A meta-level argumentation framework for representing and reasoning about disagreement

Mandy Haggith

PhD Thesis
The University of Edinburgh
1996

Declaration

I declare that this thesis has been composed by myself and that the research reported here has been conducted by myself unless otherwise indicated.

Mandy Haggith
Edinburgh, August 7th 1996.

Abstract

The contribution of this thesis is to the field of Artificial Intelligence (AI), specifically to the sub-field called knowledge engineering. Knowledge engineering involves the computer representation and use of the knowledge and opinions of human experts.

In real world controversies, disagreements can be treated as opportunities for exploring the beliefs and reasoning of experts via a process called *argumentation*. The central claim of this thesis is that a formal computer-based framework for argumentation is a useful solution to the problem of representing and reasoning with multiple conflicting viewpoints.

The problem which this thesis addresses is how to represent arguments in domains in which there is controversy and disagreement between many relevant points of view. The reason that this is a problem is that most knowledge based systems are founded in logics, such as first order predicate logic, in which inconsistencies must be eliminated from a theory in order for meaningful inference to be possible from it.

I argue that it is possible to devise an argumentation framework by describing one (FORA : Framework for Opposition and Reasoning about Arguments). FORA contains a language for representing the views of multiple experts who disagree or have differing opinions. FORA also contains a suite of software tools which can facilitate debate, exploration of multiple viewpoints, and construction and revision of knowledge bases which are challenged by opposing opinions or evidence.

A fundamental part of this thesis is the claim that *arguments are meta-level structures* which describe the relationships between statements contained in knowledge bases. It is important to make a clear distinction between representations in knowledge bases (the object-level) and representations of the arguments implicit in knowledge bases (the meta-level). FORA has been developed to make this distinction clear and its main benefit is that the argument representations are independent of the object-level representation language. This is useful because it facilitates integration of arguments from multiple sources using different representation languages, and because it enables knowledge engineering decisions to be made about how to structure arguments and chains of reasoning, independently of object-level representation decisions.

I argue that abstract argument representations are useful because they can facilitate a variety of knowledge engineering tasks. These include knowledge acquisition; automatic abstraction from existing formal knowledge bases; and construction, re-representation, evaluation and criticism of object-level knowledge bases. Examples of software tools contained within FORA are used to illustrate these uses of argumentation structures. The utility of a meta-level framework for argumentation, and FORA in particular, is demonstrated in terms of an important real world controversy concerning the health risks of a group of toxic compounds called aflatoxins.

Contents

Figures

Chapter 3. Overview of FORA

Chapter 4. Knowledge Representation in FORA

Chapter 5. Using FORA

Chapter 6. Automating Mark-up to FORA

"I know what you're thinking about," said Tweedledum :
"but it isn't so, nohow."

"Contrariwise," continued Tweedledee,
"if it was so, it might be; and if it were so, it would be : but as it isn't, it ain't.
That's logic."

Through the Looking Glass. Lewis Carroll. 1887.

Chapter 1

Introduction

The contribution of this thesis is to the field of Artificial Intelligence (AI), specifically to the sub-field called knowledge engineering. Knowledge engineering involves the computer representation and use of the knowledge and opinions of human experts. This document presents a framework called FORA (Framework for Opposition and Reasoning about Arguments) for use in situations when we wish to represent the views of many people who disagree or have differing opinions, and when we wish to use a computer to facilitate debate, exploration of multiple viewpoints, or revision of knowledge bases which are challenged by opposing opinions or evidence.

The conclusion of this thesis is that *arguments are meta-level structures*, and I argue that it is useful to represent them as such in a computer-based framework. I show that this is *possible* by presenting an argumentation framework, called FORA, and I argue that it is *useful* by demonstrating that FORA can facilitate a variety of knowledge engineering tasks.

1.1 Motivation, context, and background

Many knowledge bases in knowledge based systems only include representations of knowledge from one person, usually an expert in a domain such as medicine, law or chemistry. When knowledge bases need to represent multiple experts' knowledge, the knowledge is almost always constrained to be consistent. Techniques such as the Delphi Method ([Delbecq *et al* 75], [Ng 90]) are used during knowledge acquisition, to iron-out inconsistencies and disagreements between the experts, but only the resulting consensus viewpoint is explicitly represented in the knowledge base to be used for inference. *The over-riding tendency in most AI and knowledge engineering methodologies is to see conflict as a problem and try to resolve or remove it.*

The aim of this thesis is to explore whether we can instead view conflict as an *opportunity* for interesting reasoning. In the real world, absolute consensus or consistency is very rare. People have their own opinions and differing viewpoints, and as a result people commonly disagree with one another. A great deal of intelligent activity - discussions and debates, learning and teaching, legal trials, democratic decision making - could not occur if the people involved did not have differing viewpoints.

Some good examples of the importance of multiple viewpoints can be found in natural resource management. When decisions have to be made which involve changes to natural resources such as oceans, forests or the atmosphere, the interests of various parties need to be weighed up. These parties include land-owners, environmental pressure groups, wildlife biologists, governments and industries. Industries are currently under increasing pressure to carry out environmental impact assessments of their proposed development plans, and to integrate their results into their decision making processes. However, ecosystems are highly complex phenomena and often a range of relevant specialists will produce conflicting predictions of the effects of environmental changes (the controversy surrounding the greenhouse theory of global warming is a particularly interesting example, see section 4.3 and [Kellogg 91]). In addition, conflicts can arise due to differing interests or goals, for example, conservation of particular wildlife habitats versus economic development (see section 4.2, for an example of a conflict of interests surrounding oil industry development in Alaska). Conflicts can also result from the use of different terminologies by people from different disciplines - such conflicts are sometimes hard to untangle as both agreement and disagreement can be obscured by differing language use (see for example [Shaw & Gaines 91]).

Similarly complex examples of multiple viewpoints exist in the domain of health risk assessment, in which the views of policy makers, food and drug administrations, food and drug producers and suppliers, epidemiologists, medical experts, and the general public, all need to be taken into account in making risk assessments. I will use an example of such a risk assessment to illustrate many of the ideas described in this document (see section 3.3).

The need to handle multiple viewpoints presents a problem to the developers of knowledge based decision support systems. Single, consistent knowledge bases are incapable of meeting the needs of situations such as these. Instead we require systems containing multiple knowledge bases, each of which can represent a particular viewpoint on a problem. We then need mechanisms for identifying, exploring and evaluating conflicts within and between knowledge bases. The overall aim of the research described in this document is to support the construction of such knowledge based systems.

The users of such systems should be treated less like a novice asking a question of an expert who knows the answer, and more like a responsible decision-maker who calls, and chairs, a meeting of various relevant experts or interested parties who each argue for various positions. A debate ensues in which relevant counter-arguments need to be presented at appropriate times. By interacting with the system in this way, I believe that users would become better informed and more aware of the nature of the choices they have to make in order to make a wise decision. This model of system-

user interaction accords with the ethical need for a human *locus of responsibility* [Whitby 88], that is, a guarantee that computers or expert systems are not in a direct line of responsibility for making decisions which may turn out to be safety-critical or life threatening. In order to construct such systems we need better representational models for arguments, and this document presents one such model.

1.2 Thesis message and contribution

The central claim of this thesis is that it is possible, and useful, to represent knowledge in the form of *arguments* for points of view. Doing so facilitates the handling of disagreements between experts without requiring that the conflicts be resolved.

This document describes a computational framework called FORA for articulating arguments at a high level of abstraction and exploring the resulting structures. Arguments and debate involve reasoning *about* the relationships between the statements and chains of inference we use, and so they involve *meta-level* knowledge. Methods are provided in FORA for constructing such meta-level knowledge bases both from scratch and by automatic abstraction (*mark-up*) from formal object-level theories or knowledge bases. The usefulness of the meta-level argumentation approach is also illustrated by describing how instantiation (*mark-down*) of argument structures can support construction and criticism of object-level knowledge bases.

FORA, and the theory behind it, is a contribution to the field of argumentation. I argue that it provides a more robust and rigorous set of structures for representing and reasoning about arguments than argumentation techniques in current use such as IBIS [Kunz & Rittel 70] [Conklin & Begeman 88]. FORA's argument representation language is more abstract than most formal argumentation approaches such as the logic of argumentation, LA [Krause *et al* 95a]), and I argue that this confers some important benefits for knowledge engineers. The suite of tools described in chapters 6 and 7 provide automated and semi-automated transformations between FORA and other formal languages thus suggesting that the framework is general purpose and can integrate into existing knowledge representation practices. These tools also provide support for maintenance and adaptation of knowledge bases as their contents change or are disputed. This thesis is thus also a contribution to the area of knowledge and requirements engineering.

1.3 Overview

In chapter 2, I survey the related work on argumentation and disagreement and explain the usefulness of a meta-level approach to inconsistency handling.

In chapter 3, the FORA framework is introduced, along with an example of a debate in the domain of health risk assessment which is used to illustrate the techniques presented in the remainder of the document.

In chapter 4, the FORA representation language is defined. It enables arguments from knowledge bases to be reasoned about independently of their object-level representation. Four basic meta-level relations are introduced : *disagreement*, *equivalence*, *justification* and *elaboration*, and *arguments* are also defined. Chapter 4 also addresses the question of how to construct knowledge bases in FORA. Acquisition of knowledge from multiple viewpoints is discussed and two software tools are described which help users to express arguments (*mark them up)* using the formal structures of FORA.

Chapter 5 discusses how representations of arguments in FORA can be used to guide a user in exploring the range of opinion in a collection of knowledge bases. The notion of a *conflict set* is introduced to provide a focussed roving 'window' on the particular disagreements a user is interested in. The four primitive relations are used as the basis of definitions of more complex argument constructs, such as those encountered in the literature on argumentation. These argument constructs provide ways to automate comparison and evaluation of knowledge bases.

In order to assess the usefulness of the framework it is necessary to consider how it relates to other knowledge representation languages in which arguments can be expressed. Chapter 6 addresses how the framework can be used for reasoning about arguments in existing (legacy) knowledge bases, and describes how abstraction or *mark-up* to FORA can be automated from knowledge bases expressed in a formal logic. Two techniques for this are discussed and illustrated.

Chapter 7 tackles the converse issue, which is how to use representations of arguments as a basis for *construction* of knowledge bases in an object-level language. This transformation cannot be automated. However, the meta-level framework can provide guidance to a user who wishes to construct a formal object-level representation. Software tools are described which can ensure that a chain of reasoning is carried out by an object-level inference engine, by helping users to formalise appropriately the reasoning steps represented by an argument in FORA. An inference checker tool helps to verify that this has been successful and a knowledge base critic uses FORA representations to suggest elements at the object-level which are vulnerable to dispute.

Chapter 8 concludes by summarising the contribution of the research described in this thesis, and discussing some promising future directions for research.

Chapter 2
Literature Survey

This chapter provides a survey of literature in the fields of logic, philosophy and Artificial Intelligence (AI) which addresses the question of handling disagreements. It is divided into two themes. First there is the issue of disagreement itself, and section 2.1 surveys the literature about conflict, inconsistency and multiple viewpoints. This section characterises the 'problem'. Secondly, there is the field of argumentation. Section 2.2 describes approaches to argument in philosophy, linguistics and AI. This section describes the background to the 'solution' adopted here.

2.1 Disagreement

The study of disagreement is problematic because so many disciplines have a bearing on the issue. The following survey is therefore eclectic and in many cases gives only an overview of the work described. Full technical details can be found in the cited literature.

2.1.1 Object-level logical approaches to disagreement

2.1.1.1 Classical logic

The notion of disagreement is central to logic, as logic is concerned with the analysis of the soundness or unsoundness of arguments. Within logic the somewhat vague notion of disagreement is replaced by a number of more formal concepts, the two most important being *contradiction* and *inconsistency*. Although these notions are generally accepted as fundamental to logic, there are surprising discrepancies between different logicians' definitions of them.

Definitions

[Lemmon 65] defines a *contradiction* syntactically as 'a conjunction, the second conjunct of which is the negation of the first conjunct'. An *inconsistency* is defined as a semantic notion (*ie:* to do with the assignment of truth or falshood to propositions). A formula is inconsistent 'if it takes the truth-value F for all possible assignments of truth-values to its variables.'

[Hamilton 78] defines a contradiction as a statement which 'takes truth value F under each possible assignment of truth values to the statement variables which occur in it',

which is the same as Lemmon's interpretation of inconsistency. Hamilton reserves the label inconsistent for formal systems in which both a proposition and its negation are theorems.

Confusingly, the definitions in [Hodges 77] are different again - a set of beliefs is inconsistent if they cannot all be true in any possible situation, and a contradiction is defined as an inconsistent belief. This identification of inconsistency and contradiction is repeated in more advanced logic texts such as [Kleene 67] in which 'a formula E is called inconsistent or contradictory ... if it has a solid column of F's in its truth table'.

Hence the central ideas seem to be firstly, declaring that both a proposition (formula) and its negation hold simultaneously, and secondly, being unable to find a proposition true in any possible situation. I will generally refer to the first situation as a contradiction, following the generally accepted form of the *Law of Non-Contradiction* (eg: [Lemmon 65, p50]) which states that 'it cannot be the case that P both holds and does not hold, for any proposition P', (*ie:* $\vdash \neg(P \& \neg P)$). *Contradiction* can be thought of as primarily a syntactic clash, whereas *inconsistency* is a broader notion which applies to whole systems of belief, or theories, in which a contradiction can be generated. Contradictions are the syntactic manifestations of inconsistency of a belief set or knowledge base.

The meaning of inconsistent theories

One thorny issue here concerns the *semantics* of contradiction and inconsistency, in other words, what they mean. or how they should be interpreted. The most common picture here is that *no interpretation* can be given for an inconsistent logical theory. Inconsistent theories are incoherent and *meaningless*. More importantly any two inconsistent theories are *equally* meaningless, and therefore indistinguishable semantically. This picture has a long and noble tradition, being traceable back to Socrates and Plato through much of western philosophy and logic. It is based on a belief that the real world is self-consistent, that the law of non-contradiction applies to it. There have, of course, always been philosophers and logicians who tried to give coherent accounts of inconsistency, and thus there is a parallel tradition of philosophy (from the Greek sceptics leading most notably to Hegel and Marx) in which it is held that the world is *really* inconsistent, that the world contains real counter-examples to the law of non-contradiction, and that we thus need to be able to give meaningful interpretations of inconsistent theories.

The earliest examples of *ontological inconsistencies* as they are sometimes known (see, for example [Rescher & Brandom 79]), came from the early Greek philosopher Heraclitus, and concern movement and action. One of the famous examples is the

'river fragments' quandary, in which when stepping through a river we are both in the same river, and yet not in quite the same river at all because it is continuously changing. The resurgence of philosophical interest in inconsistency in the twentieth century has followed developments in modern physics, notably quantum theory, in which the existence or non-existence of a moving particle at a particular place can be problematic. The philosophical study of language use (particularly from the point of view of Hegel's dialectic or Wittgenstein's language games) also encourages a more open mind towards disagreement than normally demonstrated in classical logic. Wittgenstein, for example, predicted a time when people would be 'proud to have emancipated themselves from consistency', and tells us that 'we shall see a contradiction in a quite different light if we look at its occurrence and its consequences as it were anthropologically' rather than 'if we look at it from the point of view of the mathematical law-giver' [Wittgenstein 67]. Issues from the study of ethics (such as the problem of moral relativism, see [Meiland & Krausz 82]), and the logical paradoxes such as the Liar paradox [Martin 84], add pressure to these questions from science and linguistics, and result in inconsistency being an issue of profound philosophical interest.

A full survey of philosophical discussions of inconsistency throughout history is impossible here, though it is a fascinating subject. Excellent accounts of the history of ideas on inconsistency can be found elsewhere, one particularly good example being [Priest et al 89]. It is enough to note two points, firstly that there is a lack of unambiguous definitions of 'inconsistency' and 'contradiction', and secondly that there is great debate about how we should interpret these terms with respect to the real world. The lack of consensus on these basic issues is somewhat surprising given their importance in the practice of logic, and the confidence with which they are used in it. This ambiguity will not be resolved here and is presented simply to demonstrate the relevance of a systematic study of disagreements to mainstream logic. As further demonstration of this relevance the next section will examine how important contradictions and inconsistencies are in the practical business of logical proof.

The use of disagreement in proofs

There are two main uses of contradictions in logical reasoning. Firstly, there is proof by contradiction, commonly known as *reductio ad absurdum*, and secondly, there is the principle of *ex falso quod libet*, which allows the deduction of any proposition from an inconsistent theory.

In *reductio ad absurdum*, a proposition can be proved to hold by assuming its negation and deriving a contradiction from it. This is the main proof technique of semantic tableaux : in order to prove a proposition P from a set of axioms (a theory), its negation, *not* P, is added to the theory, and inference steps are applied until both a

proposition and its negation are derived, thus forming a contradiction. Assuming that the initial set of axioms cannot generate that contradiction alone, the proposition to blame is therefore the assumed one, *not P*, which therefore cannot hold. Another fundamental assumption of classical logic is now brought into play, which is the *law of the excluded middle (tertium non datur)*, which states that one of *P* or *not P* must hold, in other words, there cannot be propositions which are neither true nor false. In the proof by contradiction, the negated proposition, not P, fails to hold, as it causes contradiction, and therefore we can conclude that P must follow from the axioms.

The principle of *ex falso quod libet* demonstrates most clearly why disagreements are unwelcome in classical logic. It states that *any* proposition follows from the existence of a contradiction in a theory, *ie:* from an inconsistent theory. This is tantamount to logical chaos, because as soon as a contradiction appears in a theory, all possible syntactically correct propositions are immediately derivable thus making it impossible to distinguish between sensible and nonsensical conclusions. *Ex falso quod libet* is simple to prove using the rule of or-introduction and modus ponens.

(1) P	*Premiss 1*
(2) ¬P	*Premiss 2*
(3) ¬P v Q	*v-introduction on (2)*
(4) P→ Q	*rewriting (3)*
(5) Q	*by modus ponens from (1) and (4).*

This proves that from the assumption of both *P* and its negation, any proposition whatever (*Q*) can be derived.

It is in this sense that inconsistency is said to be *fatal* to a logical system. Allowing a proposition to be both true and false is not merely distasteful to logicians, much more significantly it *trivialises* logical deduction in a classical logic system because anything at all can be deduced. If we wish to investigate whether anything positive can be done with contradictions, we will need to find a way around *ex falso quod libet.* In the following sections some possible approaches to this will be examined. This will take us into the realm of what Quine calls Deviant Logics, in discussions of which he claims 'neither party knows what he is talking about' [Quine 70, p81], and most of which he denies are really logics at all. However, there are several coherent accounts of contradictions which successfully evade logical triviality and these are described in the following subsections (2.1.1.2 - 2.1.1.5).

2.1.1.2 Many-Valued and Intuitionistic Logic

One of the ways of rethinking inconsistency is to question the fundamental notions of truth and falsity, and in particular to reject the dichotomy between them, denying that any proposition must be exclusively either true or false. Some important branches of

logic have taken this route, and they are usually characterised as having rejected the law of the excluded middle (*tertium non datur*). Two of these branches are particularly significant.

The first is the development of *many-valued logics*, the most straightforward of which is a three-valued logic, in which a proposition may be either true, false, or something between (unsure, unknown, or possible). It is straightforward to draw up truth-tables for the standard logical connectives such as negation, conjunction *etc.* which indicate how to combine propositions taking any of the three truth-values. A good example of a three-valued logic which has 'undecided' as its third truth-value can be found in [Kleene 52]. Another famous three-valued logic is Lukasiewicz's attempt to allow statements about the future to remain contingent [Lukasiewicz 1920]. [Haack 78] provides a comprehensive survey of many-valued logics and their philosophical basis, and their computational tractability is discussed in [Turner 84].

The second branch of logic renowned for its rejection of the law of the excluded middle is intuitionism. The basis of intuitionism is a belief that there is no platonic realm of mathematical objects independent of us, about which mathematicians make discoveries, but rather that mathematics is fundamentally dependent on human thought. Mathematical objects cannot therefore be merely assumed to exist, they must be *demonstrated* to exist or *constructed* with various named properties. Proofs must also be *constructive*, in the sense that they cannot make references to mathematical objects which cannot be constructed. This means that certain classical set theoretic proofs (for example concerning infinite sets) are intuitionistically unacceptable. If there are no constructive proofs for a proposition or its negation then that proposition cannot be asserted to be either true or false, thus breaking the law of the excluded middle.

In intuitionistic logic, propositions are assigned meaning, not by truth-functional Tarskian semantics, but by explaining *how* they are to be proved or demonstrated. In particular, the logical connectives are not given truth-functional definitions, so for example, a disjunction $P \vee Q$ holds only if one of P and Q is actually verifiable. Thus for a proposition P for which no proof can be given for either it or its negation, $(P \vee \neg P)$ does not hold. Negation is likewise defined in terms of proof (derivation), as exemplified by the following statement from [Heyting 31] :

> *'The proposition 'C is not rational' signifies the expectation that one can derive a contradiction from the assumption that C is rational. It is important to note that the negation of a proposition always refers to a proof procedure which leads to the contradiction.'*

A full exposition of intuitionistic logic can be found in [Dummet 77]. The importance of constructive logic to computing, via type-theory, is presented in [Martin-Löf 82].

Many-valued logics and intuitionistic logic both result from the rejection of a straightforward dichotomy between truth and falsehood. Although Quine complains that 'it is hard to face up to the rejection of anything so basic', it is clear that doing so opens up some rich avenues of logical exploration with important applications in computer science and artificial intelligence. Before moving on to other more direct approaches to the issue of inconsistency and contradiction, it is worth summing up the reasons why the rejection of the truth falsity dichotomy is so attractive. A more lengthy discussion of this issue can be found in [Quine 70].

1. The world is not always clear-cut, issues are often not black or white, there are gradations in the world to which truth and falsity do not do justice.

2. Often our knowledge of the world is inexact or uncertain, and perhaps logic should reflect that.

3. The truth-falsity dichotomy leads us into Russellian set theoretic paradoxes.

4. Modern science, and in particular quantum physics, requires logical techniques which can handle multiple inconsistent possibilities.

5. There are mathematical assertions which can be neither properly proved nor disproved. (This is the intuitionistic argument.)

Finally, while on the topic of many-valued logics, it is worth pointing out an important area of work which is being deliberately omitted from this survey - that of logics of uncertainty. The discussion above of many-valued logics was largely limited to three-valued logics, but it is quite possible to have many more values, so, for example, a seven-valued logic might have 'truth-values' of definitely true, probably true, possibly true, unknown, possibly false, probably false and definitely false. We can see from this that the study of many-valued logics borders on the study of uncertainty and that the substitution of numbers for truth values leads into the study of probability. However the intention here is to focus on the issue of disagreement and inconsistency, rather than clouding the issue with certainty factors and levels of doubt. The work on uncertainty of most relevance to this thesis involves the use of argumentation to handle uncertainty, for details of which see section 2.2.2.

2.1.1.3 A Logic of Inconsistency

Some logicians have tackled the question of inconsistency directly, and in this section one of the classic examples of this work will be discussed, namely Rescher and Brandom's presentation of a logic which is deliberately permissive of inconsistency [Rescher & Brandom 79]. We have seen that intuitionistic logics reject the law of the

excluded middle, and Rescher and Brandom take the step of also rejecting the law of non-contradiction, hence allowing both the situation when a proposition is *neither* true nor false, and the situation when it is *both* true and false.

They start from the question 'What if the world is not ontologically consistent?' by which they mean that a proposition about the world may be shown to hold, whilst also being shown not to hold (or in their terminology, that a fact both obtains and does not obtain). They are not suggesting here that we may be reasoning inconsistently about the world, nor merely holding inconsistent beliefs (which may prove to be incorrect) but that the world may actually be inconsistent and thus it may be impossible to provide an ontology of the world which can be consistently described by a system of classical logic.

Although their work is presented on the basis of ontological inconsistency as a purely philosophical supposition or hypothetical, they do provide some sketchy examples that might lend support to the suggestion. One of these corresponds to the fourth reason for rejection of the truth-falsity dichotomy, as given at the end of the last section, namely that quantum physics is required to deal with indeterminism and the possibility of inconsistent facts obtaining. Another example is that two computer-based information systems which monitor and take measurements in the world may simultaneously gather inconsistent information, whilst also remaining mostly consistent in their coverage. In such situations we do not want *ex falso quod libet* to nullify all inference on the consistent subsets of information. They deny that inconsistency should be treated as automatically 'infectious' and assert that we can abolish *ex falso quod libet*, thus eradicating the means of its transmission.

A fundamental idea in their logic is that of *overlapping* or overlaying descriptions of the world, theories, or information systems (in AI-speak we can talk of overlapping knowledge bases). By taking the *intersection* of two consistent sets of sentences we can find ourselves in a situation where some questions can no longer be answered. If one set contains 'It is raining' and the other 'It is not raining', we no longer know from the intersection whether or not it is raining. They describe this as the world being *under-determined*. On the other hand if we take the *union* of the sets some questions have more than one answer, so we can for example claim that it is both raining and not raining. The world this time is *over-determined*.

If the sentence 'Grass is a plant' is in both sets of sentences, in both the intersection and union of the sentences this sentence remains reliable, and they then claim that the inconsistency in the over-determined world description is *local* and does not propagate doubt to the remaining sentences. Using an analogy from physics they describe the existence of *P* and *not P* in a theory as a *singularity of over-determinism* and the absence of either as a *singularity of under-determinism*. Their rejection of *ex falso*

quod libet is expressed neatly in these terms : 'It is a key thesis of our present analysis that semantical singularity can be a *local* phenomenon that does not invariably have *global* ramifications - that the occurrence of a singularity does NOT entail its recurrence everywhere.' The proof of this thesis can be found in [Rescher & Brandom 80, pp21-22].

They provide both a proof theory and a model theory for their logic. The proof theory deserves a closer examination, as some of the ideas in it are instructive in how to selectively dismantle the reasoning framework of classical logic in order to broaden its applicability.

The logic of inconsistency differs from classical first-order predicate logic with respect to the interpretation of conjunction and what they call the 'Fundamental Rule of Valid Inference' (interestingly, negation remains unchanged). Rescher and Brandom make an important distinction between *conjunction* (the 'and' of classical logic, written $P\&Q$), which indicates that both P and Q hold in the same world description, and *juxtaposition*, which is a weaker notion (written P,Q) indicating that P and Q hold independently, usually in different world descriptions. Conjunction is thus a strong notion, that two statements are claimed to be true together, at the same time, or in the same place, or according to one source. Juxtaposition is the weaker notion that two statements are both claimed to be true, but that this is the result of two separate claims, not a single claim of joint truth as needed for conjunction to apply. From a claim P and a separate claim Q we can deduce P,Q but we cannot in general deduce $P\&Q$.

In overlaying (which they call *superposing)* two world descriptions they reject the notion that all resulting descriptions are true collectively, and instead view them as being true distributively. P and Q may thus hold in two different world descriptions, and be juxtaposed giving P,Q. This does not entail that they are conjoined, *ie:* that $P\&Q$ is now also true. This interpretation of conjunction leads to a rejection of the Fundamental Rule of Valid Inference (R) , as follows :

> *R : Whenever P1, P2, ...Pn $\vdash$ Q is a valid inference principle of classical logic, and $T_w(P1)$, $T_w(P2)$,....$T_w(Pn)$, then $T_w(Q)$.*
> *($T_w(P)$ indicates that P is true in superposed world description w)*

They replace R with the following :

> *R1 : Whenever P1, P2, ...Pn $\vdash$ Q is a valid inference principle of classical logic, and $T_w(P1 \ \& \ P2 \ \& \ .... \ \& \ Pn)$, then $T_w(Q)$.*

R and $R1$ are meta-principles about the applicability of rules of inference. The significance of $R1$ is that it outlaws the application of *ex falso quod libet* in situations

where P and *not* P are merely juxtaposed, thus if a contradiction arises *between* two sets of sentences, it is not possible to derive any proposition. This is only allowed if the conjunction $P \& \neg P$ is present in one set. (Note that it also limits the applicability of 'and-introduction' to the trivial case). Thus if the sets of sentences are internally consistent any singularities which arise from their superposition will be localised.

One of the interesting aspects of Rescher and Brandom's work is its recognition that it represents a radical shift in what they call the 'ideology' of logic - and in particular a rejection of the notion of a single, objective, consistent view of the world which they call 'the myth of the God's eye view'. Their alternative ideology embraces both the notion that different people have different views of the world, providing a rich tapestry of multiple perspectives, and also that the existence of inconsistencies between these perspectives is a positive thing.

