 1984] Winston, P.H. (1984) *Artificial Intelligence*, Addison-Wesley.

[Wobcke 1992a] Wobcke, W.R. (1992) On the Use of Epistemic Entrenchment in Nonmonotonic Reasoning, *Proceedings of the Tenth European Conference on Artificial Intelligence*, pp. 324-328.

[Wobcke 1992b] Wobcke, W.R. (1992) A Belief Revision Approach to Nonmonotonic Reasoning, *Proceedings of the Fifth Australian Joint Conference on Artificial Intelligence*, pp. 278-283.