> *'Man's mind does not thrive on consistency alone : the blockage to order that results from conflicting images is a crucial goad to inquiry and a pivotal motive for enlarging our information. Intellectual disequilibrium is a powerful constructive force.'*

In the next section we will examine another logic falling under the same ideology, which has had significant impact on the thinking of logicians who attempt to handle inconsistency in computer systems.

2.1.1.4 Paraconsistent Logic

In [da Costa 74] a theory of inconsistent formal systems is presented as a variant of the classical logic in [Kleene 52]. Da Costa sees the development of a theory of inconsistent calculi as analogous to the development of non-euclidean geometry, and in particular interprets the existence of paradoxes (such as those of set theory) as similar to the points at infinity in euclidean plane geometry. This is a good analogy because his presentation simply denies two basic postulates or axioms of classical logic and investigates what happens from there, in the same way that non-euclidean geometries start from a subset of Euclid's axioms. The two outlawed postulates of logic are the law of non-contradiction and *ex falso quod libet*. The law of excluded middle and the law of double negation remain, as do most of the other basic principles, though some, such as *reductio ad absurdum*, have added premises which check for non-contradiction. To give a flavour of how this is done, here is the statement of *reductio ad absurdum* (where B° stands for $\neg(B \& \neg B)$, *i e:* there is no contradiction about B) :

$$B^\circ \rightarrow ((A \rightarrow B) \rightarrow ((A \rightarrow \neg B) \rightarrow \neg A)))$$

In other words, if B is not contradicted in the system, yet a contradiction about B can

be deduced by assuming A, then deduce $\neg A$.

Da Costa defines a series of logical calculi, progressing from propositional logic to full predicate logic with equality, and finally to a set theory based on a theory of Quine's. In each case he defines an infinite series of calculi. For propositional logic, the base calculus C0 is classical propositional logic, C1 is the restricted inconsistent form with some axioms qualified by a check for non contradiction as in the case of *reductio ad absurdum* above. C2 is then derived by qualifying these axioms with

$$A^\circ \,\&\, A^{\circ\circ}, \text{ ie: } \neg(A \,\&\, \neg A) \,\&\, \neg(\neg(A \,\&\, \neg A) \,\&\, \neg\neg(A \,\&\, \neg A)).$$

This recursive application of non-contradiction tests escalates to infinity, resulting in a series corresponding to the natural numbers of successively stronger and stronger logics. In addition, a form of 'strong' negation is defined (as $\neg A \,\&\, A^\circ$) which is shown to correspond exactly to classical negation, and to which the law of non-contradiction does apply.

In [da Costa et al 90] the question of how to implement a computational paraconsistent logic is addressed. The problem with the paraconsistent logic presented above is that it has a large number of inference rules, thus making automated inference extremely inefficient, and difficult to restrict to relevant inferences. Resolution theorem proving [Robinson 65] is a technique involving only one inference rule (resolution) on a restricted syntactic form of logic (conjunctive normal form) which is extremely popular for automated theorem proving. [da Costa et al 90] presents a resolution style proof procedure for a paraconsistent logic, in which formulae are annotated with a truth value from a lattice of many possible values. These are used, together with a unification algorithm, to restrict the use of contradictions in generating trivial inferences (thus effectively limiting the applicability of *ex falso quod libet*.). The proof procedure is proven to be sound and complete. The implementation of this proof procedure is illustrated by an example, and an extension is defined which is computationally more efficient.

A semantic tableau proof procedure for da Costa's paraconsistent logic is presented in [Carnielli & Marques 90], and its completeness is proven. Some examples demonstrate how their tableau method can be used, not only to reason over inconsistent sets of premises, but also to indicate which premises are involved in contradictions. This technique thus generates useful information about which subsets of a knowledge base are 'controversial' which can aid a user in making adjustments to that knowledge base. This approach is shown to be able to handle default information in a useful manner, and to be able to point out precisely where the use of defaults causes contradictions when reasoning with them.

2.1.1.5 Restricted Access Logics

In [Gabbay & Hunter 93b] an object-level logic called a Restricted Access (RA) Logic is presented which could underlie a meta-level system such as that discussed in [Gabbay & Hunter 91, 93a]. The RA logic is designed as an extension of classical logic (*ie:* all theorems of classical logic still hold) with a restricted form of *ex falso quod libet.* It is based on a labelled deduction system (LDS [Gabbay 90]). Gabbay and Hunter start from the common criticism of paraconsistent logics such as [Da Costa 74] which is that they outlaw certain inference rules and thus certain intuitively obvious tautologies in classical logic are not theorems of paraconsistent logic. Instead of restricting the proof rules or possible inferences, Gabbay and Hunter allow all the proof rules of classical logic, but restrict the *access* of these rules to the axioms, effectively restricting inference to consistent subsets of the theory. They do this by means of a restriction function which uses the labels of labelled formulae in the theory to determine the consistency of theory subsets. *Ex falso quod libet, reductio ad absurdum* and not-introduction incorporate this restriction function to check that any contradiction used in reasoning is localised to the most recent assumptions. This prevents contradictions in the theory from propagating as they would in classical logic.

In [Elvang-Goransson & Hunter 93], there is a further extension of the RA logic of [Gabbay & Hunter 93b] in which a modified form of the restriction function is used to partition the inference rules into three subsets. These are: the basic rules (b-rules) such as and-introduction, or-introduction and or-elimination; the paraconsistent rules (p-rules) such as the law of excluded middle, the law of double negation and the basic axioms of natural deduction involving negation; and the non-paraconsistent rules (c-rules) namely *ex falso quod libet, reductio ad absurdum* and not-elimination. For each subset there are 'sideconditions' which restrict the applicability of the rules to those propositions in the theory whose labels satisfy a certain form of the restriction function. These prevent the propagation of trivial inferences which have been generated from contradictions, whilst allowing for normal classical logical inferences over any consistent subset of the theory. This logic is thus more 'adventurous' than the paraconsistent logic of da Costa. In the second part of this paper, Elvang-Goransson and Hunter discuss how thinking of the labels of propositions as *arguments* for them allows for analysis of how *acceptable* a proposition is on the basis of the argument which is presented for it. Arguments involving inconsistencies can thus be regarded as less useful or acceptable than arguments from consistent information. A similar analysis can be found in [Elvang-Goransson et al 93].

A different technique for restricting access to subsets of an inconsistent theory is presented in [Naqvi and Rossi 90]. They start from a basic heuristic that propositions more recently added to a theory should override contradictory information already in

the theory. To facilitate inference in such a system a 'priority structure' is generated in which clauses added to the theory are assigned a 'priority level' which is higher the more recent the addition of the clause. A resolution theorem proving technique (SLIL-resolution) is then defined which incorporates a check for contradictions at any higher priority level when making inferences on lower priority information - if such contradictions exist then the inference step is blocked. Inferences involving the newer information, however, are not blocked by the existence of older contradictory information. This approach is extended to a temporal reasoner, and there is a discussion of how it can be used for hypothetical ('what-if P?' where P is consistent with the theory) or counter-factual ('what-if P?' where P is inconsistent with the theory) reasoning. Gabbay and Hunter acknowledge that their approach is extremely inefficient, however the principal problem with this system is the arbitrary decision that older information will always be over-ridden by newer information if they conflict. This may not generally be desirable, and in some extreme cases, it may simply not be possible to assign a temporal ordering to the items in a theory (particularly, for example, if they come from various different sources).

All of the logics examined so far have been attempts to cope with inconsistent theories within the logic by adapting the inference mechanism to avoid trivialising deduction in the face of inconsistent information. In the next section I turn to approaches which have involved stepping back from the logic and reasoning about the inconsistency at the meta-level.

2.1.2 Meta-level approaches to disagreement

There has been a recent upsurge of interest in inconsistency in databases and knowledge bases and some enthusiastic papers embracing it as an opportunity. For some logicians the distinction between object-level reasoning and meta-level reasoning (reasoning about the object-level propositions and inferences) provides a framework for isolating and exploiting inconsistencies. A good example is [Gabbay & Hunter 91] in which they claim :

> *'For Artificial Intelligence there is an urgent need to revise the view that inconsistency is a 'bad' thing, and instead view it as mostly a 'good' thing... There is a need to develop a framework in which inconsistency can be viewed according to context, as a vital trigger for actions, for learning, and as an important source of direction in argumentation.'*

In this paper they give various examples of people in normal situations being able to live quite happily with inconsistencies, and being able to reason at the meta-level from the existence of object-level inconsistency to appropriate actions or to express a belief in one or other of the conflicting positions depending on the circumstances. We can exploit our ability to be aware of inconsistencies and either suspend judgement until some point when one or other position is obviously stronger or actively use the

inconsistency to provide us with a greater range of possible responses to various situations.

One of their examples is of Professor Nobody who sees his research assistant, Dr Incompetent, sunbathing in the park while absent from work. When Dr Incompetent returns to work he claims to have been ill in bed. Professor Nobody acts as if he believes Dr Incompetent despite having information which contradicts his claim. Only later, after several repetitions of this incident, does Professor Nobody act on the inconsistency and sack Dr Incompetent. Gabbay and Hunter give several other examples of situations in which people express their belief in a position while simultaneously holding beliefs which manifestly disagree or conflict with that position. This is not merely deceit, as we often deliberately withhold our commitment to a belief until we are quite sure that we should change our mind, and, importantly, the factors that will persuade us to do so are often far removed from logical considerations.

We seem, Gabbay and Hunter suggest, to cope quite happily with overlapping, inconsistent belief systems without having to resolve the conflict at once - and intelligent computer systems would need to do likewise. They describe a *meta-level* framework which incorporates meta-level rules stating how to reason about, or take action according to, inconsistencies at the *object-level.*

In [Gabbay & Hunter 93a] this framework is explored further using an airline booking system as an example, and a meta-language is defined called DA (standing for *Data* and *Action*). The framework consists of an *information system*, defined as a pair *(M, O)* where *M* is a meta-level database and *O* is an object-level database. *O* is deliberately under-defined, as the meta-language is intended to support a variety of possible object-level languages. The meta-level syntax is defined in terms of the object-level variables, predicate names and logical symbols, and a set of constructors for the meta-level formulae, including the temporal operators, *SINCE* and *UNTIL*. One meta-level predicate is central to their system : it is called *Holds* and is defined to reflect all of the object-level theorems into the meta-level, *ie :* if A is an object-level formula, M the meta-level database and O the object-level database, then

$$M \vdash Holds(A) \quad iff \quad O \vdash A.$$

Various temporal operators are defined in terms of *SINCE* and *UNTIL* and are in place to represent the evolution of the database over time and the effects of actions on the database triggered by meta-level rules.

The interpretation (semantics) of the meta-language is defined as a tuple *(D, N, >=, h)* where *D* is the domain, defined as the *Herbrand Universe* generated by the meta-level rules of syntax. (The Herbrand Universe consists of all the ground terms in the

language, and a Herbrand interpretation assigns each term to be its own meaning, ie: each ground term in the meta-language is interpreted as itself. For more details see [Deville 90].) *N* and >= represent the 'flow of time', *N* being the set of natural numbers. h is a truth assignment function mapping every term, at any instant of time, to a truth value of 0 or 1. The temporal aspect thus allows an interpretation of a meta-level formula such as '*P* has held since *X* and will continue to hold until *Y*'.

They propose a temporal interpretation of propositions at the meta-level as being *declarative* for the past and present, and *imperative* for the future 'based on the intuition that a statement about the future can be imperative, initiating steps of action to ensure it becoming true'. They introduce *executable temporal specifications* which are consequents of the form

'If X then make Y true'.

In their conclusion they claim a complete and sound proof theory for the logic and also semi-decidability.

Gabbay and Hunter's approach can be contrasted with the paraconsistent logic approach as there is no need for blanket bans on certain types of logical inference at the object-level. The meta-level rules allow for much greater subtlety in selecting inference steps, or actions, depending on the nature and context of an inconsistency. Their approach can also be favourably contrasted with truth maintenance systems and belief revision systems (see section 2.1.3), as a proposition can be involved in an inference without necessarily outlawing other propositions which disagree with it from the belief set.

There are three main differences between DA and FORA (the framework presented in this document), the principle one being that Gabbay and Hunter only consider a single object level theory, whereas the contents of a FORA knowledge base reflect the contents of multiple potentially inconsistent knowledge bases. The second difference is that there is no *Holds* predicate in FORA These two points are related, as a *Holds* predicate in FORA would be profoundly inconsistent (reflecting all the differences of opinion in the knowledge bases) unless a different *Holds* predicate was defined for each knowledge base. More importantly, in FORA there is no notion of truth values of propositions (which the *Holds* predicate represents), as all that is recorded are the relations between the propositions. FORA does not adjudicate between opinions, it merely states how they relate to each other. Another difference is the lack of a temporal component in FORA. One way of integrating these two languages may be to interpret the single *Holds* predicate as representing 'the speaker's view over time' which in a system such as FORA, would be expected to change every time a new opinion is expressed. Thus the total history of the predicate *Holds*, over time, could be used to represent the 'floor' of a debate.

In general their work is extremely relevant to this thesis, as demonstrated by this statement from the conclusion of [Gabbay & Hunter 93] which accurately sums up my position:

> *'We give up a requirement to make the object-level database consistent, and rather accept such situations as inevitable. We abstract away from the object-level, and shift the requirement of consistency to the level of the meta-level being consistent.'*

Only the details of the two meta-languages differ.

A preliminary application of the DA framework is discussed in [Finkelstein et al 93] in which it is used to identify and handle inconsistencies in the development of software specifications. Finkelstein is concerned with the construction of software specifications from multiple perspectives (Viewpoints) and is constructing a distributed software specification environment. Consistency checks can be provided to ensure consistency within a single viewpoint and also to detect inconsistencies *between* viewpoints. The meta-level action framework thus allows for appropriate actions to be taken in the case of inconsistencies, such as checking for spelling or typing errors, and ultimately asking the system users for clarification. The existence of multiple object-level viewpoints (each of which can use different representation languages) makes this work highly relevant to this thesis. It is interesting to note that they identify as a major difficulty the axiomatisation of inconsistency detection at the meta-level. They have also not investigated ways of exploiting inconsistencies (once detected) to drive interaction with users, other than the extremely simple examples of asking about spelling errors. They earmark this task as requiring further work, and state the need for *protocols* for inter-viewpoint interactions. This thesis contains some ideas which may meet this need.

2.1.3 AI Approaches to Conflict

Negation as failure

There are problems with practical computation using logics with classical negation. The usual practical solution to this is to use a form of negation known as *negation as failure*, however, this inference strategy is both unsound and incomplete (see eg: [Lloyd 84], [Deville 90] for more details).

Conflict resolution

Inference engines and interpreters in knowledge based systems need to handle situtations when more than one possible inference or problem solution is applicable which may provide conflicting conclusions. Strategies for handling such choice points are usually referred to as *conflict resolution strategies*. Examples are to accept

options on the basis of criteria such as *recency* of data, *specificity* of possible solutions, *refractoriness* or obstinance in refusing to retry possible solution paths which have been attempted previously, and so forth. Such strategies are routinely built into interpreters for rule-based systems, see, for example [Jackson 90] for details.

Default logics and non-monotonic logics

If *change* in the world is to be taken seriously in AI systems, conflicts must be addressed as changes frequently create situations where a statement previously held to be true is contradicted by the current state of affairs. If addition of new propositions to a logical theory does not invalidate any of the previous items in that theory, then it is said to be *monotonic*. As more inference is carried out the knowledge base just grows and grows. However, if the addition of new propositions requires other propositions to be removed in order to restore consistency, then we have *non-monotonicity*, where a knowledge base may either expand or shrink with the addition of new information. The problem of providing logically rigorous characterisations of this process is an important endeavour in AI, and a rich seam of research exists from early work on the *frame problem* (which aspects of our knowledge do we need to adjust in the light of changes in the world, and what is persistent? [McCarthy & Hayes 69], [Raphael 71], [Pylyshyn 78]), common-sense reasoning, circumscription (from [McCarthy 80] onwards), default logics ([Reiter 80], [Reiter 85] and onwards to [Etherington 88]) and now in the mature work on non-monotonic and defeasible logics (eg: [Ginsberg 87], [Brewka 91]).

Truth and Reason Maintenance Systems

It has been recognised for some time within AI that it is useful to record the *justifications* for a belief in a proposition so that if certain assumptions used in those justifications are found to be false, the belief system or knowledge base can be adjusted. One way of doing this is to think of propositions as nodes in a network, with each node labelled to indicate which assumptions are used to support it. In this way a belief in a proposition is recorded with reference to its dependencies on other propositions. A useful analogy sometimes used here is to think of these dependencies as a lattice of scaffolding which supports all the various propositions in a theory. If there is some reason to doubt a proposition, this is analogous to taking some of the scaffolding down, which may cause other propositions to become unsupported. Likewise, adding a new piece of scaffolding to support proposition P may have the knock-on effect of increasing the support for another proposition Q, if P is used in a justification for Q. The job of a truth maintenance system is to maintain the scaffolding so that each node is supported by a consistent set of other nodes, and no contradictions are supported.

Conflict is handled in truth maintenance systems by explicitly recording which sets of assumptions lead to contradictions, and taking apart any scaffolding which supports a contradiction. In [Doyle 79] and [McAllester 80], assumptions which are guilty of supporting contradictions are found using *dependency-directed backtracking*, in which the scaffolding is traced to its 'roots', and the user of the system is then asked which assumptions will be removed, thus removing the support for one of the two contradictory propositions. See below, in the section of belief revision, for some criticisms of the way this happens in practice.

In de Kleer's ATMS (assumption-based truth maintenance system) [de Kleer 86], this scaffolding is not removed, but rather maintained for various possible scenarios, any one of which may hold depending on which assumptions are currently believed. There is a special node called 'nogood' or FALSE, which represents a state of inconsistency. Any set of assumptions which leads to an inconsistency becomes a label for the 'nogood' node, and this set of labels is used to maintain the consistency of the others, by removing assumptions which lead to inconsistencies from the justificatory labels of proposition nodes. De Kleer's terminology for his truth maintenance is rather unintuitive, as he describes the label for a proposition node as the list of all the consistent *environments* in which the proposition holds, where an environment is a set of assumptions. A somewhat clearer explanation is given by [Ginsberg 93] in which a label is said to contain the *explanations* for a proposition. The most coherent way of thinking about such labels in the context of this thesis is as *arguments* for the proposition, though being merely lists of assumptions, they contain less structure (less of the proof tree) than some of the argumentation approaches we will examine later.

De Kleer's work is sometimes described (eg: in [Jackson 90]) as allowing hypothetical reasoning over more than one possible world. We can usefully compare this approach with Rescher & Brandom's Logic of Inconsistency (see section 2.1.1.3), in which possible world descriptions are superposed. In the case where the worlds are mutually consistent, superposing their descriptions simply produces a further, more highly specific, world description. However, if the worlds are each internally consistent (as all of Rescher & Brandom's are), but mutually inconsistent, in de Kleer's system the superposition of the two descriptions is 'nogood' and ruled out altogether (*ie:* becomes the label for the 'nogood' or FALSE node). Rescher & Brandom's logic is thus much more flexible, allowing certain inferences still to be allowed from the inconsistent world description. Another way of seeing this is to note that in an ATMS *all* contradictory sets of assumptions are labels of the one node 'nogood', and thus all inconsistent theories are thus treated as *equivalent* (and equally useless). In Rescher & Brandom's logic, each inconsistent theory remains distinct.

An important feature of the ATMS should be noted here, which is that the 'scaffolding' structure is a meta level representation of the inferences and justification which is carried out on an underlying database using some sort of object level theorem prover. The operations of the truth maintenance system are thus meta-level deductions, and are theoretically independent of the particular object-level theorem prover or object-level representation language. For more detailed studies of truth maintenance systems see [Ginsberg 87].

Belief Revision

The truth maintenance systems of Doyle and de Kleer were the precursors to a significant body of recent work aimed at providing epistemologically meaningful and computationally feasible approaches to the problem of *theory change* or *belief revision*. This work is interesting because it explicitly acknowledges that as the world around us changes, old beliefs may contradict more recent experience, thus causing inconsistencies in our belief systems. As Nebel succintly explains [Nebel 92], 'belief revision is the process of incorporating new information into a knowledge base while preserving consistency'. It is thus important for researchers into belief revision to understand how beliefs can be represented in such as way as to enable detection of inconsistencies. In addition, inconsistencies need to be isolated or localised so that it is possible to see directly *where* an inconsistency arises. There is a recognition of the fact that conflicts in our belief systems are not all-pervasive, and that it is important to be able to recognise which beliefs need not be affected by a particular belief change. For example, a change in our beliefs about the current weather may affect our belief in what is a good plan for an outing today, but will not affect our beliefs in physical laws such as those about gravity or even in changable beliefs about wider issues such as the current political situation, or the leader of the country. This is really a reiteration of the familiar old frame problem.

Gärdenfors [Gärdenfors 92] identifies two main approaches to belief revision. The *foundations approach* has developed from the work described above on truth maintenance and it is characterised by explicit representations of the reasons for holding each belief, which are used as the structural basis for organising changes to the belief base and maintaining consistency.

The *coherence approach* [Harman 86], [Gärdenfors 88] is characterised by representations of 'epistemic entrenchment' which provide an ordering of items in the belief base, according to which beliefs are more or less entrenched and difficult to retract (this is along the lines of Quine and Ullian's analysis of the criteria by which we should revise scientific hypotheses, more details of which are given in the discussion of argumentation in section 2.2.1.3 [Quine & Ullian 78]). In the coherence approach,

revision is analysed in terms of the entire belief base, rather than particular reason or argument structures, according to general principles such as changing the belief base minimally. An important contribution to this area are the 'AGM Postulates' which are a set of general postulates describing the desirable properties of belief revision, belief set expansion and belief set contraction. [Gärdenfors 92] contains several theoretical and algorithmic extensions to this work. An important theoretical question here is how to represent and generate epistemic entrenchment orderings and how to use them to handle inconsistency. [Dubois & Prade 90] present a possibilistic logic approach to this problem, providing numerical measures of entrenchment.

Doyle's reason maintenance system (RMS), described above, is a classic example of the foundations approach. In [Doyle 92], Doyle responds to claims that the foundations approach is inferior (in both inferential plausibility and computational expense) to the coherence approach, by offering an analysis which suggests that it may not be possible to draw such a clear distinction between the two approaches, and also providing some interesting revelations about the practical matter of implementing belief revision systems. The logical and computational expense of testing belief bases for consistency on the basis of either reason structures or entrenchment orderings is potentially astronomical. Consistency checks are at worst undecidable in any interesting logical representation. In addition, most of the theoretical work in belief revision deals with belief sets which are closed under logical consequence. Determining logical closures of belief bases is itself a potentially infinite operation. Hence Doyle concludes that in practice, both approaches 'must abandon most requirements of logical consistency and closure to be useful in practical mechanisations'. He makes some particularly relevant remarks about how these requirements are overridden in his system :

> *'RMS, for example, lacks any knowledge of what its nodes mean, depends on the reasoner to tell it when some node represents a contradiction, and leaves the conflicting beliefs in place if they do not depend on defeasible assumptions'.*

Some research bridges the foundations and coherence approaches, taking useful ideas from each and thus organising revisions according to local reasons for propositions and more global aspects of belief base coherence. A good example is Galliers' theory of autonomous belief revision for cooperating agents whose beliefs come into conflict during communication [Galliers 92]. Her analysis provides grounds for deciding *whether* to accept a statement and revise beliefs accordingly, as well as *how* to carry out that revision.

[Hansson 92] also provides an agent-based analysis of belief revision by a dyadic represention of belief sets - the dyad is a pair of logical structures, one representing the set of core beliefs of an agent (the belief base) and the other representing the

inferences which that agent considers to be valid (possibly in terms of a simple consequence operator). This provides the grounds for an interesting, though brief, discussion of disagreement which is defined in terms of an agent as an inconsistency relative to their consequence operator or inference set. Inconsistencies in belief bases are called *factual disagreements*, and inconsistencies in the set of derivable conclusions are called *inferential disagreements*. One nice aspect of this work is that it provides a way of capturing differences between different agents' perceptions of inconsistency, and makes explicit 'how two individuals may, even with full knowledge of each others' beliefs, have different opinions on whether they disagree or not'. They unfortunately do not indicate how this analysis of disagreement could be used.

There are some similarities between belief revision systems and FORA, largely due to the use of meta-level relations between propositions. In the foundations approach there are explicit representations of arguments (or at least justifications) while in the coherence approach there is room for other relations which support epistemic entrenchment orderings. The equivalence and elaboration relations in FORA would look at home here. A main difference is the lack of anything corresponding to the disagreement relation. The exploration of conflict sets in FORA is related to the notions of belief set expansion and contraction. FORA's higher-order argument structures could provide valuable support for the coherence approach to belief revision.

In summary, research in belief revision is relevant to this thesis because it starts from the same basic question : 'What to do in the face of inconsistency in a knowledge base?', and thus requires similar logical machinery for analysing conflicts. However, most of the research in belief revision differs strongly in attitude from this thesis in that the principal answer to the question is 'eradicate the inconsistency', however some of the detailed reports summarised above suggest that in practice this is an untenable aim and is heavily compromised when building working computational systems. This thesis can thus be seen as a contribution to the belief revision debate by attempting to make a virtue out of this necessity. Rather than weakening the theory when it comes to implementation due to the inability to escape from inconsistency in practice, I have built the existence of such conflicts into the theory.

Having completed this survey on approaches to the problem of handling disagreements, I now turn to the field of argumentation in search of potential solutions.

2.2 Argumentation

The second body of literature relevant to the study of disagreements concerns the way in which we produce **arguments** for our various points of view. The construction and criticism of arguments is the fundamental activity of debate and is thus crucial to any computer-based debating system.

2.2.1 Argumentation in philosophy

2.2.1.1 Traditions of argumentation

Greek beginnings

Any survey of the argumentation literature must begin with Aristotle (see, for example, [Aristotle 84] and [Evans 77]), the first great logician, who first systematically laid out the *form* of valid argument in the shape of the four syllogisms. A recognition of Aristotle's work is relevant here for two reasons, not only because his work concerned the structure of arguments, but also because the spirit of his enterprise was closer to the spirit of this one than much of twentieth century logic. Aristotle was concerned with argument structures, not as a purely formal and abstract study, but because he sought to strengthen and facilitate the *practice* of debate in Greek society. To Aristotle, the syllogistic forms were not idealisations, nor rules for manipulating symbols, they were clarifications of the real, practical arguments used in everyday fora of criminal justice, land allocation, political decision making and theological debate[1].

Thus the early days of logic provided structures for assessing the arguments and disagreements of the real world. Only comparatively recently has logic become a formal study in its own right.

The formalist tradition

This practical emphasis in Aristotle's work has been largely neglected by later generations of formal logicians, in particular since the 'mathematisation' of logic by Boole and of linguistic argumentation by Frege. Frege's *Begriffsschrift* (concept-script) was an explicit attempt to subsume all sound reasoning by a function-theoretic calculus [Frege 50]. By comparison, natural language appeared defective, as the following commentary on Frege's work makes clear:

> *'Natural language, [Frege] thought, is rife with vagueness, ambiguity, lack of logical perspicuity, and, indeed, logical incoherence. To a large degree he identified as 'logical defects' in a language those features of it which fail to correspond with the articulations of his concept-script. The logical*

[1] The name of the FORA system is intended to acknowledge this origin.

powers of concept-script in the presentation of arguments so far outstripped anything hitherto available that Frege unwittingly employed his invention as a yardstick against which to measure natural language' [Baker & Hacker 84].

The same view still appears to be held by many artificial intelligence researchers today, and is reflected by the efforts of the natural language processing research community to capture the structure and semantics of natural language in formal logics which enable language to be processed (both understood and generated) by computer programs.

The elegance of Frege's formal systems led increasingly to a perception of natural language as inadequate for the expression of conclusive arguments and to the Russellian cult of the 'logically perfect language', of which Russell said 'a language of that sort will be completely analytic, and will show at a glance the logical structure of the facts asserted or denied' [Russell 56]. The implication is that the practical business of wrangling over disagreements lies outwith the boundaries of logic, because the very existence of disputes is rooted in the imperfections and vaguaries of the natural language used to express them. It is thus unsurprising to read Russell's dismissal of Aristotle's conception of logic:

'I conclude that the Aristotelian doctrines .. are wholly false with the exception of the formal theory of the syllogism, which is unimportant. Any person in the present day who wishes to learn logic will be wasting his time if he reads Aristotle or any of his disciples.' [Russell 46].

The interpretivist tradition

Fortunately, the story does not end there. Throughout Western Philosophy there has been an interest in debate or dialectic, and a dialectic tradition of philosophical discussion championed by philosophers such as Kant and Hegel. Kant was the first great meta-philosopher. At a time (the eighteenth century) when all great philosophers produced monumental theories to explain the way the world is, Kant created his own treatise on 'Pure Reason' [Kant 33]. But as well as being a conventional philosopher in this sense, he was also a professor and historian of philosophy, and thus also wrote about how others explain the world, and how those explanations relate to each other via the process or dialectic or argumentation. Kant aimed at untangling the huge philosophical controversy between the Rationalist tradition (headed by Descartes) who believed in the primacy of reason in our search for explanations of the world, and the Empiricist tradition (led by Locke and Hume) who believed that we must base our understanding of the world on our senses and experience. Kant's lasting contribution to this debate was to articulate the two sides of the argument, and although he also proposed an alternative way of explaining the world, this is less important here than the fact that he talked about the debate itself.

Hegel directly continued this enterprise of meta-philosophy, denying the existence of any 'truth of the matter', asserting instead that all truth is subjective, and relative to the time, place and culture in which we find ourselves. To Hegel, truth flows and changes through history by *dialectic* (see [Rosen 82]), which is a movement from a claim (or *thesis)* via its rebuttal (or *antithesis)* to a new position (or *synthesis).* He was the first interpretivist, claiming that no fact holds merely in itself but rather its truth must be determined relative to some context. Later interpretivists such as Heidegger go further, claiming that not only the *truth*, but also the *meaning* of statements is relative to a historical or personal context within which it must be interpreted before it can be understood.

Thus in much twentieth century philosophy, the role of disagreement is important, and the occurrence of different points of view is considered a part of the human condition, necessarily the case due to the variety of cultures and historical perspectives in human society. Argumentation and dialectic are central to our reasoning faculties. But in formal logic and mathematical philosophy as articulated by Russell, disagreements and inconsistencies are deeply problematic. The work of Wittgenstein spans these two schools of thought. His early work in the *Tractatus Logico-Philosophicus* [Wittgenstein 22] demonstrates his formalist philosophical beginnings, whilst his later work and particularly *Philosophical Investigations* [Wittgenstein 58] represents a radical shift towards an interpretivist stance with its emphasis on the instrinsic importance of the immediate context (or language game) to the meaning of everything we say.

The explosion of the study of knowledge, reasoning, logic and artificial intelligence, in the second half of the twentieth century, is rooted in the dynamic and controversy resulting from these two competing approaches - interpretivism and formalism.

2.2.1.2 Toulmin's practical reasoning

In the 1950s there was a backlash against the formalist monopoly of logic, led by the philosopher of science, Stephen Toulmin. Toulmin's central concern is the needs of professionals in fields such as law and the natural sciences to assess the conclusiveness of arguments in their domain and to carry out rigorous debate. Toulmin believes that the mathematical trends of formal logic have short-changed lawyers, scientists and certainly ordinary people, and in his book *The Uses of Argument* [Toulmin 58] he reasserts the practical nature of the logical enterprise.

His principal focus of attack is the Russellian claim that the only valid arguments are analytic ones, which when expressed in a 'logically perfect language' can be seen at a glance (or by a computer) to be sound. Instead he asserts the need for logic to encompass non-analytic, or *substantive* arguments which can significantly add to our

knowledge of the world. An important effect of the artificial constraint of analyticity of arguments in logic, he says, is that this has caused apparantly insurmountable problems in the field of epistemology, or the study of knowledge. Those things which we claim to know, yet can justify only by substantive, non-analytic arguments (for example, results of induction in science or the reports of eye-witnesses in court) appear epistemologically dubious when our yardstick for accepting them are the analytic arguments of formal logic. Toulmin's claim is that this should cast doubt not on the substantive arguments but on the claims of formal logicians to be accounting for sound reasoning. 'The only real way out of these epistemological difficulties is .. giving up the analytic ideal.'

The reason that Toulmin's work is important to this thesis is partly because his vision of a new field of logic is remarkably close to the reality of what is now studied in artificial intelligence. This new field has the following three requirements:

> *'(i) the need for a rapprochement between logic and epistemology, which will become not two subjects but one only;*
> *(ii) the importance in logic of the comparative method - treating arguments in all fields as of equal interest and propriety, and so comparing and constrasting their structures without any suggestion that arguments in one field are 'superior' to those in another; and*
> *(iii) the reintroduction of historical, empirical and even - in a sense - anthropological considerations into the subject.'[Toulmin 58]*

The importance of logic in AI and the claim that it is 'experimental epistemology' are indicators that AI is meeting the first need. The explicit and expanding level of interest in practical applications of AI, for example the representation of arguments from medicine, law, chemistry, geology and resource management in knowledge based systems, is an indication that the second requirement is also being met. The third claim, of the need for empirical, historical or anthropological study is being vindicated by recent interest in the situtatedness of intelligent behaviour and the cultural aspects of communication.

Having established the relevance of Toulmin's work to AI, it is worth looking more closely at his analysis of the structure of arguments, and his criticisms of the approaches of mathematical logic. Like Wittgenstein, Toulmin grounds his investigation in our everyday language, and thus many of his most telling points derive from close scrutiny of the idiomatic way in which we express ourselves. It is this sort of scrutiny which leads him in his essay 'The Layout of Arguments' [Toulmin 58] to challenge some of the unjustified simplifications he believes pervade logicians' work.

He starts with a metaphor : 'An argument is like an organism. It has both a gross anatomical structure, and a finer, as-it-were physiological one'. Toulmin wishes to

challenge some of the assumptions made at the 'physiological level', in particular the notion of logical form and the traditional way in which we carve up and label the parts of arguments. The result is an alternative model of the structure of an argument to the 'premiss set leading to conclusion' model which pervades the study of logic.

Take the syllogism:

(1) Wendy was born in the Falkland Islands.
(2) All people born in the Falkland Islands are British Citizens.
(3) Therefore, Wendy is a British Citizen.

Traditionally, (3) would be called the conclusion, (1) the minor premiss (being about a specific named individual) and (2) the major premiss (being a general rule). Although Toulmin does not press the point, most formal logics (and rule-based systems) retain this structure, and provide general inference rules (for example, elimination of the universal quantifier, followed by modus ponens) which enable this argument to be classed as valid by virtue of its *form*. The syntactic structure of the argument is emphasised, and the fact that it is about the Falkland Islands, Wendy and notions of citizenship is sidelined as irrelevant to its validity.

Toulmin introduces a more complex representation of this sort of argument by which he points out distinctions between different argument-types, which are lost under the syllogistic or classical interpretation. The following terminology is introduced :

A *claim* is the conclusion of the argument, the statement for which justification is required, in this case 'Wendy is a British Citizen.'

A *datum* is a statement of fact offered in evidence for some claim, in this case, 'Wendy was born in the Falkland Islands'.

A *warrant* is a general rule or principle which is used to support the step from a datum to a claim, it is a justification that such a step is legitimate, in this case 'All people born in the Falkland Islands are British citizens'.

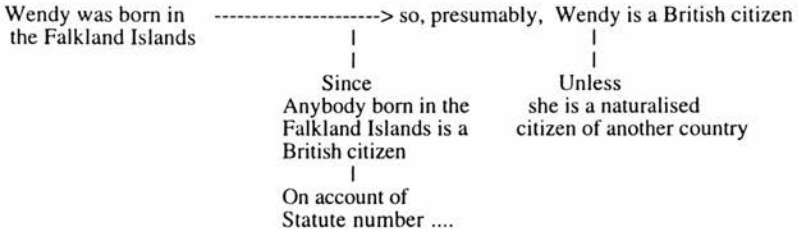
A *backing* is a support for a warrant, for example, a reference to some Act of Paliament which states the citizenship of Falkland Islanders. A crucial factor for Toulmin in distinguishing warrants from backings is that backings vary widely depending on the field of discourse - so general warrants about legal matters should be expected to have backings of a very different sort to the backings for a general warrant in the physical sciences.

A *rebuttal condition* is a statement of the condition under which we would not expect a

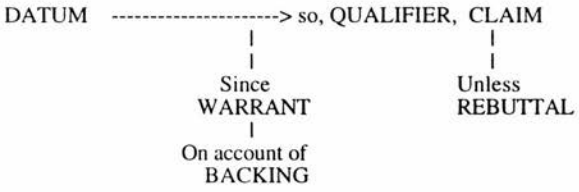
general warrant to hold, for example 'unless Wendy is a naturalised citizen of another country'.

A *qualifier* is a modal expression describing the applicability of the warrant, for example 'necessarily', 'presumably', or 'possibly'.

Thus Toulmin would lay the argument out as follows :



This fits a general template as follows :



This layout serves an important purpose which is to enable Toulmin to demonstrate how the distinction between warrant and backing reveals subtle differences between arguments which would be represented as the same in classical or syllogistic logic (or indeed, in rule-based systems). He is particularly suspicious of the use of universal quantification in conflating two different sorts of generality those resulting from past experience or empirical study ('according to a survey, 95% of Falkland Islanders are British citizens') as opposed to claims about appropriate deduction ('You can safely assume anyone born in the Falklands is entitled to Critish citizenship'). The former is a backing, whereas the latter is a warrant. To Toulmin it is vital to make this distinction because it will depend on the circumstances of a particular argument whether the former could be accepted as an appropriate backing for the warrant 'All people born in the Falkland Islands are British Citizens'. In a court of law, for example, it is unlikely to stand up. In a demographic study it might suffice. Most importantly in these two cases we would apply different criteria in assessing its

veracity. (As an aside, it is also worth noting how closely the notions of warrants and rebuttal conditions correspond to the notions within AI of defaults and exceptions.)

In summary, then, Toulmin was led to doubt whether 'the traditional pattern for analysing micro-arguments - 'Minor Premiss, Major Premiss, so Conclusion' - was complex enough to reflect all the distinctions forced upon us in the actual practice of argument-assessment' and to conclude that many fundamental problems in the logical tradition (for example the problem of attaining knowledge from non-analytic arguments) result from 'this vast initial over-simplification'.

Toulmin's position has been presented at length here for various reasons. Firstly, it has influenced several researchers in AI recently, as will be seen below. Secondly, Toulmin's work is rarely articulated in any detail and thus is not as well known as it should be, perhaps due to its radical stance (the same can be said of Wittgenstein). Thirdly, this thesis contains analysis of higher-order argumentation structures which capture some of Toulmin's requirements, for details of which see section 5.3.

2.2.1.3 Quine's Rationalism

Toulmin would perhaps be pleased by Quine and Ullian's attempt [Quine & Ullian 78] to bring logical analysis to the notice of people involved in real world argumentation. Their book, however, is a rather unsettling mixture of dogmatic rejection of 'anti-scientific' projects such as theology, coupled with wise advice to be open to arguments and changes to our belief systems. They provide detailed advice, in the form of six 'virtues' which we should strive towards in our analysis of hypotheses about the world. These six virtues have been cited by researchers into the coherence approach to belief revision (see section2.1.3) as support for various characterisations of epistemic entrenchment. They are :

1. *Conservativity* : aiming to make minimal changes to the overall belief set.
2. *Modesty* : hypotheses should be the minimal required to explain observations.
3. *Simplicity* : hypotheses which are simply statable should be preferred over complex ones.
4. *Generality* : hypotheses which explain many observations should be preferred over those which explain only particular observations.
5. *Refutability* : there should be ways of subsequently proving a hypothesis to be false.
6. *Precision* : hypotheses which provide predictions which can be quantitatively tested should be preferred over qualitative generalities.

There are obvious tensions between these virtues, as shown, for example, by Einstein's theory of relativity which though simple and general, was neither modest

nor conservative at the time, as it involved revising many basic physical beliefs about the nature of time and space, the ether and the applicability of Newtonian mechanics. Quine and Ullian relate situations in which their first two virtues can be over-ridden to Kuhn's notion of scientific revolution or paradigm shift [Kuhn 62].

In the final chapter of their book, Quine and Ullian provide some strategies for arguments. For example, 'To convince someone of something we work back to beliefs he already holds and argue from them as premises'. However, 'often there is also a negative element to contend with : actual disbelief of some of the needed premises', *ie:* disagreement. In this case there are two strategies. We can either attempt to *overwhelm* our listener by 'adduc[ing] such abundant considerations in favor of our thesis that we end up convincing the man in spite of his conflicting belief.' This strategy relates to the generation of arguments, corroborations and enlargements in the terminology of this thesis. The second strategy is to *undermine* our listener's arguments. 'We must directly challenge his conflicting belief... If he meets the challenge by mustering an argument in defense of that belief, then we attack the weakest of the supporting beliefs on which he rests that argument'. This corresponds directly to Toulmin's notion of *undercutting* of an argument. An alternative characterisation of undermining, in terms of the more indirect attack on elaborations of a position is given in section 5.3.

An interesting acknowledgement of the dynamism of debate is worth including here : 'What may occasionally happen is that our challenge to the conflicting belief is met by so able a defense that we find ourselves persuaded. In this event we are led to give up the very belief that we originally sought to propagate.' It is important to note that this account begs some crucial questions, such as how we assess weakness of supports (in order to aim our challenges where they are likely to succeed) and what factors constitute a defense being strong enough to convince us to change our mind. These are now central questions in belief revision research.

To conclude this section of some philosophical approaches to argument here is Quine and Ullian's inspired 'gardening metaphor' for debate:

> *To maintain our beliefs properly even for home consumption we must attend closely to how they are supported. A healthy garden of beliefs requires well-nourished roots and tireless pruning. When we want to get a belief of ours to flourish in someone else's garden, the question of support is doubled : we have to consider first what support sufficed for it at home and then how much of the same is ready for it in the new setting.*

2.2.2 Argumentation in AI

Given the centrality of *reasoning* in artificial intelligence research, it is not surprising that argumentation has received some attention in AI as a reasoning technique.

The first area of relevant work concerns the maintenance of consistency amongst large sets of interrelated beliefs or propositions. The solutions to this problem presented in [Doyle 79] and [de Kleer 86], *ie:* truth or reason maintenance systems, and also belief revision systems, bear a strong similarity to the work presented in this thesis. They involve reasoning, in the face of inconsistency, about the chains of justifications used to support propositions in a network. They differ from the work here in that their aim is always to restore consistency, so the existence of disagreements or inconsistencies is always only fleeting, and always resolved away. To do this, all propositions (with their justifications) are explicitly tagged as either believed or not, (IN or OUT is the usual terminology - *ie:* in, or out of, the current set of believed, or well supported, propositions). Two inconsistent propositions cannot both simultaneously be IN and sophisticated truth maintenance algorithms have been devised to resolve such conflict situations. This effectively restricts the network of propositions to a representation of a single consistent viewpoint on the world, albeit one which knows about other possible consistent views which could be, but which are not currently, believed. For more discussion of such systems see section 2.1.3.

Another important concern of AI research is the problem of uncertainty. Since the earliest expert systems, such as MYCIN [Buchanan & Shortliffe 84], were developed, it has been realised that ways are needed to reason with information which is uncertain, or with statements which are believed only to a certain degree, and that conclusions which are drawn on the basis of such reasoning should be qualified appropriately with some degree of belief or uncertainty. Most of the mainstream approaches to this problem have used the mathematical tools of probability theory, such as Bayes Theorem, Dempster-Schafer theory and so on, sometimes in modified or simplified form (*eg:* the manipulation algorithms for uncertainty handling in MYCIN). These require the level of belief in statements to be quantified, as some kind of probability, and algorithms to be devised to manipulate these quantities as reasoning is carried out, enabling the uncertainty levels of premises and inference rules to be propagated so that conclusions reflect this uncertainty. Where multiple conclusions can be drawn the numbers allow comparisons between them to be made as to which is most reliable. Such quantitative approaches to uncertainty will not concern us further here. Although they are useful in certain contexts, it has been widely argued (see next section) that such quantitative approaches to reasoning about uncertainty are limited, and that richer, symbolic and qualitative representation tools can be used to provide a more plausible account of how we actually reason in the face of uncertainty.

2.2.2.1 Cohen's Model of Endorsements

An important example of an argument against the adequacy of probabilistic approaches to uncertainty, and presentation of a *qualitative* uncertainty representation technique, is provided by Cohen's work on endorsements [Cohen 85]. My main interest here in Cohen's technique is that it involves meta-level reasoning. He argues that it is not sufficient merely to propagate numerical uncertainty factors whilst reasoning. Instead he advocates reasoning *about* the nature of the uncertainty in order to facilitate reasoning with the underlying uncertain statements. To do this, the uncertainty in a statement or inference rule is not given a numerical degree, but instead it is described by further statements (which Cohen calls endorsements) which give reasons for belief or disbelief, such as qualitative descriptions of the reliability of the information used to support the belief; they state to what degree and in what contexts the uncertainty is important; and they provide techniques for reducing or resolving it.

Cohen's endorsements can thus be viewed as a representation which enables argumentation amongst a set of possibilities. Indeed Cohen freely describes the reasons for being more or less certain of a statement (*ie:* its endorsements) as *arguments* for and against those beliefs, and he uses examples of arguments to motivate his model and to provide him with examples of particular endorsements.

For example, he analyses in some detail the anthropological arguments presented by Walker and Leakey in an article in *Scientific American*, addressing various hypotheses about the number of different species represented in fossil remains found in East Turkana in Kenya. Three types of skulls were found, described as *robust, gracile* and *erectus*. The hypotheses are that these represent either one, two or three distinct species. They provide evidence relevant to the decision and they combine subsets of the evidence into arguments for and against the various hypotheses.

Cohen analyses these arguments according to how they contribute to the task of weighing up the evidence for and against each hypothesis, which he claims is a process like account-keeping.

> *'The two steps in analyzing arguments in the model of endorsement are to decide which column of the ledger-book an argument belongs in and to do the 'accounting' of arguments.'*

To facilitate this he represents the arguments as conditionals in predicate logic, reasons with them using a backward chaining rule interpreter, and attaches them as endorsements to the statement representing their conclusion.

Arguments are only one aspect of endorsements. They provide good reason for believing a conclusion. Cohen's endorsements also include statements of the

reliability of information (*eg: 'Premise is highly believable', 'data reliability = poor'*) and of relationships between arguments (*eg: 'Corroboration between this argument and the previous one'*) which are used by his system (SOLOMON) to rank conclusions by comparing their endorsements. This is the main way in which Cohen's system differs from truth maintenance systems (in which the kind of support for a proposition is irrelevant). In Cohen's work, the nature of the support (or endorsement) is crucial for this ranking process.

A key difference between Cohen's model of endorsements and FORA is that in FORA arguments are not used for deciding the level of confidence we can have in their conclusions. Another is that in FORA the representation of arguments is more abstract - it does not involve representing them as conditionals in first order predicate logic, nor restrict reasoning to backward chaining or any particular object-level inference mechanism. There is a clearer distinction made here between object-level decisions such as these and the meta-level representation of arguments. As a result of this, notions such as corroboration between arguments, which Cohen represents as just another form of endorsement, are defined formally and in general at the meta-level and can thus be recognised automatically by the system, rather than having to be tagged by hand onto conclusions in the rather inelegant way used in Cohen's model (*'Corroboration between this argument and the previous one'* is added as an endorsement to a new conclusion identical to one drawn already using a different argument, and *Corroboration between this argument and the subsequent one'* is added to the first conclusion). It is thus possible that FORA could provide some useful extra facilities in endorsement-style uncertainty handling, however this has not been attempted in the current work.

To summarise, Cohen contends that faced with uncertain conclusions we do not merely compare our levels of uncertainty in them, but that we jump to the meta-level and reason about the uncertainty itself, in particular looking for ways in which we could seek more information to strengthen or weaken our belief in one or other conclusion. Cohen's work is similar to the work here in that it rejects the notion that inconsistencies should be immediately resolved (by numerical comparison of probabilities, for example), and advocates using them as an *opportunity* for further reasoning about the point at issue. His work is also interesting because he recognises that our levels of belief are not absolute and that reasons for believing a statement will be convincing in one situation but not necessarily in another. An argument for a belief is thus not instrinsically good or bad, it is more or less persuasive depending on the context it is used in, who it is used by and so forth.

2.2.2.2 Argumentation at the Imperial Cancer Research Fund

At the Imperial Cancer Research Fund (ICRF) in London, there is a long standing interest in using AI techniques to support the assessment of cancer risks, diagnosis of cancer in patients and clinical treatment decision making. All three of these tasks involve reasoning in the face of uncertain information. The ICRF's research has produced a body of work which, like Cohen's, moves away from traditional probability-based uncertainty handling, and instead attempts to bring rigour to the process of reasoning with qualitative or semi-quantitative measures of uncertainty [Fox and Krause 92]. One thread which runs through this research is the use of *argumentation* as a way of weighing up conflicting or inconclusive evidence about alternative diagnoses, risk-assessements or treatment regimes (eg: [Fox and Clark 91]). They have successfully demonstrated that it is advantageous to move away from classical decision theory (such as [Lindley 85]) in which which decisions are made by weighing up quantitative probability and utility functions, towards a qualitative, argument-based approach to decision making as advocated by Toulmin [Fox et al 92]. Some of their more recent research concerns the use of multiple cooperating agents to represent the various points of view which need to be taken into account when making rigorous medical decisions [Fox et al 94], [Huang et al 94], [RED 94].

The most significant result of their research from the point of view of this thesis, is the *Logic of Argumentation*, LA [Krause et al 95a], which is based on a subset of intuitionistic logic involving only 'conjunction' and 'implication', known as 'minimal logic'. LA is an extension of this logic in which propositions are labelled with formal representations of *arguments* for them. An argument for a proposition is a lambda-calculus structure representing the form of a proof of the proposition, however, arguments can be constructed in this logic which may not in fact be logically sound (*eg:* they may be based on invalid assumptions). Thus these argument representations look like proofs, but may not actually have the *force* of logical proofs. Negation ($\neg$P) is interpreted as P$\rightarrow\perp$, where $\perp$ means 'contradiction'. In most interesting cases it is possible to construct arguments for a proposition and also for the negation of the proposition from two different consistent sub-theories in the logic. Arguments are rejected if they are constructed from inconsistent sub-theories, which ensures that it is not possible to carry out the classical pathological derivation of anything from a contradiction (*ex falso quod libet*).

So, to put it simply, all formulae in LA have two parts and are written *arg : formula*, where *formula* is an object-level proposition in minimal logic, and *arg* is a meta-level expression which describes how *formula* is justified, by providing an abstract representation of its proof in minimal logic.

LA is, formally, a close relative of FORA. Chapter 6 in particular shows how FORA argument structures can be viewed as abstractions of proof-steps in an object level logic.

There are two principal differences. Firstly, in this thesis, the meta-level argument representations are more abstract than those used in LA and are not tied to any particular object-level language or proof system. The main claim of this thesis is that this is a useful feature of the work described here and that the greater level of abstraction enables a much higher degree of flexibility in manipulating and using arguments in various ways. In other words, I argue that it is useful to totally decouple the meta-level representation from its object-level counterpart, in order to achieve independence from any particular object-level representation language or inference mechanism.

The second difference between FORA and LA is that the purpose behind the development of LA is to enable arguments to be used as a qualitative way of reasoning about *uncertainty*. The reason for articulating arguments for and against a proposition is to enable decisions to be made as to whether it can be confidently concluded or should be rejected. The arguments are used to provide qualitative uncertainty measures [Elvang-Gøransson *et al* 93], [Krause et al 94], which enable apparant conflicts to be resolved. By contrast, the research involved in developing FORA has not been particularly concerned with issues of uncertainty, rather the principal purpose behind its development has been to provide an account of reasoning about conflict, and disagreement, and allowing exploration of multiple points of view. A central tenet is that no attempt is made to resolve conflicts nor 'weigh up' the arguments for competing opinions.

Recently the ICRF have promoted arguments to a more significant level, describing them in [Krause *et al* 95b] as *first class objects* which are themselves reasoned about, at the meta-level. The StAR program produces reports of the level of risk of chemical substances, by generating arguments for and against them being dangerous carcinogens. It does this using a rule-based system based on an expert system for toxicity prediction, called DEREK, which reasons about the presence in the chemical substance of molecular structures known to indicate risks. These predictions are qualified by also including information about how the substance will be administered and other factors about the patient, which may reduce or increase the level of risk. For example, a substance administered orally could be metabolised into a more dangerous substance, or alternatively it may move swiftly through the body and be excreted before causing dangerous side-effects. Thus StAR can combine arguments for carcinogenity based on chemical analysis, with other arguments or counter-arguments based on an understanding of human physiology. In order to produce useful risk reports, they address ways in which such collections of arguments can be

evaluated and presented to a user. Their intention is 'that the risk characterisation should be transparent to the recipient, and he or she acts as final arbiter'. For this to be feasible the arguments presented to the user need to be both comprehensive and clear. The linguistic uncertainty descriptors mentioned earlier [Elvang-Gøransson *et al* 93] are used to support this.

This line of research indicates a real need for a more abstract approach to reasoning about arguments and independence from any one particular object-level representation language.

2.2.2.3 AI implementations of Toulmin's practical reasoning

[Freeman & Farley 92] describes a formal theory of argumentation, closely based on Toulmin's claim-warrant-backing structures, which they have used to implement an argumentation model [Freeman & Farley 93]. They draw the usual distinction between an argument as a structured entity, for explaining the grounds for a claim, and an argument as a dialectical process between two agents who disagree. Like the ICRF, they are largely concerned with representing and reasoning about the uncertainty of claims disputed in the argument process.

They produce a taxonomy of possible qualifiers (in Toulmin's sense) which are closely related to the qualitative uncertainty factors derived by the ICRF. Their argument structures are based on *Toulmin Argument Units* (taus) which is their name for the argument representations described in section 2.2.1.2. Their contribution is to provide a recursive algorithm for the exhaustive generation of arguments for and against a claim. Further algorithms characterise a dialectic argumentation process as a series of moves, achieved in a game-theoretic turn-taking manner.

Unlike the ICRF, but like much other AI work on argumentation, however, they do not address the issue of how to achieve the basic representation of argument steps which is required for these algorithms to be used. Formalisation of the argument steps is taken as the given from which they proceed. Their argument process is also strictly two-sided. Hence as an implementation of Toulmin's basic theory this is useful research and successfully brings Toulmin into the AI arena. However, it does not yet address the question of how to use argumentation to support the construction of knowledge based systems in controversial domains. They also do not address any examples of significant complexity, limiting themselves to standard (toy) examples such as Toulmin's 'British citizenship' example, Poole's 'Republican-quaker-hawk-dove' example, the standard default reasoning ('Penguins-birds-flying') example and Pearl's 'Rain-sprinkler-wet-grass' causal reasoning example.

Another implementation of Toulmin-style argumentation is [Bench-Capon et al 91],

which produces *explanations* of the reasoning carried out by logic programs using Toulmin's argument structures as the template for explanations. In this system the object-level reasoning is carried out by a logic program interpreter, which requires that the logic program's predicates are annotated with information describing which part of a Toulmin-style argument it represents (datum, qualifier etc). This, naturally, facilitates the explanation of the proof. The description in chapter 6 of how object-level proofs can be parsed to provide meta-level arguments describing the reasoning carried out in the proof, is a direct contribution to the problem tackled by Bench-Capon, and provides a more elegant method for carrying out this abstraction step without requiring modification of the object-level representation.

2.2.2.4 AI applied to legal argumentation

Given the importance of rigorous argumentation in courts of law, and the necessity of dealing with two inconsistent points of view (the prosecutor and the defendant) in the adversarial legal practice common to many countries, it comes as no surprise that the legal domain is a rich source of interesting argumentation-based AI applications. One of the most famous is the representation of the British Nationality Act as a logic program [Kowalski and Sergot 90], the purpose of which is to automatically construct an argument for or against a candidate's eligibility for citizenship.

[Sartor 93] provides techniques whereby a logic program can assess arguments for and against a legal decision and, by providing orderings of the premises used by both sides, resolve the conflict between them. This work is particularly interesting from the point of view of this thesis as it provides rigorous formal definitions of the concepts of argument and counter-argument which can be stated using FORA's language (see section 5.3). The argument-ordering devised by Sartor is similar to the qualitative uncertainty measures developed in [Elvang-Gøransson *et al* 93].

Another legal AI application which involves argumentation is [Yang et al 9?]. This work applies case-based reasoning techniques to the problem of formalising Scottish building regulations. Planning permission for buildings involves a combination of building rules (which are deliberately open ended) and precedent, *ie:* reference to previous buildings which have been granted planning permission which are deemed similar to the case in question. These cases allow the rather vague rules to be interpreted in particular cases, by the construction of IBIS-like argument structures (see section 2.2.3.2) about the similarity or dissimilarity of the current case to previous cases, and the relevance of various aspects of the regulations.

[Loui *et al* 93] is another example of work which seeks to allow legal reasoning to integrate both arguments based on general policies or rationales, and arguments based on particular precedents. These are combined in a version of defeasible logic in order

to provide syntactic methods for determining which of several arguments is the strongest. An interesting avenue this research has taken is to explore, in cases where there is no obvious winning argument, ways of nonetheless drawing conclusions. One such is the use of notional 'resources' which can be applied to arguments, to limit the time spent on arguing for various options. [Loui 92] describes strategies for this, such as allocating resources to generating counter-arguments when a strong argument exists, so that after a finite amount of resource has been used up, it can be concluded that a position is upheld despite a maximal attempt to undermine it. A similar approach to providing heuristics for allocating resources to arguing is suggested in [Sillince 94]. This idea is reflected in the 'arguer' program described in chapter 4 of this thesis, which given a strong argument for one position, suggests ways in which a user of the program can generate counter-arguments to it. Again, though, this thesis differs from Loui's work in approach by not aiming for computer-based conflict resolution, merely exploration, and by not being tied to a particular object-level logic.

2.2.2.5 AI applied to environmental argumentation

The environmental domain is only recently gaining the interest it deserves from AI researchers and relatively few examples exist which attempt to use AI techniques to represent environmental or ecological arguments. One is [Robertson et al 91] in which ecological models are treated as arguments for particular patterns of relations occurring in an ecosystem. These arguments are represented as logic programs, which can be run as simulations. Some more recent research [Robertson and Goldsborough 94] has taken a more direct approach to environmental argumentation, representing as logic programs the arguments for and against the selection of sites for wildlife reserves. Again this is restricted to a particular object-level representation.

A much more informal approach to capturing the structure of agricultural extension documents has been developed in [Beck & Watson 92], but this has little to say about argument structures, being concerned more with the 'molecular structure' of sentences than their 'ecological' relationships with other statements.

[Haggith 95a] and [Haggith 95b] discuss the use of FORA for representing various arguments for and against the granting of oil production leases in the environmentally sensitive coastal zone off Alaska. More details of this example are given in chapter 4 of this document.

2.2.3 Less Formal Approaches to Argumentation

2.2.3.1 Linguistic Perspectives on Argumentation

Linguists are interested in argumentation because a pattern of argument is often reflected in the structure of a piece of text. There is thus a large body of research which assesses how such structures affect the coherence of text. One particularly influential example is the Rhetorical Structure Theory (RST) of [Mann & Thompson 87a], [Mann et al 89]. This theory provides a set of structures which can be used to analyse (or 'mark-up') text. These structures are binary relations between pieces of text, or sets of pieces of text, so for example, one piece of text might be linked to another which provides background information. The first would be called the 'nucleus', the second would be called the 'satellite' and the link between them would be called 'background'.

Although the idea of marking up textual structures to reveal the form of the underlying argument is interesting, and RST has inspired considerable research (particularly in the computational linguistics area, eg: [Knott & Dale 93], [Daradoumis 93]), it has some significant problems. In particular, it is not suitable as a representation framework for handling arguments from conflicting points of view, for the following reasons.

Firstly, in the various papers on RST a large set of binary relations is proposed. These relations are not given any formal interpretation, and it is assumed that people will use them in a uniform fashion. There is a laissez-faire attitude toward the generation of new relations, and as a result there is a proliferation of relations with no clear statement of their intended meaning. This problem is linked to a lack of indication of how Mann *et al* intend to use their theory, other than as a way of organising linguistic analysis and spotting when texts do not 'hang together'. There is no indication in particular that their theory could be used for automated analysis, and as a result their definitions are semantically unclear. This makes it difficult to compare their structures with alternative ways of representing arguments, for example, the method described in [Fisher 88]. A somewhat more formal approach to the representation of argument structures in rhetorical texts (newspaper editorials) is the OpEd system of [Alvarado 89] which is based on structures they call Argument Units to represent patterns of support and attack relationships between beliefs. However, the underlying belief relationships suffer from the same proliferation of link-types as exemplified by RST.

The second problem is that this is a theory about 'rhetoric', which they claim is comprehensive enough to be used in the complete analysis of over 400 sample texts,

but the assumption here is that these texts are monologues. The focus of this thesis is to support analysis of *debate* and thus the central need is for analysis tools which are not restricted to a single writer or voice. Some of their requirements for text coherence are thus inappropriately restrictive. Having said that, their relation definitions frequently refer to the reader of the text (in addition to the writer), and they are explicit in their theory of text being part of a theory of communication. The problem is that their theory restricts this communication to being one-directional, so there is total asymmetry between reader and writer. This is clearly a big problem, particularly given the central role of disagreement in argumentation. There is no way their theory can be expected to handle this notion effectively within the constrained view of 'rhetoric'. Even their notion of 'antithesis' (normally considered an aspect of dialectic) is treated in [Mann and Thompson 87b] as a style of monologue in which the writer attempts to counter a possible objection to their main claim.

More recently there has been some research done into extending RST to handle dialogues, for example, [Daradoumis 93a&b] and [Fawcett & Davies 92], but though these tackle the limitation of a 'single voice' in RST, the problem of informality remains.

2.2.3.2 Issue-based Information Systems

Another well-known approach to the representation of argument structures is the Issue-based Information System (IBIS) of [Kunz & Rittel 70]. This was developed as a way of structuring political decision making processes, and in particular the development of policies by disparate groups of people, potentially with strong vested-interests, which involves a lot of coordinated debate. The aim was to bring closer together the structure of actual discourse in political organisations, and the way in which the resulting decisions are documented, and 'to stimulate a more scrutinized style of reasoning which more explicitly reveals the arguments'. IBIS has been used by many decision making bodies, from universities, to the World Health Organisation and US government departments. It involves people stating their 'position' on policy suggestions already suggested by other people, and providing arguments either to support their position or to attack others. Although designed as a purely paper-based system, it has led to various implementations of argumentation systems, including [Conklin & Begeman 88].

[Casson & Stone 92] describe an 'expertext' system (combining IBIS style hypertext argument links with expert system facilities) which allows legal regulations to be explored, and redrafted, using high-level relations between sections of text which indicate, for example, that one regulation subsumes another.

An important limitation of the work on IBIS is its informality - the relations are not

logically constrained, nor intended to have a particular formal interpretation, which provides the systems with great flexibility, but limits the amount of reasoning which can be automated using these relations.

2.3 Summary

This chapter has provided a brief survey of literature which addresses the question of disagreement. This survey was divided into two main areas. Firstly, the issue of disagreement itself was addressed in section 2.1 with a survey of the literature about conflict, inconsistency and multiple viewpoints. This section characterised the 'problem' addressed by the research described in the remainder of this dissertation. Secondly, the area of argumentation was surveyed and section 2.2 described approaches to argument and debate in philosophy, linguistics and AI. This section presented the background to the 'solution' adopted in FORA, which is introduced in the next chapter.

Chapter 3

FORA Overview

This chapter gives an overview of the components of FORA, (Framework for Opposition and Reasoning about Arguments). It explains how these components link together, and where to find more detailed descriptions of them in other chapters. Section 3.1 reviews the context and goals of the research which led to FORA's development. Section 3.2 summarises the FORA system. The basis of FORA is a language for representing the structure of arguments (described in chapter 4). FORA is composed of modules for reasoning with arguments represented in the language. Some of these modules illustrate ways of using the language and provide an informal operational semantics for it. Other modules provide ways of mapping between formal representations in FORA and an object-level logic. All the modules are illustrated using a running example which is introduced in section 3.3.

3.1 Motivation

Chapter 2 presented a survey of research in the areas of disagreement and argumentation. It revealed three important things.

Firstly, in AI, the most usual approach to disagreement is to try to resolve it, either before constructing an AI system, or when a conflict is encountered by a reasoning system. The community of AI researchers most tolerant of disagreement includes people who build tools for qualitative reasoning with uncertainty - to them conflict need not necessarily be resolved as it can provide useful information about how confident we should be in certain conclusions. Argumentation is a useful way of generating this information. However, there is very little research attempting to build knowledge bases which represent many differing points of view without attempting to reach an overall consensus, though there is a clear need to do this.

Secondly, most logic-based approaches to inconsistency and disagreement attempt to handle it within the logic (*eg:* paraconsistent logics, the Logic of Inconsistency, Restricted Access logics). The alternative is to handle inconsistency by providing a more abstract layer of representation, at the meta-level, in which to reason *about* the existence of disagreements and arguments for differing points of view. I argued that this is a more coherent approach to argumentation. It is a meta level activity, and methods are needed for representing arguments independently of any particular object-level logic.

Thirdly, amongst the various approaches to representation of arguments, two broad themes can be seen:
(i) Informal argumentation: In most cases, arguments are represented informally, usually as some kind of network structure in which the nodes and links are given informal meanings (*eg:* IBIS, RST, Expertext, Toulmin Argument Units).
(ii) Formal argumentation: In the remaining cases, arguments are represented in terms of a particular formal logic (*eg:* the defeasible logic used by Loui; the ICRF's labelled logic of argumentation, LA; the logic programming language used by Bench-Capon, etc).

Both approaches have strengths and weaknesses. The main strength of the informal approaches is that they are flexible (for example people can express relationships between statements in a relatively unconstrained way), and there is evidence that people find it relatively straight-forward to sketch out connections between argument parts in order to use them. The main problem with informal argumentation is that its lack of clear definitions makes it hard to provide much in the way of computer-based analysis or support, as the formal meaning of the links and nodes are unclear. At best, rules can be written to help a user to navigate the network of sentences, but it is not possible to reason automatically across the links to draw conclusions, because the network creators are not restricted to any particular semantics of the links.

The main strength of the formal approach is that algorithms can be easily written to manipulate formal arguments. The main problem with most logic-based argumentation systems is that they provide a representation of arguments which is tied to a particular logic. The very close relationship between object-level logical representations and argument definitions makes it difficult to compare arguments represented using different object-level languages or to separate out the overall structure of arguments and reason with them independently of the low level details of the argument parts.

My thesis, that there is a need for an argumentation system handling multiple points of view, has therefore been refined so I can now say what characteristics this system should have.

1. The system should not focus on conflict resolution, but must provide facilities for exploration of the content and structure of a debate.

2. It should enable the many useful argumentation structures from the literature (such as counter-arguments, corroborations, rebuttals and undercutting) to be expressed.

3. It should be based on a formal language which does not encourage proliferation of primitive relations without clear meaning.

4. Most importantly, the arguments should be represented at the meta-level in a way which is independent of any particular object-level language, but which can be formally interpreted in terms of object-level logics.

The main evidence I present to show that such a system is *possible* is the description of the implemented FORA system, which has all of the above characteristics, at least to some extent. FORA is summarised in the next section. The argument that such a system is *useful* is that it can handle an important health policy debate as demonstrated by the running example introduced in section 3.3 and used to illustrate all the modules of FORA in the remaining chapters.

3.2 FORA system overview

The remainder of this chapter describes an argumentation framework, FORA, (Framework for Opposition and Reasoning about Arguments) which is intended to combine some of the advantages of the two positions of flexible informality and rigid formalism. It is based on a simple argument representation language, which is independent of any particular object-level knowledge representation language, and therefore more abstract than most other formal argumentation approaches. The argument representation is based on a small set of relations which are formally defined to enable automated reasoning over them to be possible.

Figure 3.1 gives a diagrammatic overview of how the tools in FORA interrelate. They form four distinct groups which each tackle a different use of arguments. Each group of tools is described in a separate chapter. These are summarised in the next four subsections.

3.2.1 The FORA knowledge representation language

In chapter 4, FORA's argument representation language is introduced. The basic syntax and primitives are stated and then used to define a variety of argumentation structures.

The second half of chapter 4 discusses knowledge acquisition from the point of view of FORA. Acquiring many inconsistent points of view, and arguments for them, requires a slightly different approach from standard knowledge acquisition for expert systems. A series of experiments in acquiring inconsistent viewpoints allowed some valuable lessons to be learned and these are discussed in section 4.3. The final section of chapter 4 describes a hypertext mark-up tool for helping to build knowledge bases in FORA starting from text describing various viewpoints. This tool produces output in the form of a knowledge base (or part thereof) which can be interpreted and manipulated by FORA.

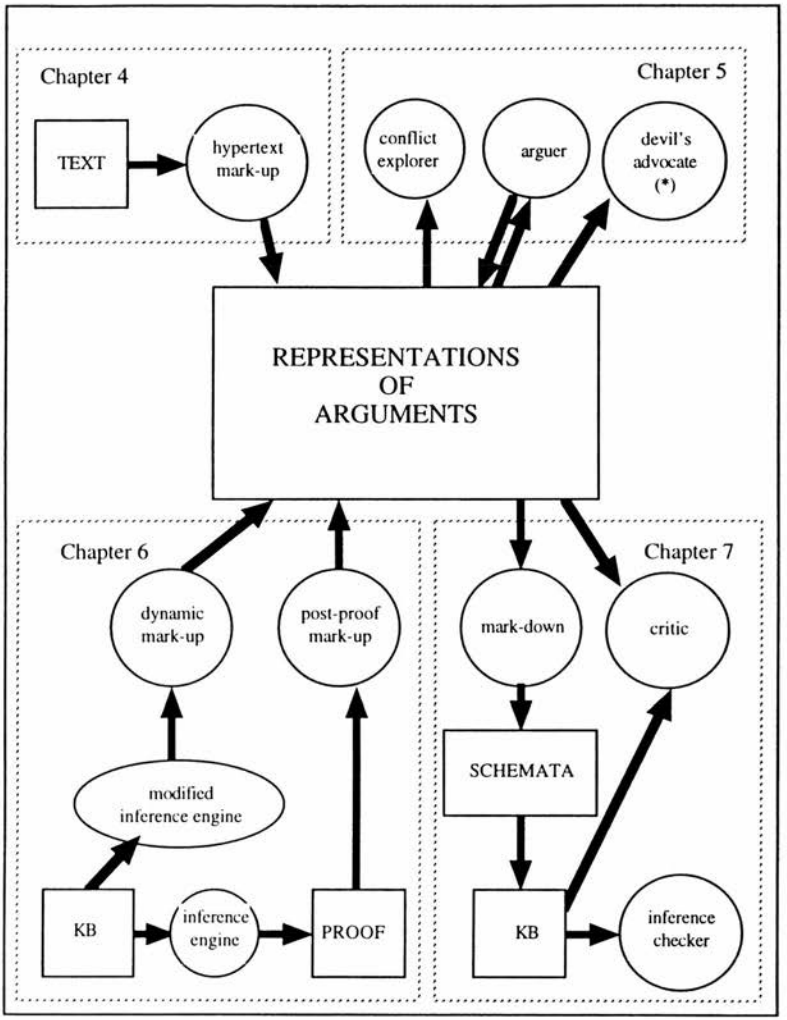


Figure 3.1 : The FORA system.

Rectangles represent static structures such as knowledge bases, libraries of definitions or proofs. Circles represent tools for manipulating or transforming these structures. Arrows indicate flow of information from one or more tools or set of structures to another.

(*) The devil's advocate tool was implemented by an MSc student supervised by the author. All other tools were implemented by the author.

3.2.2 Using knowledge bases in FORA

Having defined arguments in chapter 4, and shown how to construct knowledge bases containing them, chapter 5 describes three tools which allow a user to *explore* them.

1. The first of these is a tool for exploring conflicts in the knowledge base by following the threads of the arguments used to support and oppose conflicting points of view.

2. The second tool is an 'arguer' program which suggests ways in which the user may like to add additional arguments to the knowledge base - it does this by pointing out weak statements which are not argued for, statements which require corroboration, statements which are undisputed and so forth. This is useful for both evaluating the arguments in a knowledge base and as a tool for knowledge base extension.

3. The third tool is a simple tutor called the 'devil's advocate' which encourages a student user to follow an argument for a particular point of view, whilst the system produces counter arguments to point out weak points in the student's argument and to challenge her to keep strengthening it. This tool manipulates arguments represented in the FORA representation language. It was designed to illustrate how FORA could form the basis of debating strategies, and, under my supervision, implementation was carried out by an MSc student [Retalis 95]. It was evaluated by first year Artificial Intelligence students using knowledge bases expressing philosophical arguments in the debate about the 'mind body problem' with promising results [Retalis *et al* 96].

3.2.3 Automating Mark-up

It is awkward if all FORA's knowledge bases have to be constructed from scratch, by marking up text. There are many knowledge bases in existence which can be interpreted as expressing arguments for points of view, and it would be useful if these arguments could be integrated into FORA. There is a significant overhead in marking up such knowledge bases by hand. To support the thesis that meta-level argument representation is useful it is important to provide evidence that the mark-up process can be automated.

Chapter 6 discusses this issue and describes two techniques for automating mark-up of formally represented knowledge bases. These techniques are illustrated by providing

tools which mark-up (or abstract from) object-level knowledge bases represented in first order predicate logic (FOPL).

The first technique involves modifying the inference engine so that it records argument structures as it constructs proofs. The second technique is more elegant and involves parsing completed proofs to infer various levels of argument from them.

3.2.4 Supporting Mark-down

The final issue and group of tools concerns the inverse issue to mark-up, ie: generating an object-level knowledge base from arguments represented in FORA. To provide further support to my thesis that meta level argument representation is useful, chapter 7 shows how it can support knowledge engineers in building knowledge based systems. This is done by using arguments as 'sketches' of proofs or inference chains which we would like an inference engine or theorem prover to be able to carry out. Arguments in this sense are *requirements* for knowledge bases - they are a way of describing the reasoning we need a knowledge based system to carry out, as a first step in building the system.

Argument structures are more abstract than knowledge bases represented at the object-level. Translating from an object-level representation to the meta-level description of it is called 'mark-up', so the reverse process, giving a particular object-level representation which fits a meta-level description, is called 'mark-down'. This is a rather ugly epithet, but it neatly captures how knowledge base specification (mark-down) is to be viewed as the inverse of abstraction (mark-up). Figure 3.2 illustrates this idea.

It is important to notice that mark down cannot normally be automated like mark-up, because it generally requires a user (in this case a knowledge engineer) to add information to the arguments. To understand this it is useful to compare mark-down with mark-up.

Mark-up takes a proof and uses abstraction to strip out the structure of the proof (the argument), ignoring the detail of how the content of the proof is formally represented at the object-level (which is what gives it the 'force' of a proof). So mark-up *removes* information from a proof. Mark-down, as the inverse operation, starts with an argument representing the structure of a proof, and *adds* information about how the steps in the proof will actually be represented at the object-level, in order to give the argument the force of a proof. This information needs to be supplied by a knowledge engineer who knows which object-level language they wish the proof, or inference chain, to be carried out in, and who can provide object-level representations of the propositions used in the argument. The structural information contained in the FORA

argument representation can be used to guide the knowledge engineer and to constrain choices they need to make. Chapter 7 of this document explains how this is done.

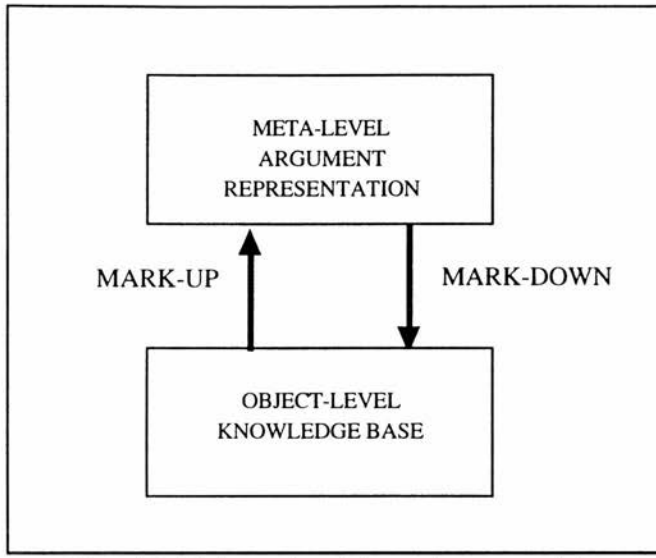


Figure 3.2 : Mark-up and mark-down as inverse translations between object-level and meta-level representations.

A schema-based approach to supporting mark-down is described in chapter 7, along with some useful tools for maintaining knowledge bases implemented in this way. The first of these is an inference checker which verifies that inferences expressed in arguments in FORA can indeed be carried out by the new object-level knowledge base, and if not, points out where they fail. The second is a knowledge base critic which uses counter-arguments to the argument from which the knowledge base is derived to suggest possible weaknesses in it. These tools demonstrate how sketching out arguments in FORA can guide knowledge base construction and maintenance. Like the demonstration of automated mark-up in chapter 6, the object-level language chosen for the examples in chapter 7 is FOPL.

3.2.5 Summary of FORA

In section 3.2 I have introduced the FORA system and given an overview of the next four chapters which provide detailed discussion of the following aspects of the system:

1. FORA's representation language (chapter 4);

2. Construction of knowledge bases in FORA, knowledge acquisition and a hypertext tool for constructing FORA arguments from text (chapter 4);

3. Tools for exploring, using and reasoning about arguments in FORA (chapter 5);

4. Automating mark up of existing formal knowledge bases or logical proofs as arguments in FORA (chapter 6);

5. Supporting mark-down of arguments to help knowledge engineers interpret arguments as requirements for object level reasoning which can support the construction and maintenance of object-level knowledge bases (chapter 7);

6. The operational semantics of FORA (chapters 4 and 5) and definitions of FORA structures in relation to first order predicate logic proofs (chapters 6 and 7).

3.3 Introduction to the running example

To illustrate the technical discussions in the remainder of this document an example will be used of an important real world debate about the health risks posed by a group of chemicals called aflatoxins. Some other examples will occasionally appear as they were used to test some of the tools and techniques, but for clarity each chapter includes an illustration of the ideas presented there in terms of this running example which will be referred to from now on as 'the aflatoxin debate' [Rodricks 92].

3.3.1 The aflatoxin debate

Sometimes stored peanuts (ground nuts) go mouldy. If the mould is *Aspergillas flavus*, sometimes it produces chemical substances called aflatoxins [UK Government 92]. There is concern that aflatoxins might cause cancer in humans [Eaton & Groupman 94]. They have been shown in laboratory experiments to cause liver disease including liver cancer in rats, mice, ferrets, guinea-pigs, monkeys, sheep, ducks, hens and rainbow trout [Rodricks 92]. Aflatoxins are also produced by moulds on cotton, corn and other nuts. The aflatoxins are not retained in the oils of these foods, but they are found in the raw food, and in meal made from them [MAFF 82]. If livestock and poultry are fed this meal then the aflatoxins can show up in their milk, eggs and meat [Mohi Eldin 88]. Animals have died due to liver disease after eating contaminated meal, (notably, a large number of turkeys whose death in the 1960s led to the discovery of aflatoxins), but to date there are no human deaths which can be directly attributed to aflatoxins though there are suspiciously high levels of liver cancers in some parts of the world where it is very likely that there could have been aflatoxin contamination of food as a cause [McDonald 76], [ICRISAT 87].

Aflatoxins have been the focus of controversy in the USA [Rodricks 92], particularly within the Food and Drink Administration (FDA), who produced a policy that aflatoxin levels should be restricted to their minimal detectable level. In the 1960s this level was 30 parts per billion (ppb). Due to improvements in analysis methods, several years later this was set at 20ppb (parts per billion). Not everyone agrees with the FDA's policy, which is based on the 'no threshold' hypothesis that certain carcinogenic chemicals may have no level below which they are inactive. As is typical of issues which are socially important, and scientifically uncertain, the aflatoxin issue has generated debate, and conflict. The examples in chapter 4 and 5 of this document will illustrate how these arguments and conflicts can be represented and reasoned about using FORA.

Subsequent further improvements in laboratory techniques mean that we can now detect aflatoxins (by their fluorescence) at levels as low as 1 part per billion (1ppb) [Rodricks 92]. When the detection level dropped from 30ppb to 20ppb, the FDA revised their 'acceptable level' accordingly. However, although the detectable level has dropped radically, the FDA's policy level has not. Rodricks explains this as follows :

> *".. a large fraction of the peanut butter produced by even the most technically advanced manufacturers would fail to meet a 1ppb limit, and it was also apparent that other foods - corn meal and certain other corn products, and certain varieties of nuts (especially brazils and pistachios) - would also fail the 1ppb test pretty frequently. The economic impact of a 20ppb limit was not great. The impact of a 1ppb limit could be very large for these industries" [Rodricks 92].*

To apply the principle of 'minimum detectable level' (the 'analytical detection limit' as Rodricks calls it) would mean the destruction of huge amounts of contaminated food. In order to justify action like this, policy makers need a high level of scientific certainty of the human cancer risks of such food. At the present time, the industrial inconvenience and economic impact win out over the risk to human health and the aflatoxin limit rests at 20ppb, at least 20 times the minimum detectable level.

Changes in available evidence like this can result in knowledge bases representing the current 'received wisdom' needing to be modified. This example of improvements in detection sensitivity will be used in chapter 7 to illustrate a knowledge base critic which suggests modifications to a knowledge base in the light of this new counter-argument to the acceptability of a 20ppb limit.

In the prologue of his book, Rodricks asks whether the FDA's position is scientifically defensible. He presents two arguments, each representing a different point of view in the debate.

> *"(1) Yes. The FDA clearly did the right thing, and perhaps did not go far enough. Aflatoxins are surely potent cancer-causing agents in animals. We*

don't have significant human data, but this is very hard to get and we shouldn't wait for it before we institute controls. We know from much study that animal testing gives a reliable indication of human risk. We also know that cancer-causing chemicals are a special breed of toxicants - they can threaten health at any level of intake. We should therefore eliminate human exposure to such agents whenever we can, and, at the least reduce exposure to the lowest possible level whenever we're not sure how to eliminate it.

(2) No. The FDA went too far. Aflatoxins can indeed cause liver toxicity in animals and are also carcinogenic. But they produce these adverse effects only at levels far above the limit FDA set. We should ensure some safety margin to protect humans, but 20ppb is unnecessarily low and the policy that there is no safe level is not supported by scientific studies. Indeed, it's not even certain that aflatoxins represent a cancer risk to humans because animal testing is not known to be a reliable predictor of human risk. Moreover, the carcinogenic potency of aflatoxins varies greatly among the several animal species in which they have been tested. Human evidence that aflatoxins cause cancer is unsubstantiated. There's no sound scientific basis for FDA's position." [Rodricks 92]

3.3.2 Why choose this example?

The aflatoxin debate was chosen as the running example for the following reasons.

1. Much of the development work on FORA was carried out using examples from the environmental domain, including controversies surrounding the greenhouse effect, disagreements in agroforestry, arguments about sheep management in Scotland, and disputes over the environmental impacts of oil production in Alaska. The aflatoxin example is from the medical and public policy domain and therefore enabled an evaluation of the domain independence of FORA.

2. The aflatoxin example is used by the Imperial Cancer Research Fund whose logic of argumentation has been identified as a close relative of FORA. Use of the same example clarifies comparisons with their work. The aflatoxin debate is presented as a 'canonical example' for investigations of argument-based risk assessment in an unpublished technical report [Fox 94] specifically to facilitate comparisons between their work and other approaches to argumentation.

3. The aflatoxin example concerns a real public policy debate and has important medical and social implications as it is about carcinogenicity.

3.3.3 The Imperial Cancer Research Fund's approach to formalising the aflatoxin debate

As already mentioned, the aflatoxin debate has been analysed by researchers at the Imperial Cancer Research Fund (ICRF), and comparisons with their work is one of

the ways in which the contribution of this thesis will be evaluated. So, their approach to formalising the debate is described and some preliminary critical remarks are made here as background to the comparisons which appear later.

In [Fox 94], John Fox discusses how to generate a knowledge base representing two of the conflicting views about the cancer risk of aflatoxins. The approach which Fox takes to constructing a knowledge base about aflatoxins has two stages. First, the arguments both for and against the FDA's policy are articulated at a high level of abstraction, using English language sentences and diagrams representing the relationships between these sentences. Secondly, this high level representation is used to drive the formalisation of the content of the arguments into a formal language (the Logic of Argumentation, LA) which can be reasoned with using the Argumentation Theorem Prover (ATP) [Krause *et al* 95a].

Fox's approach to formalising the aflatoxin debate in [Fox 94] is as follows. First, the arguments are subdivided into the following 11 propositions :

1. Aflatoxins are potent cancer causing agents in animals.
2. Not known whether aflatoxins are potent cancer causing agents in humans.
3. Animal testing is a reliable indicator of human risk.
4. Assume that carcinogens can act at any level of exposure.
5. Must minimise exposure to the lowest possible level.
6. Minimum detectable level is 20ppb.
7. Aflatoxins have proven adverse effects only at levels >> 20ppb.
8. No scientific evidence that there is no safe level of exposure.
9. Animal testing not known to predict human cancer risk.
10. Carcinogenic potency of aflatoxins varies greatly across species.
11. Human evidence that aflatoxins cause cancer is unsubstantiated.

The first six statements are used in the 'case for' the FDA policy, the other five in the 'case against'. The overall argument structure of the debate is represented in Figure 3.3.

Fox then goes on to produce a formalisation in LA which captures some of the relationships shown in the diagram. This is done by stating first general rules, then instantiations of these rules, and also of facts, which together enable inference to be carried out automatically at the object-level. The representation shown in *Fig 3.3* is a guide in this process, but it is not used in any formal sense to provide constraints on the selection and instantiation of appropriate rules.

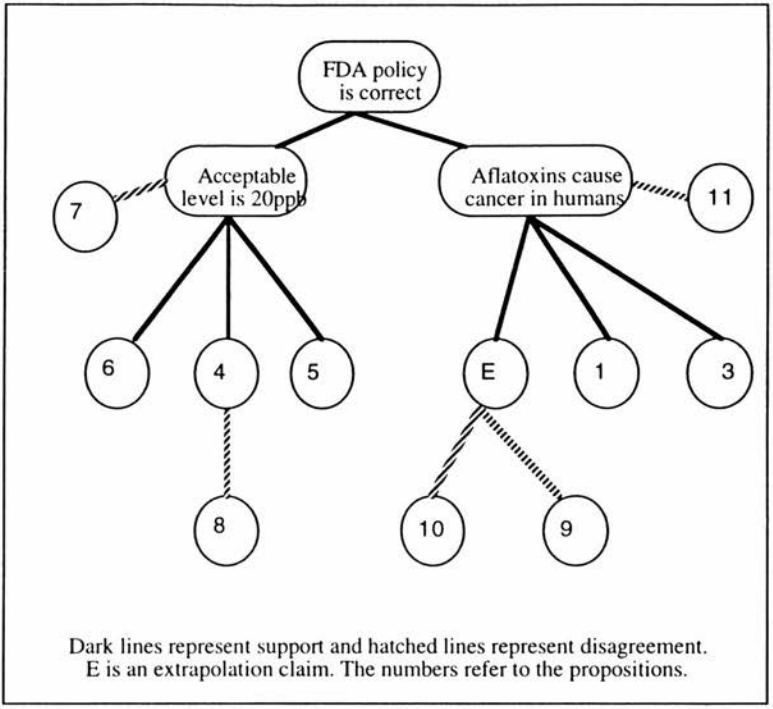


Figure 3.3 : The aflatoxin debate as structured by Fox.

The ICRF's approach to the aflatoxin debate is interesting for several reasons :

1. It takes into account both sides of the argument, not attempting to resolve the conflict.

2. It starts with a rough 'sketch' of the argument, describing the structure of relationships between the statements in it, before worrying about how the detail of the formal representation will work out. In other words, the meta-level problem is tackled first.

3. This rough sketch is declared to be useful in deciding how to formalise the argument, *ie:* the meta-level representation supports the object-level decisions.

However, there are also problems with the ICRF's approach. These include :

1. The two sides of the argument are not given equal weight - the argument for the

FDA policy is 'central' (see figure 3.3) and the structure of the *con* argument is given only relative to the *pro* argument. It would be more elegant, and would give an unbiased view of the debate, if both sides of the debate were represented in the same way.

2. The object-level formalisation which Fox produces includes some argumentation knowledge, for example, the existence or lack of corroboration is represented as an object-level predicate whereas this is actually an argumentation concept which can be more cleanly and generally reasoned about at the meta-level (see section 5.3.1).

3. The argument structure is purely diagrammatic, and only indicates rather than defines the structure. LA produces formal definitions of the arguments only after the object-level formalisation has occurred, effectively by abstraction from (in my terminology, 'mark-up' of) the inference chains. The formal representation of the arguments is thus dependent on the use of a particular object level logic.

4. It is not clear *how* the structure of the argument is used to guide the formalisation, and in particular there is no suggestion that a computer-based tool could help to guide the formalisation.

Deciding how knowledge will be formalised is crucially important when building a knowledge base as it is this which controls what inferences will be possible. Fox makes crucial choices in deciding on particular predicate names, the number of arguments these predicates take, the selection of premises and conclusions in the implications, and so forth, as he formalises the statements in the minimal logic underlying LA. It is these choices which then determine exactly which arguments will be abstracted by LA and represented at the meta-level as labels of propositions. The piece of the jig-saw which is missing from Fox's account, is how the diagrammatic structure of his argument representation relates to the final formal argument structure generated by LA.

The remainder of this document advocates taking a formal approach to the meta-level argument structures from the start, and argues that these structures are then useful tools in guiding knowledge base construction and formalisation. The formal articulation of both sides of the debate at a high level of abstraction enables a knowledge engineer to see the logical weaknesses in an argument before committing herself to a particular logical representation. These formal structures can be used to guide and verify object level formalisation and tools which use these formal representations can show how knowledge bases may be vulnerable to attack, or extendible, by taking into account related argument structures.

3.4 Summary

This chapter has given an overview of the FORA system. The FORA system is discussed in four parts, in the next four chapters :

(i) its representation language and tools for knowledge acquisition,

(ii) tools for manipulating arguments,

(iii) tools for marking-up or abstracting from object-level knowledge bases, and

(iv) tools for marking-down or constructing knowledge bases.

The final section of this chapter introduced 'the aflatoxin debate', a controversy about the safe level of a set of carcinogens called aflatoxins which occur commonly in food. This debate is used as a running example to illustrate the FORA argumentation system throughout the remainder of this document.

Chapter 4

Knowledge Representation in FORA

This chapter describes FORA's language for articulating disagreements and arguments, and discusses how to construct knowledge bases in it - addressing both knowledge acquisition and representation. A hypertext tool to support construction of knowledge bases in FORA is described, and illustrated with the running example of the aflatoxin debate.

First, in section 4.1, the language is defined by stating the syntactic definitions of constructs allowable within it, and an example of their use is given in section 4.2. Section 4.3 discusses knowledge acquisition from multiple, conflicting sources, and section 4.4 describes a hypertext tool for marking-up text (such as transcripts of interviews) into the FORA language. Section 4.5 returns to the debate about aflatoxins and shows how the hypertext tool can be used to mark up texts from [Rodricks 94] into a representation in FORA. Section 4.6 is a summary.

4.1 The FORA language

In this section, the language for representing arguments in FORA is presented, by defining objects, terms, formulae and rules.

4.1.1 Meta-level objects

The language consists of three syntactic categories of meta-level *objects* :

1. *Proposition Names* (*Ai ...Aj*, *Bi....Bj*, or quoted text strings, denote proposition names, which are atomic objects)

2. *Arguments* (<= is the argument constructor - see section 4.1.2)

3. *Sets* ($\cup$ (union), $\in$ (member) are the set constructors)

The language also uses the relation names given in section 4.1.3, together with the usual logical connectives.

4.1.2 Term constructor definitions

The *terms* of the meta-language are constructed by three methods.

1. All *proposition names* are terms

2. *Sets of proposition names* are terms which are constructed by the usual set construction :

{} is a set (the null, or empty set)
If T is a term, then {T} is a set.
If S_1 is the set $\{T_1...T_i\}$ and S_2 is the set $\{T_j...T_n\}$
then $S_1 \cup S_2$ is the set $\{T_1...T_i, T_j...T_n\}$

3. *Arguments* are terms which are are constructed recursively. There are two parts of the definition, the base case and the recursive case:

Base case : *If P is a proposition name and S is a set of proposition names then P <= S is an argument.*

P is called the *conclusion* of the argument, and the elements of S are the *premises*. The <= operator indicates that the conclusion can be drawn from the premises.

Recursive case : *If P, Q are proposition names, S, R are sets such that P <= S, Q <= R are arguments, and S = S1 ∪ {Q}, then P <= (S1 ∪ {Q <= R}) is an argument.*

The recursive case extends an argument for *P*, by taking one of the premises in the argument, *Q*, and replacing it by the *argument* for *Q*. The argument for *P* thus has an argument for *Q nested* inside it. Other premises in the argument for *P*, and/or premises in the argument for *Q* could be replaced by arguments in the same way, forming a tree structure directly analogous to a proof tree.

Example : Given two arguments as follows :

A1 <= {B1, B2}, and *B2 <= {C1, C2, C3}*

we can form the argument which has the second nested inside the first :

A1 <= {B1, B2 <= {C1, C2, C3}}

4.1.3 Formula constructor definitions

The *well formed formulae* (wffs) of the language are defined as follows:

1. Relations between objects are wffs. There are four primitive binary relations, interpreted as follows (also see section 4.1.6 for some relationships between them):

a. *equivalent(P, Q)* : *P* and *Q* are names of propositions which mean the same, in terms of some function for transforming propositions in one object-level language into propositions in another language, or in terms of an object-level equivalence relation, or according to the opinion of some person marking up text.

b. *disagree(P,Q)* : *P* and *Q* are the names of propositions which disagree or express a conflict, according to some object-level notion such as negation, or temporal inconsistency, or according to the opinion of some person marking up text.

c. *elaboration(P, S)* : P is a proposition name and *S* is a set of names of propositions which elaborate upon or or give more details about *P*, according to multiple instantiations at the object-level, or according to the opinion of some person marking up text.

d. *justification(P, S)* : *P* is a proposition name and *S* is a set of names of propositions which are a justification of *P* according to some object-level justification procedure such as modus ponens, abduction, or some other justification such as source or evidence, in the opinion of some person marking up text.

2. Wffs can also be constructed in the usual way using the logical connectives.

These four basic relations were selected for two reasons. Firstly, they were adequate for capturing the kinds of argument structures acquired during preliminary knowledge acquisition exercises, for more details of which see section 4.3. Secondly, they were a 'core' of the many relations used to represent texts in Rhetorical Structure Theory (see section 2.2.3.1) and used elsewhere in the argumentation literature. Given the problem of proliferation of such relations in RST, I drew up a list of about 30 commonly used relations and eliminated all those which are very closely related (*eg:* questioning, doubting, and opposition were all deemed to be very similar in intention to 'disagreement'). I then eliminated all those which could be interpreted as a complex of simpler relations (so, for example, counter-argument was eliminated as being definable in terms of argument and disagreement). The definitions of these more complex argumentation structures in terms of this basic set of relations (plus the argument definition just given) are presented in section 5.3.

It should be noted that the distinction between elaboration and justification is not absolutely clear cut. It is intended to enable a distinction to be made between a step in an argument which has the force of a deductive step (justification) and a weaker step such as giving an example, or explaining a term (elaboration). Particular formal interpretations of these relations can of course be given in terms of an object-level representation language which make the meaning of the two relations unambiguous with respect to that language (see chapters 6 and 7).

4.1.4 Term destructor definitions

It is necessary to be able to know the contents of FORA terms once constructed, so we also need destructor definitions. There are two basic destructor relations for looking inside argument and set terms.

1. *Set membership*, $\in$, which is a two-place, infix relation.

Definition : $P \in S$ iff

$$\exists S1.\ S = S1 \cup \{P\}$$

2. *Argument*, which is a three-place relation.

Definition : *argument(A, P, S) iff*

A is the argument $P <= S$.

P is called the *conclusion* and S is called the *top level set.*

Note that if the argument is nested then the top level set is a set of proposition names and arguments.

3. *Premise_set*, which is a three-place relation.

Definition : *premise_set(A, P, S) iff*

A is the argument $P <= Set$, and

S is the set of proposition names occurring within Set.

S is called the *premise set.*

Note that the premise set is the result of 'flattening' the argument.

Three other useful predicates can be defined using the three destructors.

1. *Support* is a binary relation which holds between two propositions, the second of which occurs in an argument for the first.

Definition : *support(P,Q) holds for the proposition names P and Q iff*

$$\exists A.\ premise\_set(A, P, S)\ \&\ Q \in S.$$

2. *Support* is a three-place relation extending support/2 with an extra place holder for the argument in which the support relation holds.

Definition : *support(P,Q, A) holds for an argument A,*

and proposition names P & Q iff

$$premise\_set(A, P,S)\ \&\ Q \in S.$$

3. *Support _set* is a binary relation which holds between a proposition name and the complete set of proposition names which support it. Using the analogy between an argument and a proof tree, a support for P is a node of the proof tree for P and the support_set is the complete set of nodes on all proof trees for P.

Definition : For a proposition name P and a set of names S,
support_set(P, S) iff
$\forall Q$ *support(P, Q)* $\rightarrow Q \in S$.

4.1.5 Types of argument

Now some further definitions of types of arguments can be given. This is necessary as the argument definition given earlier gives the form of an argument, but does not restrict which proposition names can be used as premises for which conclusions. The meta level relations can be used to capture that the <= operator indicates the conclusion following from the premises.

The strongest way of doing this is to restrict the argument operator to those sets of premises and conclusions which are linked by the justification relation. These arguments are thus called *strong* arguments. Note that this still does not mean that they are as logically forceful as proofs, because the justification relation need not necessarily be used only to represent logically valid deduction.

Definition : If P is a proposition name, and S is a set of proposition names,
and justification(P,S)
then P <= S is a strong argument
If P, Q are proposition names and R,S are sets such that
P <= S, Q <= R are strong arguments, and S = S1 $\cup$ *{Q},*
then P <= (S1 $\cup$ *{Q <= R}) is a strong argument.*

Strong arguments can be indicated by:

$P <=_S S$

Arguments using justifications, elaborations and equivalences are weaker and are called *hybrid* arguments.

Definition : If P is a proposition name, S is a set of proposition names and
justification(P,S), or elaboration(P, S) or S ={E} and equivalent(P,E),
then P <= S is a hybrid argument.
If P, Q are proposition names and R,S are sets such that
P <= S, Q <= R are hybrid arguments, and S = S1$\cup${*Q},*
then P <= (S1 $\cup$ *{Q <= R}) is a hybrid argument.*

Hybrid arguments can be indicated by:

$$P <=_H S$$

Finally, arguments are *complete* if there are no premises which can be replaced by arguments - this only happens when all the leaves on the tree are arguments of the form P <= { }. This indicates that there is no argument for P, in other words, that P is an assumption.

Definition : An argument is complete iff
either its set of premises is empty,
or all members of the set of premises are complete arguments.

4.1.6 Meta-level rules

It is now necessary to state some rules which constrain the interpretation and use of the four primitive relations introduced in section 4.1.3. The FORA system includes the following set of general rules.

1. Disagreement is symmetrical.
   $disagree(P, Q) \rightarrow disagree(Q, P)$

2. Equivalence is symmetrical.
   $equivalent(P, Q) \rightarrow equivalent(Q, P)$

3. Disagreement and equivalence are mutually exclusive.
   $disagree(P, Q) \rightarrow \neg(equivalent(P,Q))$
   $equivalent(P,Q) \rightarrow \neg(disagree(P, Q))$

4. Disagreement is 'contagious' within equivalence sets.
   $(disagree(P,Q) \ \& \ equivalent(Q, R)) \rightarrow disagree(P,R)$

5. Elaborations apply to all members of equivalence sets.
   $(equivalent(P,Q) \ \& \ elaboration(P, S)) \rightarrow elaboration(Q, S)$

6. Elaborations must not disagree with the statement they elaborate upon.
   $elaboration(P, S) \rightarrow (\forall Q.Q \in S \rightarrow \neg(disagree(P, Q)))$

7. Justifications must not disagree with the statement they justify.
   $justification(P, S) \rightarrow (\forall Q.Q \in S \rightarrow \neg(disagree(P, Q)))$

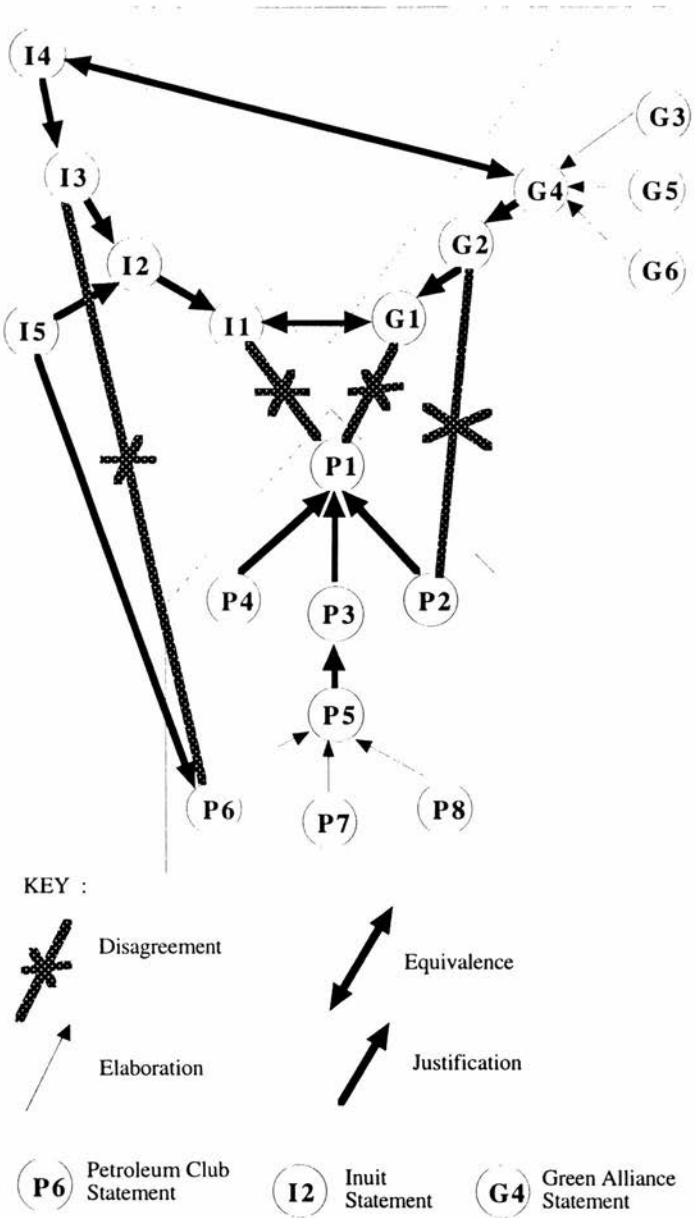


Figure 4.1 : The Alaskan Oil Production Lease controversy

4.2 Example

For the purposes of illustration, here is an example taken from [Dryzek 83] concerning the views of various parties in Alaska about the granting of state leases to drill for oil off the Alaskan coastline. Three parties are represented :

P: government bodies and industries interested in encouraging self-sufficiency in US oil production (Dryzek dubs these interests the 'Petroleum Club');

I: the indigenous Inuit population which is internally divided about oil production, particularly on the issue of employment; and

G: the 'Green Alliance' of environmental pressure groups and government bodies concerned with conservation of natural resources and concerned by the harm caused by oil production.

The Petroleum Club argues in favour of the granting of new oil production leases, on the basis that satisfactory environmental impact assessments have been carried out and new leases will increase the rate of oil production. This, they argue, is a good thing as increased oil production generates state revenue by increasing employment, increasing the government's share of oil company profits and providing taxable revenue. The Green Alliance disagree that new leases should be granted, because they don't agree that the environmental impact statement is satisfactory. They argue that increased oil production will reduce fish and marine mammal populations by causing pollution, which kills them, and disturbance, which causes them to migrate to other regions of ocean. The Inuit are in agreement with some of the Green Alliance views, and on the whole agree that new leases should not be granted. Their argument is that they will impact negatively on their indigenous culture by shifting work patterns away from the traditional fishing and hunting into working for the oil and service industries. The shift away from fishing and hunting is justified by use of the Green Alliance's argument that fish and marine mammal populations will be damaged by more oil production. Their prediction of increased employment in the oil industry corroborates the Petroleum Club's claim that this will be an outcome of new leases, but interestingly the Inuit draw a different conclusion from this piece of evidence.

This short summary of the conflict is inevitably a gross simplification of the situation which has involved years of legal disputes and political activity, involving many people who do not necessarily all fall neatly into the three camps as characterised by Dryzek. For example, he is at particular pains to point out that there are a significant number of Inuit people who believe that new jobs in the oil industry would be a positive thing and that more oil leases should be granted. However, for the purposes

of demonstrating how a real debate can be articulated in FORA, this example will suffice.

Figure 4.1 shows a diagrammatic representation of the three way conflict. The propositions representing the opinions are given below, together with FORA relations and arguments representing the views of each group, followed by some relations and arguments showing how multiple viewpoints can be combined.

THE PETROLEUM CLUB

P1 : A lease for oil production should be granted.
P2 : An adequate environmental impact statement has been produced.
P3 : Maximum rate of production of oil should be attained.
P4 : The lease for oil production will increase rate of production.
P5 : Oil production provides state revenue.
P6 : Oil production increases employment.
P7 : Oil companies share profits with the state.
P8 : Profits from oil production and employment are taxable.

The following three relations capture the main steps in the argument:
'The Petroleum Club argues in favour of the granting of new oil production leases, on the basis that satisfactory environmental impact assessments have been carried out and new leases will increase the rate of oil production.'

justification(P1, {P2, P3, P4}).

'This, they argue, is a good thing as increased oil production generates state revenue ...'

justification(P3, {P5}).

'...by increasing employment, increasing the government's share of oil company profits and providing taxable revenue.'

elaboration(P5, {P6, P7, P8}).

The overall argument is as follows :

A1 : P1 <= {P2, P3 <={P5 <={P6, P7, P8}}, P4}

THE INUIT PEOPLE

I1 : A lease for oil production should not be granted.
I2 : Oil production is damaging to indigenous culture.
I3 : Oil production reduces work in fishing and hunting.
I4 : Oil production damages fish and whale stocks.
I5 : Oil production produces work in the oil and service industries.

'The Inuit are in agreement with some of the Green Alliance views, and on the whole

agree that new leases should not be granted. Their argument is that they will impact negatively on their indigenous culture.. '

justification(I1, {I2}).

'...by shifting work patterns away from the traditional fishing and hunting into working for the oil and service industries.'

justification(I2, {I3, I5}).

'The shift away from fishing and hunting is justified by .. the .. argument that fish and marine mammal populations will be damaged by more oil production.'

justification(I3, {I4}).

The overall argument is as follows:

A2: I1 <= {I2 <= {I3 <= { I4}, I5}}

(Note that this is a *strong* argument)

THE GREEN ALLIANCE

G1 : A lease for oil production should not be granted.
G2 : The environmental impact statement is not satisfactory.
G3 : Oil production causes pollution and disturbance.
G4 : Oil production reduces populations of fish and marine mammals.
G5 : Pollution kills fish and mammals.
G6 : Disturbance causes migration of animal populations.

'The Green Alliance disagree that new leases should be granted, because they don't agree that the environmental impact statement is satisfactory.'

justification(G1, {G2}).

'They argue that increased oil production will reduce fish and marine mammal populations ...'

justification(G2, {G4}).

'...by causing pollution, which kills them, and disturbance, which causes them to migrate to other regions of ocean.'

elaboration(G4, {G3, G5, G6}).

The overall argument is as follows :

A3 : G1 <= {G2 <= {G4 <= {G3, G5, G6}}}

INTER-AGENT RELATIONS

The following relations state links *between* the views of the three parties.

disagree(P1, I1).
disagree(P2, G2).
equivalent(G1, I1).
equivalent(I4, G4).
justification(P6, {I5}).
(Note support by the Inuit for the Petroleum club's argument)
disagree(I3, P6).
(Note the disagreement too)

MULTI-AGENT HYBRID ARGUMENTS

Having identified an equivalence between G4 and I4 the Inuit argument , A2, can now be extended to include this equivalence and also the elaboration of G4, resulting in the following hybrid argument.

A4 : I1 <= {I2 <= {I3 <= { I4 <= {G4<= {G3, G5, G6}}}, I5}}

Similarly, the new justification of statement P6 by the Inuit claim I5, results in a hybrid extension to the Petroleum Club's argument, A1, as follows :

A5 : P1 <= {P2, P3 <={P5 <={P6 <= {I5}, P7, P8}}, P4}

This example has demonstrated how the FORA language can be used to capture the structure and interrelationships of arguments in a debate between several points of view. It provides a summary of the key points of contention and shows how the positions of the three parties relate to each other. It has done this by building up structures out of the four basic relations. In chapter 5 some richer structures will be defined which can further demonstrate the range of concepts from debates which can be characterised in FORA, and which can provide insight into the overall structure of a complex set of arguments.

The benefit of representing debates in this way is that they can then be reasoned with automatically, providing support for extending the debate, integrating it with existing formal representations of knowledge and using it as the basis of more fine grained logical representations. Such support is provided by the reasoning tools described in the remaining chapters of this document. First, though, we need to address some of the practicalities of how knowledge can be acquired from multiple conflicting points of view, and represented in FORA's language. This is the subject of the remainder of this chapter.

4.3 Knowledge acquisition from multiple conflicting viewpoints

This section discusses the acquisition of many people's knowledge. It describes a pilot study in which various knowledge acquisition techniques were used with varying success. The problems encountered are discussed as they enable a set of guidelines to be developed for acquiring conflicting knowledge. The main issues covered in this section are : the level of interaction of the researcher with the informant, the breadth of topics in interviews, recording details about the informants, the acquisition of controversial opinion and analysis of interview transcripts.

Conventional knowledge acquisition techniques tend to assume that the knowledge being acquired is a consistent body of expertise in some domain (this is a dominant assumption in the standard texts such as [Hart 86], [Kidd 87], [Gaines & Boose 88], [Wielinga 90]). Such techniques sometimes even assume that only one expert will be involved and where multiple experts are consulted, the techniques usually include methods for resolving any conflicts of opinion under the assumption that one expert will know best [Moore & Miles 91], or that conflicts simply suggest that a further level of detail is required to distinguish between two different situations in which different conclusions should be drawn (see for example [Boose 86], [Politakis & Weiss 80], [Trice & Davis 89], [Easterbrook 91])

In acquiring knowledge to represent in FORA, I was deliberately setting out to acquire multiple conflicting points of view and so the standard conflict-avoiding or conflict-eliminating knowledge acquisition techniques were quite inappropriate. It became clear that acquisition of arguments from a range of different viewpoints was an interesting research issue in itself. To draw some conclusions about appropriate knowledge acquisition techniques, it was necessary to acquire some samples of conflicting points of view and then evaluate how this was achieved. This section describes three pilot experiments using different acquisition methods. By comparison of the results, some conclusions are reached about how knowledge acquisition can be approached to optimise the level of controversy and to produce transcripts containing clear arguments.

The underlying goal of the experiments was to acquire and represent people's opinions in a formal manner in such a way as to make explicit the structure of their arguments. A parallel, methodological aim was to minimise the intervention of the researcher, to ensure that the representations were as true to the statements of the informants as possible and to minimise the distortion of these statements by the researcher. There are tensions between these two objectives (formalisation is bound to involve the researcher's intervention at some stage) so another purpose of this set of experiments was to bring out, and more clearly understand, the interplay between them.

Note that it was not my aim to capture the 'cognitive' structure of people's arguments. These were not psychological experiments. My aim was to observe the arguments people make, to provide some empirical evidence about patterns of disagreement, argument and debate, and to evaluate the suitability of three knowledge acquisition techniques for capturing conflicting viewpoints. The results of each experiment in turn fed into the choice of method for the subsequent experiment.

In these experiments the choice of domain was important as it must be clear to the informant that they need not worry about being 'wrong'. If lay informants were asked to offer opinions on medical diagnoses or toxicological risk assessment, they may justifiably feel ill-informed and lacking in the specialist knowledge required for this task. However, speculation about the causes and influences of the greenhouse effect is not threatening in the same way because it is commonly accepted that even experts in the environmental sciences are undecided or in dispute about many projections. The greenhouse effect has also been widely reported in the media so most people could be expected to know something about it.

4.3.1 The 'Fly on the Wall' Experiment

Aim

To ascertain that lay people would disagree about the greenhouse effect and to acquire their arguments for these different points of view.

Method

The first experiment took a very simple approach to knowledge acquisition. Six informants were interviewed, each interview lasting between ten and twenty minutes. Informants were asked the open-ended question "What do you speculate will be the impact of the greenhouse effect on the climate of Great Britain within your lifetime?" The informants were asked to talk as freely as possible about anything they thought relevant to this issue, and were also prompted to speculate about how food production may be affected by the greenhouse effect. My only other inputs during interviews were occasional prompts of "Why? " to encourage the informants to offer justifications of their opinions. I noted down on paper what was said by the informants, and then attempted to represent these informal language notes in first order predicate logic[1].

Results

As the records of the interviews are notes, some of what was said was inevitably not recorded. This caused problems in verifying the accuracy of the representation. An important result is that some of the innocuous-seeming details of how an informant

[1] Note that at this stage of the research, the FORA language was not finalised and so FOPL was chosen as a formal 'lingua franca'. In fact the FORA language components were decided partly on the basis of the results of these knowledge acquisition experiments.

states their opinion are important indicators of the form of their argument. For example, a set of notes may include the factual statements an informant makes but without the 'little words', like 'therefore', 'for example', and 'however' it can still be unclear how they are using these statements to support their argument (see [Knott & Dale 93] for more on the importance of these 'little words' in articulating argument structures). Likewise it is important to know if the informant was responding to a prompt such as 'why?'. In addition, the vagaries of speech are such that not all necessary details for a formal representation at the level of first order predicate logic are made clear, for example when an informant says that 'the effect of global warming will not be great' this does not make clear their intention as to 'the effect on what?'. With some informants it was possible to check these intentions by getting them to verify the logical representation which resulted. With others, this was not possible either because of a lack of knowledge of logic or just unavailability.

Conclusion

Despite these problems the informants were forthcoming and a useful body of information was obtained which indicated both the existence of disagreements (for example, on the effects of the greenhouse effect on winter weather, some predicting colder winters, some warmer winters), and the ability of informants to argue their position. This was a formative experiment, intended to give some indication of the styles of arguments people use. These arguments include claims (*eg:* that weather will become more extreme) which are then justified in various ways, such as referring to sources of information (*eg:* the media) or by explanation. Often people make use of particular examples in their justifications, such as referring to a particular place where something is happening and then generalising from that to a more global conclusion (*eg:* reference to a particular pacific island which is shrinking, to infer a general conclusion about sea-level rise). This pattern of mixing general justifications and elaborations of particular points in the argument by reference to examples was a significant motivation for including both justification and elaboration relations in FORA.

4.3.2 The 'Blue Peter' experiment

Aim

The second experiment attempted to reduce the level of interpretation by the researcher by providing the informants with a framework for producing their statements in a formal representation directly.

Method

Asking lay people to write down their opinions on the greenhouse effect in some form of logic was clearly not feasible, so an intermediate method was sought. Analysis of the results of the first experiment indicated that many of the statements could be

represented using a limited set of predicates and keywords, principally :

1. names of things;

2. causal relationships;

3. statements about the increase and decrease (or change) of attributes such as rainfall, temperature, sea-level and food production;

4. indications of time and spatial location.

A collection of predicate names and keywords was drawn up with the aim that an informant could use them as their vocabulary for expressing their views on the greenhouse effect.

Some means was needed of indicating to the informant the syntax of this language - for example to indicate that cause is a two place predicate, or to show how statements can be qualified by time or place ('The average temperature will increase in Britain over the next 10 years'). The experiment was therefore set up using a collection of cardboard templates with names on, representing predicate names or keywords[2].

The intention was that by borrowing an intuition of fitting together jigsaw pieces to make a picture, the informant would fit these templates together to form statements and chains of reasoning.

Predicates with different arities were given distinctively shaped cards :

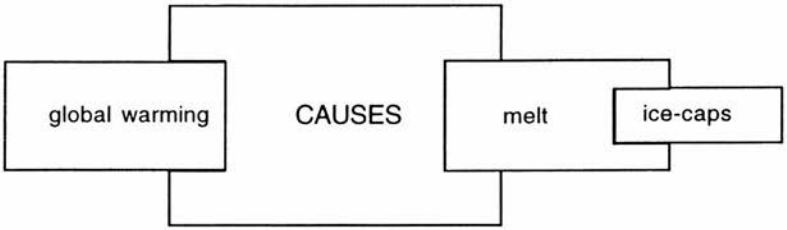
1. Constants (arity 0) were represented as rectangles

2. One-place predicates were represented as rectangles with a slot cut into one of the short sides, to suggest that a constant can be slotted into it. For example the template for 'increase' might have the constant 'rainfall' slotted into it.

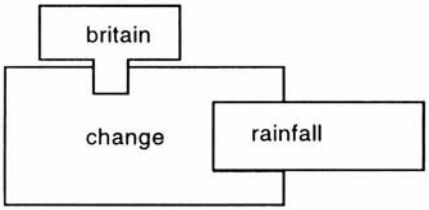
3. Two-place predicates such as 'controls' and the logical connectives 'and', 'or' and 'if ... then', were represented as larger rectangles with a slot cut into both short sides, indicating that constants (or one-place predicates) could be slotted into both indentations. Informants were given connective cards with 'causes' written on them which were interpreted formally as the same as implications. Hence for example, the constant 'global warming' might be slotted into one side of the connective 'causes'

[2]The fact that the templates were cut out of cereal packets and my use of an example ('Here's one I made earlier') led to the affectionate name *The Blue Peter Experiment* after the UK children's TV programme.

while the other side could take the one place predicate 'melt' applied to the constant 'ice-caps'. In this case 'global warming' is in fact being interpreted as a *proposition* such as 'There is global warming'. This is shown in the following diagram.



4. Time and location qualifiers were represented as a small rectangle with a stalk (sign-post shape), the intention being that they could be attached as labels to as many predicates as necessary and act as extra arguments, or informal indicators of modal operators. For example, the one place predicate 'change' might be applied to the constant 'rainfall' and have the place label 'britain'. This would be interpreted as adding an extra argument to 'change', *ie:* indicating *change(rainfall, britain)*, as in the following diagram.



It became clear when setting up this experiment that this visual representation of the logical features of the intended language was limited and contained ambiguities. In particular, nesting of predicates and connectives was problematic; there were no means of representing uncertainty or vagueness of any sort; and the temporal and spatial labelling was under-defined. (This was in fact deliberate as it was hoped to avoid building in too many assumptions about any temporal and/or modal components of the object-level logical language at this stage).

A further problem was that the named templates (for example, 'ice-caps' and 'sea-

level') would act as prompts and therefore influence the statements which the informant would make, thereby breaching the first knowledge acquisition guideline. The intention was thus to require the informant to ask the researcher for templates as they wanted them, rather than browsing through a collection of cues. This, although intended as a safeguard against building expectations about subject-matter into the experiment, in fact caused considerable problems as it forced the researcher to be too closely involved with the use of templates.

Results

This experiment was only carried out with two informants, one of whom was familiar with predicate logic, one of whom was not. Although construction of a few statements was achieved by both of these people, severe problems were encountered.

1. Considerable introductory explanation was required by the researcher which had not been necessary for the first experiment and this resulted in the informants starting with preconceived ideas. In particular, illustrative examples, such as those given above to describe the syntax rules, provide topic cues such as the relevance of ice-caps, rainfall and so on. This problem could have been eliminated by using a second set of templates, about a different domain, for introductory purposes. However, this would not have alleviated the remaining problems.

2. Most of the interesting reasoning, justification of statements and explanations by the informants was not recorded in template form.

3. Both informants frequently asked "Is this right?" or "Is this finished?", indicating that they felt that the cardboard templates suggested to them a puzzle or game with an objective or solution for which they should be aiming (like a jigsaw puzzle). This was in direct contrast to the free expression of opinion sought.

4. The restriction of syntax rules was difficult for the informants to grasp. In the case of the informant familiar with predicate logic this was because the experiment used such a restrictive syntax and contained ambiguities. In the case of the other informant it was because of the unfamiliarity of logic. In particular, the informants' syntax errors caused a problem because asking the informant to correct their statement only served to emphasise further the erroneous impression of 'right and wrong answers'.

5. The logistics of fitting together cardboard shapes, even on a large table, meant that informants were reluctant to join statements together. As a result most of what was recorded were rather fragmented comments. Much of the informants' reasoning was expressed as asides while they attempted the exercise. For example, one informant constructed the sentence "Global warming will cause insect plagues in

Britain", and verbally reeled off a string of justifications which he refused to represent in template form, because of their speculative nature, which did not merit the effort of using templates to express. The implication of this was that few controversial or less-than-certain statements would be represented and disagreements would thereby be less likely to occur.

Conclusions

Tape-recordings of the informants' comments while using the templates would provide more useful information than the statements created during the exercise. Notes taken during one of the experiments substantiate this. An informant produced a sophisticated chain of reasoning to justify his claim that global warming will not cause sea-levels to rise. This reasoning involved considering the relative densities of water and ice at different temperatures, the distribution of ice between the Arctic and Antarctic, the proportion of this ice currently displacing sea water, and so forth. Although the informant could express this information eloquently in English, he said he didn't know how to start expressing it 'in cardboard'.

It was thus concluded that this technique built in too many of the researcher's assumptions about the intelligibility of logical syntax. A formalism which appears quite natural to those trained in logic and logic programming can in fact be quite threatening to people with no logical training. The informants' comments of being 'unsure' and 'confused' led to the conclusion that the involvement of formal logic in knowledge acquisition activities with such informants would severely impede their progress.

Discussion of the first two experiments

The first experiment was a better model for knowledge acquisition for this project than the second, but the problems of verification observed in it need to be addressed. These problems were minimised by the following knowledge acquisition plan.

1. Interviews should be tape- or video- recorded (to provide a record for transcription and verification of the statements formally represented).

2. The relevant material from interviews should be transcribed.

3. Attempts should be made to focus on the chains of reasoning and arguments the informants use.

4. Formal representation of the arguments should be left until later.

4.3.3 The 'Transcript' Experiment

Method

Based on the conclusions drawn from the two pilot knowledge acquisition exercises, a further set of interviews were carried out, asking lay people about their views on the greenhouse effect. These were unstructured interviews lasting between 15 minutes and half an hour. Eight interviews were carried out, the results of which form the basis of a knowledge base together with some extra scripts taken from the media. None of the informants for these interviews were experts on the greenhouse effect, or environmental scientists of any sort. Three were male, five female. Five were British, one was from the US, one from Greece and one from Malaysia. All lived in Edinburgh at the time of the interviews. Four were students, four were not.

Each interview proceeded as follows. First the informant was given a sheet entitled "Your views on the greenhouse effect" which gave them some background information and gave them some idea of what they were required to do. They were asked to read this. Once they had done so, I asked them if I might start the tape recorder. They sometimes had questions at this stage. I started recording. I began the interview by asking them "to predict the impact of the Greenhouse effect in Great Britain within their lifetime". From then on, I kept quiet as much as possible. With some informants this was easy, with others less so. In one extreme case I spoke as much as the informant, getting monosyllabic replies or silence in response to most questions.

Results

The responses to the initial question about the impact of the greenhouse effect in Britain during the informant's lifetime resulted in a variety of kinds of responses, some concerned with climatic changes, and some with broader issues such as the level of concern about the greenhouse effect within society at large. A considerable body of the predictions are about possible changes to human behaviour which may result from changes to social attitudes. I have been surprised and interested by this, as it suggests that the representational techniques needed to fully express the contents of these views should handle not only factual statements about the physical world, but also propositions expressing levels of belief in statements by various parties. So, there are first-order statements of the form 'If sea-levels rise then East Anglia will flood', but also higher level statements of the form 'If the government believes that sea-levels will rise, then power will be devolved from London, and Scotland will attain independence'.

The discussions about climate ranged from local weather forecasts and previous experiences, predictions about global climate patters, winds and sea-currents, references to the ice-caps, glaciers, atmospheric behaviour and particular gases (notably CO_2, CO, CFCs, nitrous and sulphurous gases, ozone and methane) and radiation (ultra-violet and long-wave). Other environmental topics included flooding, infestations, pollution, desertification and deforestation. Social issues included health (eg: skin cancer), transport, education and the media. Political issues included the government, Europe, the West's relationship to the developing world and economics.

In some interviews I encouraged the informants to go off on some tangent (for example, I encouraged one informant to talk about bees, and another to talk about the government). This ensured that the breadth of content from the interviews was quite wide. Two of the informants from non-British countries spontaneously spoke about their own countries, and the impact of the Greenhouse effect there.

In every interview I asked the informant to predict what would be the impact of the greenhouse effect on food production, which proved an interesting question in that the range of answers was very broad. In many cases, as hoped, the informants used the predictions they had already made about climate, the environment or society to justify their claims about changes to farming or food consumption.

I also took cuttings from the press to include in the analysis, as they express views in clear conflict with some of what was said in interviews. These were about the beneficial effects of CO2 on food production and the impact of the greenhouse effect on volcanoes and subterranean faults.

Analysis

[Ericsson & Simon 84] suggest a structure for protocol analysis. The steps involved are

1. transcription,
2. establishing syntax and vocabulary for the representation (*ie:* the language described in section 4.1),
3. segmentation of the transcript into self-contained chunks (propositions),
4. coding of segments, (*ie:* formal representation).

In place of coding, in order to achieve a FORA knowledge base, *linking* is required, which is the process of adding relationships or links between segments.

The first stage of analysis is transcription. As already noted, some information (for example, pauses) is lost. I retained as much of the actual content of people's speech as possible (so the transcripts are littered with 'I mean' and 'you know'). Some

information was also added, in particular I added punctuation. The transcripts were proof-read and edited once and then the text file ported to the hypertext tool described in the next section (4.4) in order to enable segmentation and linking of propositions. For details of the results see [Haggith 94].

Interviewer Intervention

One of the interesting parts of transcription is to look closely at the intervention by the interviewer. On the whole I have intervened much more than I thought I had. The majority of my interjections significantly influence what the informant says afterwards. Most of what I asked was of the form "Why is that?" or "What is the connection there?" to try to encourage people to elaborate on a prediction they have made, or to provide a justification or a statement of cause. Most people seemed to respond well to this sort of prompt. Another common prompt I used when the informant had totally dried up, was to pick up on something they said earlier and ask them to be more specific, or give another similar case, so for example in one interview I said 'Earlier you said that rainfall is going to increase, will that be all year round?', and in another I said 'So you have predicted hurricanes and smog, do you have any other predictions about the weather?'

I have not transcribed pauses (except where they are exceptionally long) however, a significant proportion of interjections were the result of pauses when I felt the informant needed prompting. As a result of such intervention it is possible to set in place a 'question-and-answer' routine, in which the informant behaves as if responding to a questionnaire and does not talk freely. It is, however, sometimes awkward to leave the informant sitting silently, apparently not knowing what to say, and using gestures and body language to indicate that they want you to speak. To remain quiet in this situation can lead to very pronounced and awkward silences and an air of psychoanalytic seriousness which can seem rather threatening to the informant. I tried as much as possible to give the impression of 'having a chat', in which I said rather less than I normally would!

Ericsson and Simon's book on Protocol Analysis [Ericsson & Simon 84] includes an appendix containing 'some practical advice and information on how to elicit various types of verbal reports under standard conditions', which includes the advice that the researcher should be out of the field of view of the informant, preferably sitting behind them ('like behind the couch'), and restricting their comments to 'keep talking'. According to these criteria the interviews I have carried out are at fault. However, I believe that imposing these conditions (reminiscent of the techniques of psycho-analysis), would put more pressure on the informant, and lead to a less speculative and free series of exchanges.

Ericsson and Simon have views on the interpretive role of the researcher which are

almost directly at odds with my own. To them, the cardinal sin of an experimenter is to 'intrude' into the interview, and they offer a metric for such intrusion, as follows. 'The number of verbalizations that are social and directed to the experimenter may be used to evaluate how much the experimenter has intruded.' *ibid*, P 376. They do not provide any guidelines as to how many such social verbalisations constitutes an unacceptable level of intrusion.

It seems unrealistic that a non-intrusive, 'God's Eye View' of an informant's beliefs is attainable, even in principle, so I start from the premise that the researcher will be involved in a process of interaction and interpretation. The position of the researcher's chair and level of engagement with the informant thus take on a different importance. I believe that, contrary to Ericsson and Simon's claim, 'social verbalisation' may in fact be seen as signs of the informant feeling relaxed, which may be conducive to talking freely. Their assumption is that intrusion is always detrimental. This opinion is extremely limiting, as it restricts their comments to the tiny subset of verbalisations which are effectively monologues.

> *"Our discussion here is limited to situations where a single person is performing a task, and does not include those where two or more persons are working together, as in reaching a decision...We also ignore the influence of the experimenter on the subject. Two person interaction, each person providing stimuli for the other, presents additional problems we do not wish to deal with here". (ibid P278)*

Clearly, for those of us whose research involves interactive concepts such as debate, dialogue and argument, this is no help at all, other than as an admission of the lack of generality of their approach. For analysis of the information collected here, therefore, alternative techniques needed to be adopted.

I have commented on Ericsson and Simon's view of correct interview methodology for two reasons. Firstly, to try to demonstrate why the 'classic' work on verbal protocols is too narrow a method for knowledge acquisition of multiple viewpoints, and thus to justify why I do not adhere to it. Secondly, because despite its faults, their structure for transcript analysis (transcription, segmentation, encoding) is useful. It seems important to point out that using this structure does not necessarily imply adoption of any other parts of their methodology or of their theoretical position.

4.3.4 Conclusions

The main conclusion drawn from the experiments was that people do disagree with each other and can provide complex arguments for their points of view and that it is possible to record some of these by means of simple interview and transcription processes. It is less useful to ask an informant to represent their knowledge directly in predicate logic (and this causes real problems with informants without logical

training). Even with logically trained informants, the detail of how to represent the content of statements as predicate logic formulae can obscure the overall structure of their argument. The level at which argument structure can be usefully represented thus seems to be more abstract than FOPL.

Here are some guidelines for acquiring knowledge about controversial issues.

1. Provoke controversy by carefully choosing questions which encourage speculation.

2. Do not influence the informants to say things which they might not otherwise have said - it is particularly important to avoid asking leading questions which appear to have a 'correct' answer.

3. Encourage the informants not merely to state their opinion, but also to justify and explain it, and record those justifications as completely as possible. 'Why?' is a useful question to ask.

4. Record what informants have said as accurately and completely as possible, preferably using full transcription as this provides a resource that can be returned to later to trace the pattern of the informant's argument.

5. Do not introduce unnecessary formal tools which may also give an erroneous impression of there being a 'right' way for the informant to contribute their views.

4.4 Mark-up using hypertext

A tool for helping in construction of FORA knowledge bases has been implemented in HyperCard, a hypertext tool for Macintosh computers. This tool supports the segmentation and encoding phases of transcript analysis, as described in the previous section. In other words it lets a user take a piece of text, select propositions from within it (segmentation) and state how they relate to other propositions in terms of the structure of the argument (encoding).

HyperCard works on the basis of small units of information called 'cards' (by analogy with the catalogue card systems in libraries) which are grouped into 'stacks' with similar structure or information content. Each card can include fields for text, buttons and graphics. Clicking the mouse-button over any of these items can cause something to happen, if the object clicked on has a program (called a 'script') associated with it.

The FORA mark-up tool consists of three stacks :

1. *Mark-up*, which contains the text to be segmented.

2. *Propositions*, which contains the segments of text, records the links between propositions or sets, and outputs the knowledge base in Prolog syntax.

3. *Sets*, which contains sets of propositions.

These are now described in turn, and followed by illustrative figures.

Stack 1. Mark-up :

This is the entry point and it consists of a stack of cards which contain the text to be segmented (See Figure 4.2). Any text file can be selected and loaded into the mark-up stack, by clicking the 'read from file' button. Several files of text can be included in the stack, and each card can have unique source information recorded in the top right hand text field. Long pieces of text will be spread out over several cards - arrows indicate how to 'turn the page' backwards and forwards through the text.

The text has two modes - *read* or *segment*. In *read* mode, clicking the mouse on text produces normal hypertext behaviour (*ie:* if you click on part of a marked-up argument, you move to where it is represented). In *segment* mode, clicking the proposition button, the user can select a piece of text with the mouse and make it into a piece of hypertext. The selected text appears in an editable box so that the user can alter it (if the text appearing in the flow of a passage does not make complete sense standing alone, for example because of anaphora such as 'it' which need to be replaced by noun phrases). This edited text becomes a proposition in FORA and is recorded along with its source information in the propositions stack, ready for linking with other propositions.

Stack 2. Propositions :

The propositions stack is where most of the important information and functionality of the tool is based. It contains the propositions (segments of text) selected by the user, and enables linking between them. Each card in the stack (see Figure 4.3) contains a single proposition and a record of all the links from it. These links are to either other propositions or sets. The link types are the four binary relations in the FORA language - *disagree* and *equivalent* (linking proposition to proposition), and *justification* and *elaboration* (linking proposition to set).

The user can look at the current relations by clicking on a relation button and the relations of that type from the current proposition appear in the big text field.

The user can also create a new relation linking from the current proposition by clicking the new relation button and the appropriate relation-type button. A message is then presented asking the user to move (by hypertext links, arrows etc) to the proposition or set they wish to link to, and then hit the 'enter' key which selects that point as the target of the link. They are then returned to the starting proposition and the new relation is recorded on its card. In the case of justifications and elaborations, the target is a set, so the user moves to the set stack and selects an existing set. They may need to create a new set, which they must do before being able to make the link. The details of how this is done are given in the section on the sets stack.

Finally, from the propositions stack the complete FORA knowledge base, consisting of propositions with sources and relations, can be generated in Prolog syntax and saved to a file readable by Prolog to be used by all the other tools which make up FORA. This is done by clicking the 'Dump Code' button and giving a file name for the knowledge base.

Stack 3. Sets :

The sets stack cards each contain a set of propositions, and enable the user to carry out the usual set operations, namely union, intersection, addition of an element, deletion of an element, copying the set and deleting the whole set. This is done by clicking the appropriate button on the card (see Figure 4.4). Addition, unions and intersections are carried out in the same way as linking of propositions. The user clicks a button on the starting point, say they click 'add proposition' on a set containing one proposition. They are given a message to move to the chosen proposition (by hypertext links, arrows etc) and then hit the 'enter' key which selects that point as the target of the link. They are then returned to the set which has the new proposition added to it. A new set can be created by taking the union or intersection of other sets or by selecting a proposition in the propositions stack and making it into a new set by clicking the 'new set' button.

Constraints are imposed on changes and deletions. A set which is the target of a relation cannot be changed or deleted. A proposition which appears in a set or is related to or from another cannot be deleted. So to delete a proposition, first all relations to and from it must be deleted, then any sets in which it occurs must be deleted and only then can it be deleted itself.

Figure 4.2: Mark-up stack

The text here is part of an interview transcript in which an informant (who chose the name Captain James T. Kirk) gave their views on the greenhouse effect.

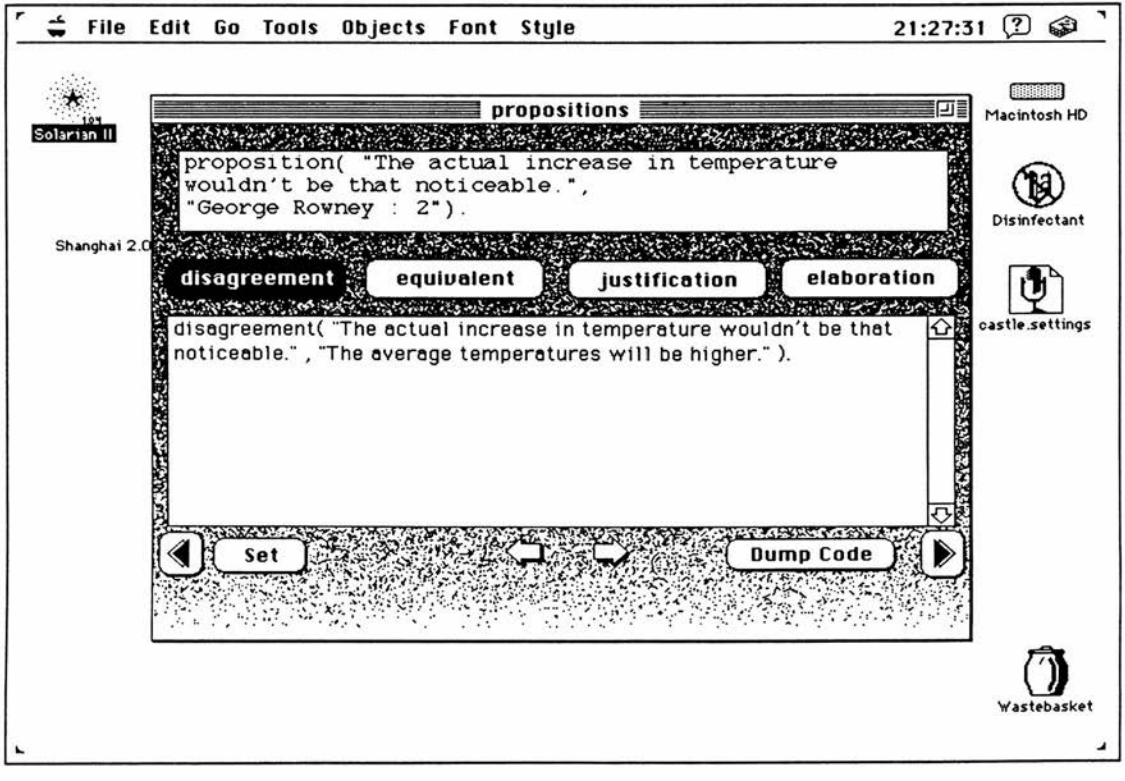


Figure 4.3: Propositions Stack

Figure 4.4 : Sets Stack

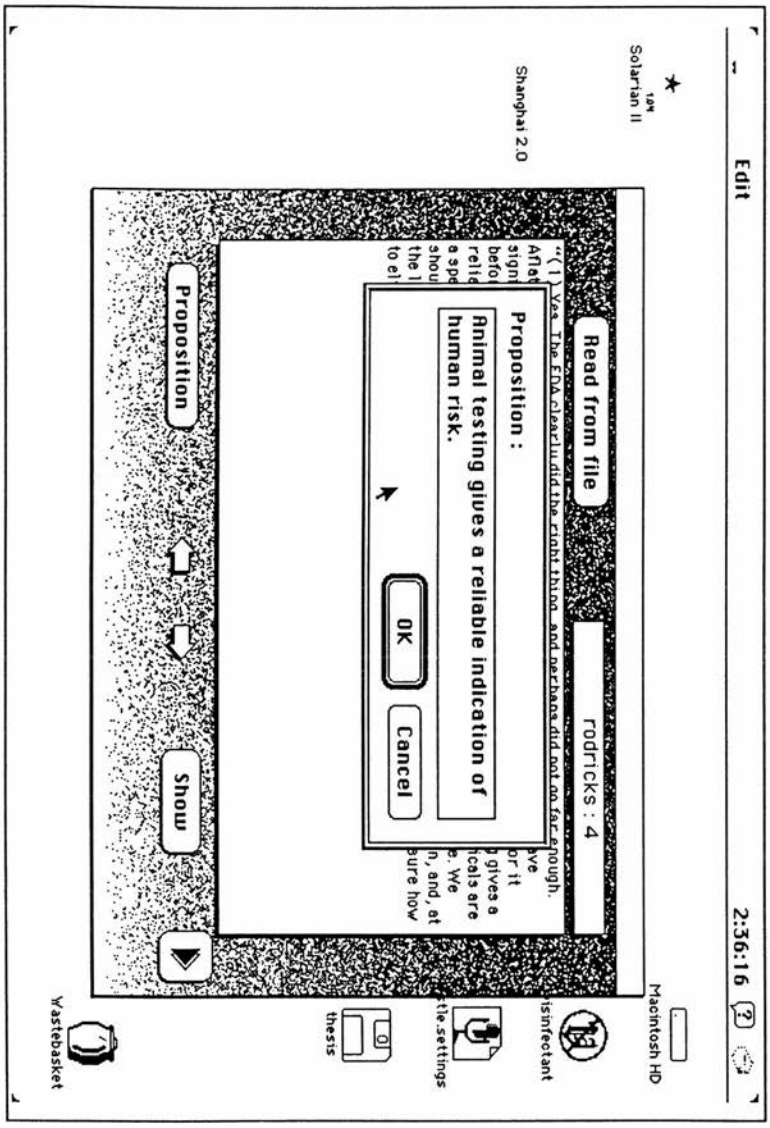


Figure 4.5 : Mark-up of the aflatoxin debate - illustrating segmentation

4.5 Example : Representing the aflatoxin debate in FORA.

To illustrate the construction of a knowledge base in FORA containing the conflicting views about aflatoxins, the text in section 3.3.1 can serve in place of a transcript and be loaded into the HyperCard mark-up tool. This enables segmentation of text, and mark up or linking of segments, to be illustrated and also allows a return to the thread of the running example. The text appears on cards, as illustrated in figure 4.5, and propositions and the arguments for them were marked up using the hypertext tool. The result, after using the tool and finally choosing the 'dump code' button to produce the FORA knowledge base in Prolog-readable syntax, is as follows.

FORA knowledge base

After segmentation the propositions are as follows :

```
proposition('The FDA policy level for aflatoxins should be 20ppb', pro).
proposition('The maximum acceptable level of aflatoxins is 20ppb', pro).
proposition('Aflatoxins cause cancer in humans', pro).
proposition('There is no safe level of aflatoxins', pro).
proposition('The minimum detectable level of aflatoxins is 20ppb', journal).
proposition('Aflatoxins cause cancer in non-human animals', pro).
proposition('Aflatoxins are a kind of chemical', pro).
proposition('Animals are good indicators of the cancer risk of chemicals to humans', pro).
proposition('Aflatoxins cause cancer in non-human animals', pro).
proposition('Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable',
        pro).
proposition('The FDA policy level for aflatoxins should not be 20ppb', con).
proposition('20ppb is not the maximum acceptable level of aflatoxins', con).
proposition('The level of adverse effects of aflatoxins is 200ppb', con).
proposition('200ppb is much greater than 20ppb', con).
proposition('Extrapolation of cancer risk from one species to another is unreliable', con).
proposition('The effect of aflatoxins varies greatly between species', con).
proposition('There is a lack of scientific evidence showing no safe level of aflatoxins', con).
proposition('Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable',
        con).
proposition('There is a lack of scientific evidence showing aflatoxins cause cancer in humans',
        con).
proposition('Aflatoxins cause cancer in animals', journal).
proposition('Aflatoxins cause liver toxicity in animals', journal).
```

The propositions are then linked to show they relate to each other in terms of the arguments. There are lots of justifications, on both sides of the debate.

```
justification('The FDA policy level for aflatoxins should not be 20ppb',
        ['20ppb is not the maximum acceptable level of aflatoxins']).
justification('20ppb is not the maximum acceptable level of aflatoxins',
        ['The minimum detectable level of aflatoxins is 20ppb',
        'The level of adverse effects of aflatoxins is 200ppb', '
        200ppb is much greater than 20ppb']).
```

```
justification('Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable',
        ['Extrapolation of cancer risk from one species to another is unreliable']).
justification('Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable',
        ['The effect of aflatoxins varies greatly between species']).
justification('The FDA policy level for aflatoxins should be 20ppb',
        ['The maximum acceptable level of aflatoxins is 20ppb',
        'Aflatoxins cause cancer in humans']).
justification('The maximum acceptable level of aflatoxins is 20ppb',
        ['There is no safe level of aflatoxins',
        'The minimum detectable level of aflatoxins is 20ppb']).
justification('There is no safe level of aflatoxins',
        ['Aflatoxins cause cancer in humans']).
justification('Aflatoxins cause cancer in humans',
        ['Aflatoxins cause cancer in non-human animals',
        'Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable']).
justification('Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable',
        ['Aflatoxins are a kind of chemical',
        'Animals are good indicators of the cancer risk of chemicals to humans']).
```

The elaboration relation is used here to elaborate on the type of adverse effects caused by aflatoxins.

```
elaboration('The level of adverse effects of aflatoxins is 200ppb',
        ['Aflatoxins cause cancer in animals','Aflatoxins cause liver toxicity in animals']).
```

The following disagreements can be given.

```
disagree('The FDA policy level for aflatoxins should be 20ppb',
        'The FDA policy level for aflatoxins should not be 20ppb').
disagree('The maximum acceptable level of aflatoxins is 20ppb',
        '20ppb is not the maximum acceptable level of aflatoxins').
disagree('Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable',
        'Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable').
disagree('There is no safe level of aflatoxins',
        'There is a lack of scientific evidence showing no safe level of aflatoxins').
disagree('Aflatoxins cause cancer in humans',
        'There is a lack of scientific evidence showing aflatoxins cause cancer in humans').
```

Finally, here is an example of two equivalent statements.

```
equivalent('Aflatoxins cause cancer in non-human animals',
        'Aflatoxins cause cancer in animals').
```

Argument for the FDA policy

The definitions of arguments allow the construction of the following argument for the FDA policy.

The FDA policy level for aflatoxins should be 20ppb <=
The maximum acceptable level of aflatoxins is 20ppb <=
There is no safe level of aflatoxins <=
Aflatoxins cause cancer in humans <=
Aflatoxins cause cancer in non-human animals<= Assumption
Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable<=
Aflatoxins are a kind of chemical<= Assumption
Animals are good indicators of the cancer risk of chemicals to humans <= Assumption
The minimum detectable level of aflatoxins is 20ppb<= Assumption
Aflatoxins cause cancer in humans<=
Aflatoxins cause cancer in non-human animals<= Assumption
Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable<=
Aflatoxins are a kind of chemical<= Assumption
Animals are good indicators of the cancer risk of chemicals to humans<= Assumption

The argument is laid out in this manner (with indentation to show the nesting more clearly than brackets can) by one of the programs described in the next chapter - the 'arguer'. The internal representation is just as defined in section 4.1. This is a *complete, strong argument* (see the definitions in section 4.1). It is *complete* in the FORA sense, because none of its leaf nodes are extendible (from the knowledge base just given) - all the leaf nodes' justifications are empty sets. These are indicated here as 'Assumptions'. The argument is *strong* in the FORA sense because none of the steps in the argument are elaborations or equivalences.

Argument against the FDA policy

There is a *complete, hybrid argument* against the FDA policy as it contains the elaboration, and the equivalence. Note that the use of an equivalence as a support in an argument causes a loop (because if P and Q are equivalent this produces the argument $P <= [Q <= [P <= ....]]$). The hybrid argument against the FDA policy is as follows :

The FDA policy level for aflatoxins should not be 20ppb<=
20ppb is not the maximum acceptable level of aflatoxins<=
The minimum detectable level of aflatoxins is 20ppb<= Assumption
The level of adverse effects of aflatoxins is 200ppb<=
Aflatoxins cause cancer in animals<=
Aflatoxins cause cancer in non-human animals<=
Aflatoxins cause cancer in animals<= Loop
Aflatoxins cause liver toxicity in animals<= Assumption
200ppb is much greater than 20ppb<= Assumption

4.6 Summary

This chapter has introduced the meta-level framework which is used as the representation language within FORA. It is abstract and simple, focussing on argument structures defined in language-independent terms. It is intended to capture the form of arguments as represented in ordinary knowledge representation languages, such as rules, or logics of various types, and it is abstract enough not to presuppose any particular object-level representation language.

The second half of the chapter discussed how to construct knowledge bases using this language. It addressed the general issue of acquiring knowledge from multiple conflicting sources and summarised some experiments in knowledge acquisition. Then a hypertext tool for helping to *mark-up* texts into FORA's argument representation language was described. Finally, the FORA language and mark-up tool were illustrated with the aflatoxins example.

A formal semantics (model theory) for this framework has not been included here. Chapter 5 provides an *operational semantics* for the language: it describe its *meaning* by defining how it is *used*. Chapters 6 and 7 describe how to map between FORA and first order predicate logic and so the reader who is interested in semantics will find there definitions of what kind of logical interpretations are possible for statements in the FORA language. These two chapters define correspondences between the meta-level relations, and object-level structures in particular languages (propositional logic and first order predicate logic respectively). FOPL was chosen as the object-level language for these examples in order that these mappings can be thought of as a possible semantics of FORA. The meaning of the relations in FORA is given by showing their relationship to proof structures in FOPL. However it is important to note that because arguments do not have the logical force of proofs, sometimes an argument may correspond to a set of propositions and inference steps which is not logically valid or sound, for example because assumptions do not hold, or because it uses invalid inference, such as abduction.

It would be possible to provide a formal model theory for the framework, and the logic of [Rescher and Brandom 79] would be a good starting point for this enterprise, providing semantics of the primitive relations in terms of superpositions of possible worlds. For the present, however, the main aim is to investigate the computational and practical properties of the framework, articulating its meaning in terms of its use. The next chapter therefore describes some tools for exploring and using FORA knowledge bases.

Chapter 5

Using FORA

This chapter describes some tools for using FORA knowledge bases. The chapter begins in section 5.1 with a discussion of who FORA is intended to be used by, and what kind of functionality FORA needs to provide for them. Section 5.2 describes the basic implementation of FORA, including the simplest FORA tool for exploring knowledge bases, called *explorer*. Section 5.3 is the core of this chapter, and contains formal definitions of a set of argumentation structures built from the primitives of the FORA language presented in the previous chapter. These structures are used by a tool called *arguer*, described in section 5.4.2, for exploring, modifying and extending arguments in a knowledge base. This section includes an illustration of how the argumentation structures are used to explore and extend the aflatoxin debate. Section 5.4.3 gives a brief description of a 'Devil's Advocate' implemented by an MSc student as an argumentation tutor based on the FORA argumentation structures. This chapter is summarised in section 5.5.

5.1 FORA users and their requirements

My thesis is that formal and abstract representations of arguments are useful. The FORA language presented in the previous chapter enables formal, abstract representation of arguments, and so my task is now to argue that such representations are useful. The first question to answer is 'useful for whom?' In this section I will describe the intended users of FORA and by doing so provide some motivation for the suite of tools which have been implemented in the FORA system.

5.1.1 Users

The primary intended users of FORA are knowledge engineers whose task is to construct, revise and maintain knowledge based systems representing some area of expertise. In particular, my aim has been to support knowledge engineering in *difficult* domains, where by difficult I mean controversial, rapidly changing, partially understood, or fragmented.

The domain of toxicological risk assessment is difficult in all these senses. It is controversial, because there are many different vested interests with conflicting goals who have a stake in risk assessment decisions (for example, food producers who want to minimise the amounts of their produce which must be destroyed due to

contamination by high-risk chemicals, versus medical experts who want to minimise any extra risks of cancer). It is rapidly changing as research is continuously providing new evidence about the risks of chemicals, analytical chemistry is continually improving detection techniques, and the role and distributions of risky chemicals change from day to day. It is only partially understood : there are many areas of toxicology (for example, the precise mechanisms by which chemical substances cause cancer) in which understanding is patchy. It is also fragmented, as assessment of risks involves consideration of many different issues, from chemical transport mechanisms, human exposure patterns, epidemiological evidence, mechanisms of pathology, consumption, metabolism and excretion, evidence from animal experiments, testing of food products, agricultural methods of using chemicals, and so on. No one person can be expert in all the areas which need to be taken into account, and so risk assessment involves integrating knowledge from different specialisms. This is a profoundly difficult task - and it involves, for example, combining very low levels of understanding of levels of actual exposure to aflatoxins through food consumption, with a good understanding of how we metabolise some of the aflatoxin compounds, with a poor understanding of the mechanisms by which some aflatoxins cause disease.

In difficult domains such as these, knowledge engineers need to build knowledge bases with an eye on the future. The knowledge bases are likely to need to be updated and revised in the light of new evidence. Some of this evidence may lower uncertainty levels, but we cannot assume that this is the case. New evidence (like the combination of existing evidence) may well conflict with the contents of current knowledge bases, and so require significant re-engineering of knowledge bases. In addition, knowledge based systems have generally been constructed to support reasoning in a highly specific corner of expertise, but in difficult domains, conclusions from such corners need to be combined with evidence from other corners, or with currently useful rules of thumb from areas which are 'wide open' and where research is only just beginning.

5.1.2 Functionality requirements

FORA is intended to be used by knowledge engineers constructing knowledge based systems which involve:

a) knowledge from multiple sources;
b) conflicts of opinion between sources;
c) a high likelihood of new evidence conflicting with current evidence;
d) a need for regular updating and revision of knowledge bases;
e) a lack of algorithmic decision procedures for reaching conclusions.

I claim that in such cases knowledge engineers need to support argumentation between knowledge sources, and that carrying out this argumentation in a formal, computer-based system such as FORA will help them to build and maintain knowledge bases

representing some of the current viewpoints. In addition, they need to be able to carry out this argumentation in the abstract, independently of any particular knowledge representation language, in order to integrate and debate between viewpoints represented in different languages.

This leads to a requirement for the following functionality :

1. to express arguments;

2. to explore existing arguments;

3. to relate the arguments from one source to those of another;

4. to add new arguments and to challenge existing arguments;

5. to represent the arguments of an existing knowledge source;

6. to criticise views of a knowledge source which are particularly controversial;

7. to construct or modify a knowledge source;

8. to check that a knowledge source actually reasons according to an argument;

9. to identify how new evidence might require changes to a knowledge source.

FORA provides, in prototype form, all of this functionality. The remainder of this chapter describes points 1-4, which are all meta level functionalities. Points 5 and 6 involve handling existing object-level knowledge sources and these are the subject of Chapter 6. Points 7-9 involve constructing and modifying object-level knowledge sources and they are discussed in Chapter 7.

Having identified the intended users of FORA and some of their functionality requirements, I now turn to describing how this functionality has been provided.

5.2 Implementation of FORA

Except for the hypertext mark-up tool described in the previous chapter, FORA is implemented in Prolog. The implementation is modular, consisting of core language components and a toolkit of programs for manipulating knowledge bases expressed in this language.

5.2.1 Implementation of the language

The implementation of the FORA language involves some set handling procedures (in order to treat Prolog lists as sets), and some predicates for construction and destruction of arguments. Meta-level knowledge bases containing propositions and FORA relations are stored as sets of facts. Propositions are represented using the predicate *proposition/2*, in which the first argument is the name of the proposition, and the second represents its source. Relations are represented using the predicates *disagree/2*, *equivalent/2*, *elaboration/2* and *justification/2*.

Argumentation structures such as counter-arguments (see section 5.3) are defined declaratively in terms of their constituent relations, using an object constructor. These definitions are stored in an argumentation structure library which can be altered without needing to change the underlying procedures for object construction.

5.2.2 User interface

The user interface for FORA is basic, consisting mainly of windows displaying text and simple dialogue boxes. Access to knowledge bases and tools for using them is achieved by selecting options from pull-down menus. Knowledge bases constructed by mark-up can be loaded into FORA from files. They can also be constructed from scratch and modified within FORA, by selecting options (such as 'add relation') from a pull-down menu which triggers appropriate user interaction.

5.2.3 Exploration of FORA knowledge bases

FORA allows knowledge from multiple sources to be represented, and is intended to be particularly useful when those views conflict. The intended users of FORA (as identified in section 5.1) will be interested in these conflicts and therefore need to be supported in exploring the controversial issues. A proposition is particularly interesting if it is controversial and in FORA this means that it is in direct disagreement with another proposition.

> *Definition : A proposition, P, is controversial , iff*
> *$\exists Q$ such that disagree(P,Q) or disagree(Q, P).*

In order to help a user to interact with the objects in FORA, the concept of a *conflict set* is useful. This set is a subset of all the proposition names currently loaded, and it is used to represent those propositions which a user is currently interested in. Typically it contains at least two propositions which are in disagreement, plus other statements equivalent to or elaborating on those statements. It can also contain justifications for the propositions which may then generate further points of

disagreement.

The basic knowledge base exploration tool, *explorer*, starts with an initial conflict set which is a pair of propositions which disagree. The conflict set is initialised either as the result of the user selecting a controversial proposition, or a controversial proposition being selected at random. The explorer tool then offers the user various ways of exploring the controversy. This results in the conflict set being added to, or having propositions deleted from it.

The conflict set can be thought of as a window on the controversies within the knowledge base which can move around according to the relations between propositions. Exploration results from altering the conflict set, by either *expanding* it to include new propositions related to those already in the conflict set, or *focussing* it to eliminate those propositions which are not interesting or which have been sufficiently explored. There are various different ways in which expansion and focussing can occur. The following methods have been implemented in the explorer tool to demonstrate the use of all four basic FORA relations.

Expansion of the conflict set
Here are four simple methods for expanding the conflict set.

1. *Adding support* : take a proposition in the conflict set, find a justification relation for it, and add the justifications to the conflict set.

2. *Adding disagreement* . take a proposition in the conflict set, find a disagreement relation involving it, and add the conflicting proposition to the conflict set.

3. *Substituting equivalents*: take a proposition in the conflict set, find another proposition equivalent to it and replace the first by the second in the conflict set.

4. *Substituting elaborations* : take a proposition in the conflict set, find an elaboration of it and replace the proposition with the set of elaborations.

Focussing of the conflict set
Here are two simple methods for focussing the conflict set.

1. *Removal of equivalents* : if the conflict set contains two or more propositions which are equivalent to each other, remove all but one of them.

2. *Removal of uncontroversial propositions* : if any proposition in the conflict set is not in disagreement with any other proposition in the knowledge bases, remove it from the conflict set.

All of these methods of exploration have been implemented in FORA. The current interface shows the user the conflict set expanding and contracting. It can be tentatively concluded that this is a useful technique for revealing the connectivity implicit in a knowledge base, and allowing the user to simulate the 'flow' of an argument or debate.

However, although the conflict set exploration tool is a useful first step in allowing a user to explore FORA knowledge bases, it is limited, primarily because it does not use the *arguments* which were defined in the previous chapter. The definition of arguments provides a much more comprehensive way of showing not merely the individual relations between propositions, but also the way these relations combine into chains of reasoning. Users also need to be provided with some kind of overview of these chains of reasoning held implicitly in the individual relations.

In the next section I return to arguments and show how the basic argument definition can provide a wealth of more complex argumentation patterns which show the overall structure of a knowledge base and which can help a user to assess the weight and coherence of the evidence surrounding a controversial issue.

5.3 Argumentation structures

An important use of FORA involves construction of more complex meta-level structures defined in terms of the primitives. The four relations and the argument definition enable formal definitions to be given for concepts such as counter-argument, rebuttal and corroboration. In this section a set of argumentation structures is defined. They have all been implemented in FORA and they are illustrated being used by one of FORA's tools in section 5.4.

Three of these structures (*rebuttal*, *undercutting* and *counter-argument*) were chosen because they are each used in other work on argumentation. The first, rebuttal, is used in current work on dialectical reasoning by Morten Elvang-Gøransson [Elvang-Gøransson *et al* 93]. The second, undercutting, is from Toulmin's theory of argumentation [Toulmin 58]. The definition of counter-argument is taken from Sartor's work on representation of legal arguments [Sartor 93]. The fact that they can each be characterised in terms of the FORA language helps to demonstrate that FORA is a general purpose system capable of handling standard formal argumentation concepts.

In section 5.3.1, three structures are defined which do not involve disagreement and characterise the support for a proposition and the supporting role played by a proposition in a knowledge base (*corroboration*, *enlargement* and *consequence*). In

section 5.3.2, six structures are defined which capture opposition or conflict within a knowledge base (*rebuttal, undermining, undercutting, counter-argument, target* and *counter-consequence).* Finally, in section 5.3.3, two structures (*context* and *counter-context*) are defined which give a broad brush indication of the supporting and oppositional parts of a knowledge base with respect to a proposition.

The 11 structures defined in sections 5.1.1 - 5.1.3 are representative of how to build more complex patterns of debate from the basic relation and argument definitions. They are not a definitive or complete set as no such a set can be given. There is an arbitrary number of ways in which the relations could be strung together to generate larger structures. The ones that appear here include the most simple - those structures which relate a proposition to an argument by a single relation - plus some other significant structures from the literature. A lesson to be learned from Rhetorical Structure Theory (RST) [Mann *et al* 89] is that depending on the context, people may find various different combinations of relations to be meaningful or important. FORA enables other structures to be defined if a user should wish to use structures other than those provided in the structure library.

The definitions all have a similar form. They are all given as structures relating to a single proposition named P. All but the final two are defined as a single argument. (The final two are sets of propositions.) So, for example, a *rebuttal* of a proposition is an *argument.* It is sometimes useful to describe the relationship between two arguments, and a discussion of my terminological conventions for such descriptions is in section 5.3.4.

The definitions make much use of the argument and set destructors as defined in section 4.1.4, so it is useful to recap the following :

1. In the destructor expression *argument(A, P, S)*, A is an argument, P is its conclusion and S is the set of propositions at the first level of A.

2. In the destructor expression *premise_set(A, P, S)*, S is the set of propositions occurring anywhere in the argument A whose conclusion is P.

3. The meta-level rules in section 4.1.6 allow *equivalent(P,Q)* to be deduced from *equivalent(Q,P)* and likewise for *disagree/2* (*ie:* these relations can be treated as symmetrical).

5.3.1 Arguments for and from a proposition

This section gives definitions of argumentation structures. Each definition is first described in words, and then given a formal definition. I have also sketched a diagram for each to give an informal sense of how the proposition is related to the argument in each case. The argument A is represented in each diagram as a simple tree structure in a box with dotted-line edges. FORA relation-names are indicated as italic labels on double-headed arrows.

A. Corroboration

A *corroboration* for a proposition is defined as an argument for a proposition equivalent to it. In the example in section 4.2, the Inuit's claim that oil production damages fish and whale stocks is corroborated by the Green Alliance's argument about reduction in wildlife populations.

Definition : corroboration(P, A) iff
P is a proposition name and A is an argument, such that
∃Q. equivalent(P, Q) & argument(A, Q, _).

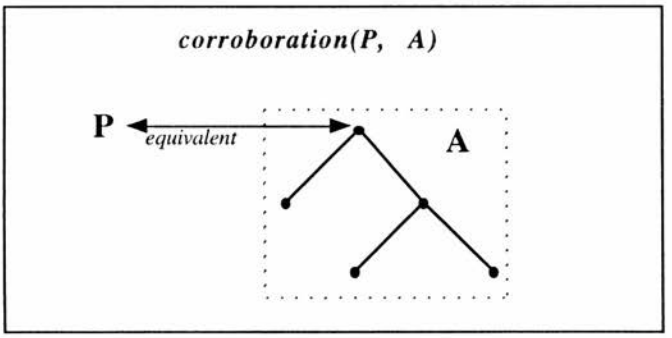


Figure 5.1 : Corroboration

B. Enlargement

An *enlargement* of P is an argument for an elaboration of P. Again, in the example from 4.2, the Inuit argument that the oil industry produces jobs in oil and services, is an enlargement of the claim that oil production brings in state revenue.

Definition : enlargement(P, A) iff
P is a proposition name and A is an argument, such that
$\exists E,S.$ elaboration(P, S) & $E \in S$ & argument(A, E, _).

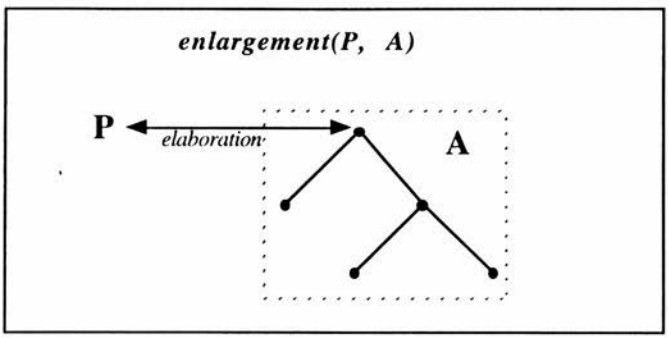


Figure 5.2: Enlargement

C. Consequence

A *consequence* of a proposition name P is defined as an argument in which P is a premise, in other words it is an argument *from* P. The argument that the Inuit's indigenous culture is damaged by oil production is a consequence of the claim that oil production damages fish and whale stocks.

Definition : consequence(P, A) iff
P is a proposition name and A is an argument, such that
$\exists Q,S.$ *premise_set(A, Q, S)* & $P \in S$.

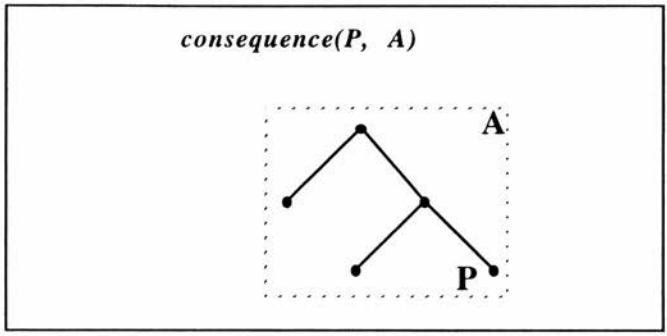


Figure 5.3: Consequence

5.3.2 Opposition about a proposition

D. Rebuttal.

In [Elvang-Gøransson et al 93], *rebuttal* is defined as a relation between arguments whose conclusions disagree. Strictly speaking, they say that the truth of the conclusion of one must imply the falsity of the conclusion of the other. However, this is just one of many possible object level interpretations of 'disagreement' as expressed at the meta-level in FORA. The definition of rebuttal in terms of meta-level disagreement, has, I claim, greater generality than a definition in terms of object-level implication and negation, as it leaves open the possibility that a user could define disagreement in terms of some other object level language; for example, in terms of temporal inconsistency. The FORA definition of rebuttal is as follows :

Definition : rebuttal(P, A) iff
P is a proposition name and A is an argument, such that
$\exists Q,S.$ *disagree(P,Q) & argument(A, Q, S)* .

In other words, a rebuttal of a proposition P is an argument for any proposition which disagrees with P. Two arguments rebut each other if one is a rebuttal of the conclusion of the other. Thus the Petroleum Club's argument that the oil production lease should be granted rebuts, and is rebutted by, the Green Alliance's argument that the lease should not be granted.

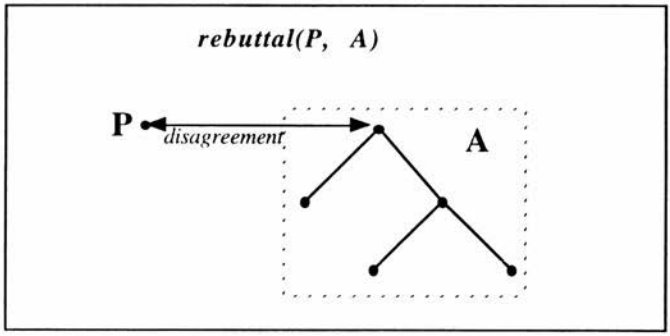


Figure 5.4: Rebuttal

E. Undermining.

An *undermining* of a proposition is defined as an argument for a proposition which disagrees with an elaboration of P or an equivalence with P, in other words, a rebuttal of the elaborations and equivalences of P. The Inuit argument that work is lost in fishing and hunting as a result of oil production undermines the claim of increased state revenue.

Definition : undermining(P, A) iff
P is a proposition name and A is an argument, such that
$\exists$E,S. (*equivalent(P, E)* $\vee$ *elaboration(P, S)*)
& $E \in S$ & *rebuttal(E, A).*

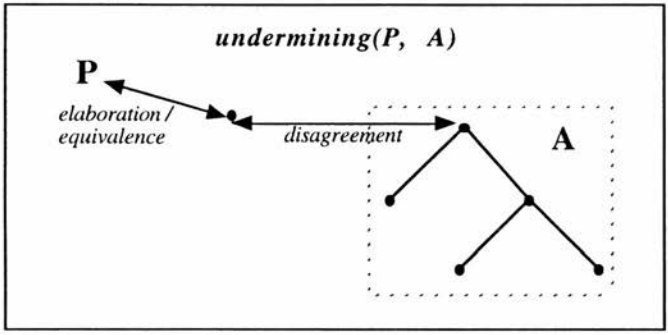


Figure 5.5: Undermining

F. Undercutting.

In [Toulmin 58], another primitive is *undercut*, which means the following : an argument A, with conclusion Q, undercuts B if for some premise P of B, P disagrees with Q. In other words, an argument undercuts another if the first rebuts a premise of the second. In the example, the Green Alliance's argument that the environmental impact statement is unsatisfactory undercuts the Petroleum Club's claim for a lease.

Definition : undercutting(P, A) iff
P is a proposition name and A is an argument, such that
$\exists A1, S, P1, Q,S2.argument(A1, P, S) \ \& \ P1 \in S \ \&$
$disagree(P1, Q) \ \& \ argument(A, Q, S2)$

Alternatively, using the definition of rebuttal above,

Definition : undercutting(P, A) iff
$\exists A1, S, P1. \ argument(A1, P, S) \ \& \ P1 \in S \ \& \ rebuttal(P1, A)$

Note that when the arguments are *strong* (using only justification steps - see section 4.1.5) then undercutting is a notion quite distinct from undermining, as the premise relation in a strong argument represents justification. When *hybrid* arguments are used, the premise relation can be either equivalence, elaboration or justification and in this case, the set of underminings is a subset of the undercuttings.

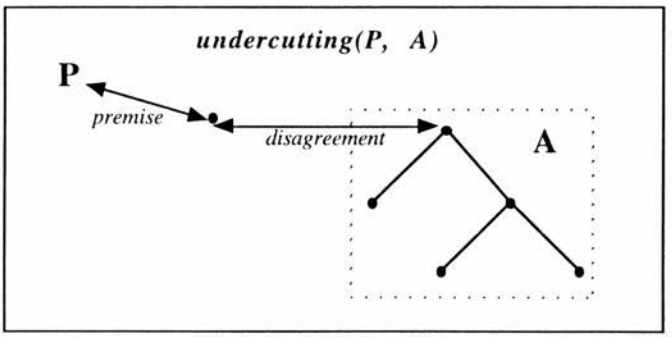


Figure 5.6: Undercutting

G. *Counter-argument.*

This structure is closely related to the definition given by [Sartor 93], in which a *counter-argument* to a proposition P is defined as an argument for the 'complement' (negation) of any proposition used anywhere in an argument for P. Again we can interpret this in two ways, either strictly, if we restrict the arguments to *strong* arguments or more flexibly, using *hybrid* arguments. This is a recursive extension of the rebuttal and undercutting definitions (the recursion happens in the premise_set destructor). Rebuttal involves a disagreement with the conclusion of an argument. Undercutting involves a disagreement with a premise of an argument. Counter-argument involves a disagreement with any proposition appearing in an argument and it thus subsumes rebuttal, undercutting (and in the hybrid case, undermining).

Definition : counter-argument(P, A) iff
P is a proposition name and A is an argument, such that
rebuttal(P, A)
$\vee$ $\exists B$. *premise_set(P, B, Set)* & $Q \in Set$ & *rebuttal(Q, A).*

The Inuit and Green Alliance argument that oil production reduces work in fishing and hunting is a counter-argument to the Petroleum Club's claim that a lease should be granted for oil production. The argument that employment will be increased is a counter-argument to the Green's claim that the lease should not be granted.

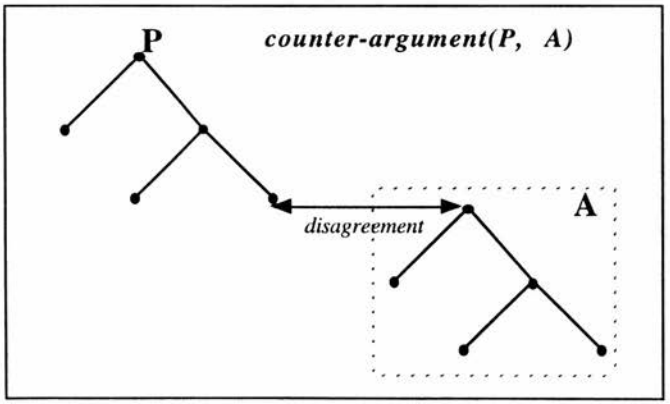


Figure 5.7: Counter-argument

H. Target

The *target* of a proposition in debating terms is an argument challenged by the proposition. This involves an argument which contains a premise with which the proposition disagrees. The target of a proposition is thus the consequence of a disagreement with P. For example, the target of the Petroleum Club's claim that the environmental impact statement is satisfactory, is the Green Alliance's argument against the oil production lease.

Definition : target(P, A) iff
P is a proposition name and A is an argument, such that
∃Q. disagree(P, Q) & consequence(Q, A).

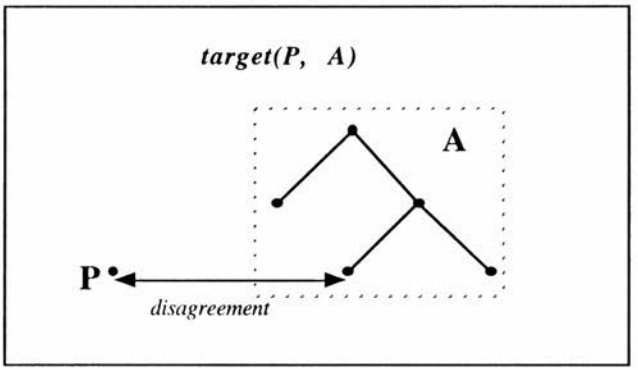


Figure 5.8: Target

I. Counter-consequence

The *counter-consequence* of a proposition P is an argument whose conclusion is counter-argued by the consequence of P. In other words, arguing *from* P produces a counter-argument to another proposition Q. The argument for Q which is challenged by this argument from P is called P's counter-consequence. In FORA this means it is possible to draw a conclusion from P which disagrees with one of Q's supports. Whereas the target is an argument which is directly challenged by P, the counter-consequences are only indirectly challenged by P (in the target, P is the 'head' (conclusion) of the counter-argument and disagrees directly with a premise, while in the counter-consequence, P is in the 'body' of the counter-argument).

Definition : counter-consequence(P, A) iff
P is a proposition name and A is an argument, such that
∃B,P1,P2. *consequence(P, B) & argument(B, P1, _)*
& disagree(P1, P2) & consequence(P2, A).

In the example, the argument in support of an oil lease is a counter-consequence of the Green Alliance's claim that oil production causes pollution and disturbance.

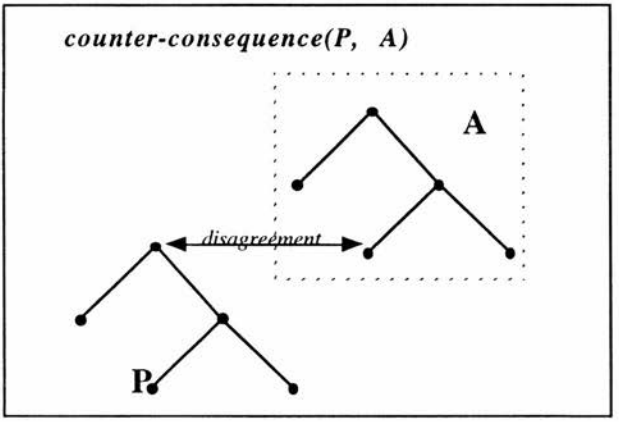


Figure 5.9: Counter-consequence

5.3.3 Summaries of support and opposition

Given a knowledge base containing many arguments and counter-arguments, with potentially very complex structure, it is useful to be able to give a user an unstructured view of the content of a knowledge base which is relevant to a proposition. This can be of two sorts : either the propositions used in arguments for and from the proposition, which is called the *context*, or all the propositions used in opposition to the proposition, which is called the *counter-context*. Both are just sets of propositions with no argumentation structure. Propositions which occur in both sets (*ie:* evidence used by both sides of a debate) are either common knowledge or highly controversial.

J. Context

The *context* of a proposition is the set of propositions in arguments for it and in its consequences. In other words it is the set of propositions used by one side of a debate.

Definition : context(P, S) iff
P is a proposition name and S is the set such that

$$\forall Q.(\ (\exists A,S1.\ premise\_set(A,\ P,\ S1)\ \&\ Q \in S1) \vee$$
$$(\exists A,C,S1.\ consequence(P,\ A)$$
$$\&\ premise\_set(A,\ C,\ S1)\ \&\ (Q = C \vee Q \in S1))$$
$$\rightarrow Q \in S)$$

K. Counter-context

The *counter-context* of a proposition is the set of propositions in counter-arguments to it, in its targets and in its counter-consequences.

Definition : counter_context(P, S) iff
P is a proposition name and S is the set such that

$$\forall Q((\exists A,C,S1.\ counter\text{-}argument(P,\ A)$$
$$\&\ premise\_set(A,\ C,\ S1)\ \&\ (Q = C \vee\ Q \in S1)) \vee$$
$$(\exists A,\ C,S1.\ target(P,\ A)$$
$$\&\ premise\_set(A,\ C,\ S1)\ \&\ (Q = C \vee\ Q \in S1)) \vee$$
$$(\exists A,C,S1.\ counter\text{-}consequence(P,\ A)$$
$$\&\ premise\_set(A,\ C,\ S1)\ \&(Q = C \vee\ Q \in S1))\ )$$
$$\rightarrow Q \in S)$$

5.3.4 Some comments about terminology

Because these argumentation structures are an important part of the thesis, and because they will reappear throughout the rest of this document, it is necessary here to clarify a few points of terminology. In informal language some of these terms have rather looser meanings than the definitions just given allow for. In particular, in normal speech they do not exclusively refer to a relationship between a proposition and a single argument, as sometimes they are used to relate two arguments, or two propositions. In the remainder of this document, there are three ways in which the usage of terms may vary from the strict relationship of a proposition to an argument, so these usages need to be explained.

1. Sometimes the set of all arguments of a particular type is referred to, in which case the plural is used. For example, instead of saying 'the set of arguments each of which is a rebuttal of P', I use the expression 'the rebuttals of P'.

2. Sometimes it is also useful to use the same terminology to describe the relationship between two arguments. So, for example, the expression 'a counter-argument to A' (where A is an argument), should always be interpreted as meaning, strictly, 'a counter-argument to the *conclusion* of A'. Likewise, the expression, 'A rebuts B' (where A and B are both arguments) means 'A is a rebuttal of the conclusion of B'.

3. Sometimes what is intended is a relationship between a proposition name P and the conclusion of a related argumentation structure. For example in the expression 'one of the consequences of P' (where P is a proposition name) this means 'a conclusion of a consequence of P'.

Where the intended interpretation is not clear from the way the expression is used, the verbose version will be given. The simpler expressions are used in an attempt to make the text easier to read, but it should be borne in mind that the formal definitions are always intended when this is the case.

5.3.5 Summary

This section has defined a set of argumentation structures which provide a broad view of the patterns of reasoning possible within a knowledge base, and provide a rich, coherently organised source of information relevant to any particular proposition. The FORA system can construct all of these structures for any proposition in its knowledge base using the tools described in section 5.4.

The argumentation structures provide a user with detailed comparisons between knowledge bases, detailed in that they provide, for each individual proposition, an analysis of how it relates to other propositions in the knowledge bases. The counter-context sets and the sets of rebuttals, undercuttings, underminings, counter-arguments, counter-consequences and targets provide a detailed analysis of *conflicts*, so that those propositions (such as the claim that leases for oil production should be granted) with large sets are highly controversial. Propositions with large corroboration sets are points of *consensus*. Large context sets and many enlargement and consequence sets are indications of highly *coherent* knowledge bases. Likewise small context sets or few enlargements or consequences indicate *fragmentary* knowledge. These structures can provide evaluations of the overall 'shape' of FORA knowledge bases, either during their construction or to guide a user in exploration of completed knowledge bases. The next section describes some tools for this sort of exploration and gives an example of their use in terms of the aflatoxin debate.

5.4 Tools for using argumentation structures

The first part of this section describes how the structures defined in section 5.3 are built automatically in FORA. The second part describes a tool for suggesting FORA knowledge base extensions using the structure definitions.

5.4.1. Synthesis of argumentation structures

In FORA, the structures defined in the previous section are defined declaratively. They can be automatically synthesised, given any FORA knowledge base, providing the user with a lot of useful information about the form of support or opposition for any particular proposition they choose.

For example, suppose while using the aflatoxin knowledge base a user wishes to know what support or opposition there is for the FDA policy. FORA can automatically construct counter-arguments, rebuttals, corroborations and so on and allow the user to browse them. A pull-down menu allows a user to select a particular type of argumentation structure - or all possible structures - for the proposition they are interested in. Here is the result of the user selecting the statement of the FDA policy and requesting all argumentation structures relating to it within the knowledge base as shown in chapter 4.

Argumentation structures relating to :
The FDA policy level for aflatoxins should be 20ppb

Arguments :
The FDA policy level for aflatoxins should be 20ppb<=
The maximum acceptable level of aflatoxins is 20ppb<=

There is no safe level of aflatoxins<=
Aflatoxins cause cancer in humans<=
Aflatoxins cause cancer in non-human animals<=
Aflatoxins cause cancer in animals<=
Aflatoxins cause cancer in non-human animals<= Loop
Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable<=
Aflatoxins are a kind of chemical<= Assumption
Animals are good indicators of the cancer risk of chemicals to humans<= Assumption
The minimum detectable level of aflatoxins is 20ppb<= Assumption
Aflatoxins cause cancer in humans<=
Aflatoxins cause cancer in non-human animals<=
Aflatoxins cause cancer in animals<=
Aflatoxins cause cancer in non-human animals<= Loop
Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable<=
Aflatoxins are a kind of chemical<= Assumption
Animals are good indicators of the cancer risk of chemicals to humans<= Assumption

There is a direct rebuttal of the FDA policy.

REBUTTAL of
The FDA policy level for aflatoxins should be 20ppb

Disagrees with
The FDA policy level for aflatoxins should not be 20ppb

Argument :
The FDA policy level for aflatoxins should not be 20ppb<=
20ppb is not the maximum acceptable level of aflatoxins<=
The minimum detectable level of aflatoxins is 20ppb<= Assumption
The level of adverse effects of aflatoxins is 200ppb<=
Aflatoxins cause cancer in animals<=
Aflatoxins cause cancer in non-human animals<=
Aflatoxins cause cancer in animals<= Loop
Aflatoxins cause liver toxicity in animals<= Assumption
200ppb is much greater than 20ppb<= Assumption

There are two undercuttings of the FDA policy, as both immediate premises (the maximum acceptable level claim, and the claim that aflatoxins cause cancer in humans) are disagreed with.

UNDERCUTTING of
The FDA policy level for aflatoxins should be 20ppb

Premise :
The maximum acceptable level of aflatoxins is 20ppb

which disagrees with
20ppb is not the maximum acceptable level of aflatoxins

Argument :
20ppb is not the maximum acceptable level of aflatoxins<=
The minimum detectable level of aflatoxins is 20ppb<= Assumption
The level of adverse effects of aflatoxins is 200ppb<=

Aflatoxins cause cancer in animals<=
Aflatoxins cause cancer in non-human animals<=
Aflatoxins cause cancer in animals<= Loop
Aflatoxins cause liver toxicity in animals<= Assumption
200ppb is much greater than 20ppb<= Assumption

UNDERCUTTING of
The FDA policy level for aflatoxins should be 20ppb

Premise :
Aflatoxins cause cancer in humans

which disagrees with
There is a lack of scientific evidence showing aflatoxins cause cancer in humans

Argument :
There is a lack of scientific evidence showing aflatoxins cause cancer in humans<= Assumption

There are 4 counter-arguments to the FDA policy. The first two are the two undercuttings, as just shown, so these are omitted here (though FORA generates them). There are three further counter-arguments attacking two other premisses of the argument for the FDA policy (there are two rebuttals of the extrapolation claim).

COUNTER-ARGUMENT for
The FDA policy level for aflatoxins should be 20ppb

Support :
There is no safe level of aflatoxins

Disagrees with :
There is a lack of scientific evidence showing no safe level of aflatoxins

Argument :
There is a lack of scientific evidence showing no safe level of aflatoxins<= Assumption

COUNTER-ARGUMENT for
The FDA policy level for aflatoxins should be 20ppb

Support :
Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable

Disagrees with :
Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable

Argument :
Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable<=
Extrapolation of cancer risk from one species to another is unreliable<= Assumption

COUNTER-ARGUMENT for
The FDA policy level for aflatoxins should be 20ppb

Support :
Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable

Disagrees with :
Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable

Argument :
Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable<=
The effect of aflatoxins varies greatly between species<= Assumption

The context of the FDA policy claim is the set of statements used in the argument for it.

CONTEXT of
The FDA policy level for aflatoxins should be 20ppb

Set of propositions :

The maximum acceptable level of aflatoxins is 20ppb
There is no safe level of aflatoxins
The minimum detectable level of aflatoxins is 20ppb
Aflatoxins cause cancer in humans
Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable
Aflatoxins cause cancer in animals
Aflatoxins cause cancer in non-human animals
Aflatoxins are a kind of chemical
Animals are good indicators of the cancer risk of chemicals to humans

The counter-context of the FDA policy claim is the set of statements used in counter-arguments, rebuttals etc.

COUNTER-CONTEXT of
The FDA policy level for aflatoxins should be 20ppb

Set of propositions :

There is a lack of scientific evidence showing no safe level of aflatoxins
There is a lack of scientific evidence showing aflatoxins cause cancer in humans
Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable
The FDA policy level for aflatoxins should not be 20ppb
20ppb is not the maximum acceptable level of aflatoxins
The minimum detectable level of aflatoxins is 20ppb
The level of adverse effects of aflatoxins is 200ppb
200ppb is much greater than 20ppb

5.4.2 Extending the knowledge base by arguing

Once arguments exist, it is useful to be able to augment the knowledge base to support or attack the arguments within it. A useful application of the formal argumentation structures is to use them to *suggest* additions to the knowledge base. The *arguer* program is an implementation of this facility. If the user wishes to attack a proposition, they can be told of all the ways currently available to weaken the argument for that

proposition, by producing counter-arguments or rebuttals, *etc.* Alternatively, they may wish to support existing arguments by providing additional corroborations and so appropriate information to enable this is also generated. Here is an example of the output of the arguer program, when asked to suggest ways of arguing about the FDA policy statement.

Some suggested ways of arguing about :
The FDA policy level for aflatoxins should be 20ppb

You could produce a corroboration for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
equivalent(The FDA policy level for aflatoxins should be 20ppb, Equivalence)

You could produce a enlargement for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
elaboration(The FDA policy level for aflatoxins should be 20ppb, Elaboration_Set)

You could produce a consequence for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
justification(Conclusion, [The FDA policy level for aflatoxins should be 20ppb, Justification_Set])

You could produce a rebuttal for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(The FDA policy level for aflatoxins should be 20ppb, Disagreement)

You could produce a undercutting for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(The maximum acceptable level of aflatoxins is 20ppb,Disagreement)

You could produce a counterargument for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(The maximum acceptable level of aflatoxins is 20ppb,Disagreement)

You could produce a counterargument for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(Disagreement, There is no safe level of aflatoxins)

You could produce a counterargument for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(Disagreement, The minimum detectable level of aflatoxins is 20ppb)

You could produce a counterargument for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(Disagreement, Aflatoxins cause cancer in humans)

You could produce a counterargument for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(Disagreement, Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable)

You could produce a counterargument for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(Disagreement, Aflatoxins cause cancer in animals)

You could produce a counterargument for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(Disagreement, Aflatoxins cause cancer in non-human animals)

You could produce a counterargument for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(Disagreement, Aflatoxins are a kind of chemical)

You could produce a counterargument for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
disagree(Disagreement, Animals are good indicators of the cancer risk of chemicals to humans)

You could produce a target for
The FDA policy level for aflatoxins should be 20ppb
by adding a relation of the form :
justification(Conclusion, {The FDA policy level for aflatoxins should not be 20ppb\Justification_Set})

5.4.3 The Devil's Advocate

This section sketches an tutor helping a student explore the various points of view represented in a set of FORA knowledge bases.

OLIA is an argumentation tutoring system, specifically for use with opposing points of view represented in FORA. One part of this system plays 'Devil's Advocate' to a user (student) who is encouraged to follow a line of argument for one position. As the user selects propositions which lend support to their view, the Devil's Advocate argues against them by attacking their position, or arguing in favour of an opposing viewpoint. It does this using argumentation structure definitions as given in section 5.3. A FORA knowledge base is used as the expert model in the tutoring system, and it includes various tutoring strategies which are all defined in terms of FORA argumentation structures. The most interesting of these is the Devil's Advocate strategy which attempts to produce a rebuttal of every proposition the user selects, and then challenges the student to further justify their position. The student does this by selection of other propositions from the knowledge base. If they are indeed

justifications for the student's current position, then the 'Devil' continues its process of rebuttal, but if the student fails to justify the position the 'Devil' informs them of this, and offers tutoring support using a different strategy, such as more straightforward coaching in the use of the argument structures. This program was implemented as part of an MSc project and full details can be found in [Retalis 95].

The Devil's Advocate was evaluated using two cohorts of students. A preliminary (formative) evaluation involved MSc students who were tutored using a simple knowledge base about the greenhouse effect, which I constructed from the results of the knowledge acquisition experiments described in Chapter 4. This provided useful feedback about the implementation of the tutor. A further evaluation was later carried out by building another FORA knowledge base containing arguments for the idealist and materialist positions on the 'mind body problem' which forms part of the first year curriculum in Philosophical Issues in AI at the University of Edinburgh. This example was used by first year students in a practical session which complemented a tutorial and allowed us to conclude that students both enjoy and find useful the process of arguing with a computer as part of their studies. About half of the students found that using the tutor helped to clarify the structure of the arguments more effectively than their tutorial. Further details can be found in [Retalis *et al* 96] .

This student project provides evidence that there are useful educational applications of FORA and these are currently being pursued further. It also provides evidence that FORA can act as a basis for definition not only of debating *structures* as shown in section 5.3, but also of debating *strategies* which characterise how patterns of arguments can be combined in a meaningful interaction with a user.

5.5 Summary

In this chapter I have identified the intended users of FORA (primarily knowledge engineers in difficult domains) and described some tools implemented in FORA for exploring and modifying the arguments in FORA knowledge bases. I have also shown how the basic FORA language enables declarative definitions of more complex argumentation structures to be written which give more insight into the structure of a knowledge base.

The first tool, *explorer,* is a basic mechanism for exploring conflicts in a knowledge base and discovering relationships between propositions.

The second tool is a program called *arguer* which shows the user ways in which the arguments in the knowledge base can be challenged or expanded. It does this using the library of argumentation structures.

The third tool described briefly is a tutoring tool which adopts the position of Devil's Advocate to encourage a student to construct stronger arguments and meet the challenge of counter-arguments. Experience with the Devil's Advocate suggests useful future applications of FORA in an educational role.

In the first section, I made some assumptions about functionality which would be useful to a knowledge engineer using FORA to assist in the construction of knowledge bases in difficult domains. The first four functionality requirements have now been met. They were :

1. To express arguments - this is possible using the hypertext mark-up tool described in the previous chapter, or directly within FORA.

2. To explore existing arguments - the *explorer* tool lets a user explore FORA knowledge bases containing arguments.

3. To relate the arguments from one source to those of another - the argumentation structures provide overviews of the way arguments relate to each other, and FORA generates these automatically.

4. To add new arguments and to challenge existing arguments - FORA allows new arguments to be added, and the *arguer* tool suggests ways in which existing arguments could be either augmented or challenged.

The argumentation structures defined in this chapter play an important role in this thesis because they demonstrate the benefit of taking both a formal and a meta level approach to argumentation.

1. In FORA, concepts from the literature on argumentation, like counter-argument and corroboration, can be formally defined from the four basic relations and arguments, and as a result they can be constructed and reasoned with automatically. This is a new contribution to existing, more informal work on argumentation, in which there is little attempt to define existing concepts in terms of other, more simple, concepts.

2. Having taken an abstract, meta-level view of arguments, we have been able to go a long way towards providing useful support for argumentation without having to consider at all what kind of object level logic is appropriate for representing the content of propositions, nor what sort of logical inference will provide the reasoning steps in an argument. We can decide the structure of these steps without having to decide how to give them logical 'force'. FORA is therefore a contribution towards existing research on formal argumentation in which these object-level decisions have to be

made in order to represent the structure of arguments.

The next two chapters examine how FORA relates to knowledge bases represented using particular object-level logics.

Chapter 6

Automating Mark-up to FORA

6.1 Introduction

This chapter discusses how FORA knowledge bases can be generated automatically by abstraction from existing knowledge bases represented in a formal language.

It is useful to integrate existing knowledge bases into FORA because the arguments they express can then be combined with arguments from other knowledge bases, or text, or the opinions of a user. The FORA framework enables a user (as we saw in the previous chapter) to challenge an argument and provide a counter-argument, or to shore up a claim by providing corroborating evidence, and to weigh up the overall weight and coherence of multiple arguments for controversial points of view. Enabling existing representations of knowledge to be included in this process is useful, particularly where the knowledge bases express expertise in rapidly expanding or changing fields of knowledge (and which may therefore need revision as they become out of date, or are challenged by new evidence), or where the knowledge base expresses the views of only one of many important stakeholders in a decision-making situation and there is a desire to broaden the range of views taken into account.

Knowledge bases built in the past, which are no longer adequate to meet the needs of users and require updating, are known as 'legacy' knowledge bases. This chapter contributes to our understanding of how to reuse such knowledge bases. Some of the problems of using legacy knowledge bases are that they need to be integrated with other information, they need to be updated and refined, and they sometimes need to be re-represented in order to be used with new reasoning techniques. The re-engineering of such 'legacy' knowledge bases is an important current issue.

In chapter 4 the process of building knowledge bases using hypertext techniques was called 'marking-up' text, and I use the same word here to describe computational techniques for discerning the structure of arguments expressed in knowledge bases. It is clearly not yet feasible to automate mark-up of texts, as this would require both understanding of natural language and interpretation of the intended line of reasoning. However, it *is* feasible to automate mark-up where knowledge has already been formally expressed and where chains of reasoning can be carried out by an inference engine. This chapter shows that it is feasible, by explaining how it can be done.

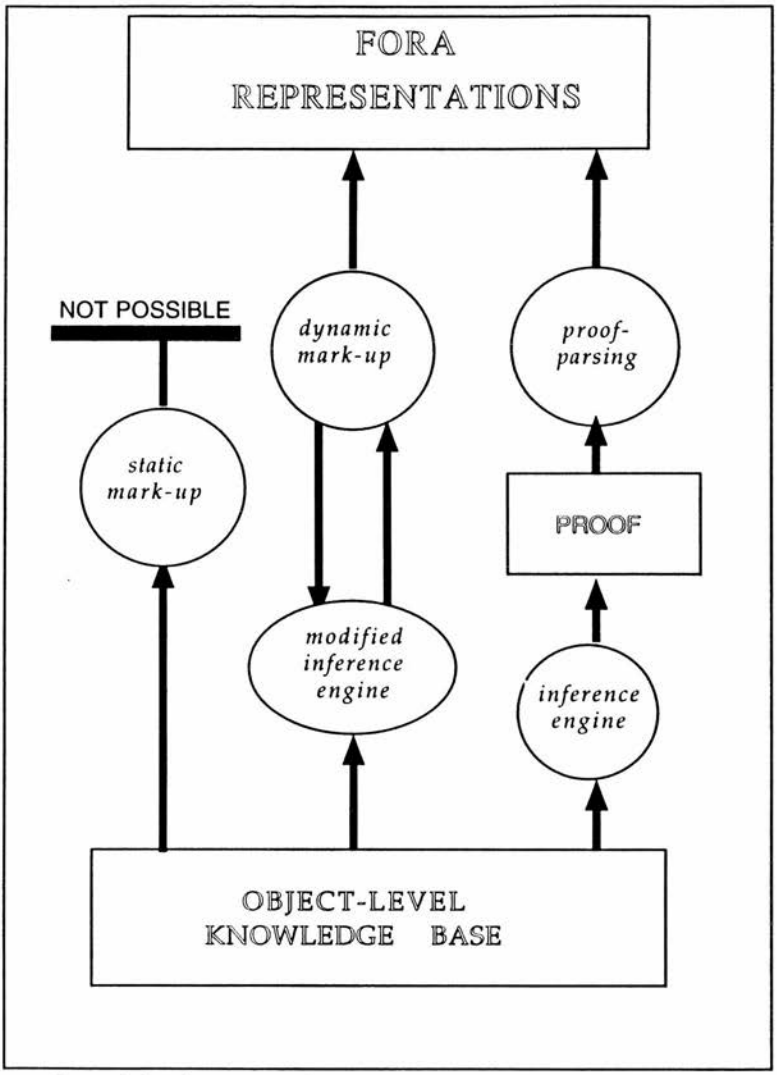


Figure 6.1 : Three approaches to automating mark-up. The left-hand path from the knowledge base shows static mark-up, which fails to create a full FORA representation, due to lack of reasoning. The middle path shows dynamic mark-up, using a modified inference engine which can record meta-level representations of its reasoning steps. Proof parsing is the right-hand path, which more elegantly abstracts a FORA representation from completed proofs generated by an unaltered inference engine.

This chapter describes three approaches to the mark-up task. They are illustrated in Figure 6.1, as the three attempted paths from the knowledge base at the bottom of the diagram, to its representation in FORA at the top of the diagram.

The left-most path of Figure 6.1 shows what happens when mark-up is attempted by simply looking at the syntax of the knowledge base. This is called 'static mark-up' and it *fails* as a method of extracting arguments from a knowledge base. Section 6.2.1 explains why it fails by showing that mark up *must* involve inference at the object-level.

The remainder of section 6.2 describes a successful method for automating the mark-up process. This is the central path from the knowledge base to FORA, shown in figure 6.1. It is called 'dynamic mark-up'. Dynamic mark-up is achieved by using and adapting an object-level inference system so that it records meta-level representations of the reasoning steps it uses when drawing a conclusion from the knowledge base. The object-level system used to illustrate this chapter is a sequent calculus theorem prover. It is described in section 6.2.3. The details of the dynamic mark-up technique are given in section 6.2.4. Some problems with dynamic mark-up are spelled out in section 6.2.5.

In Figure 6.1, the right-hand path from the knowledge base to FORA uses an unaltered object-level inference mechanism to generate a proof of some conjecture. This proof is then parsed to generate a FORA representation of the reasoning it involves. This is a more elegant approach to mark-up because it does not require rewriting of the object-level inference engine. Once parsed, proofs can be described at various levels of abstraction : from a step by step description to a more abstract *proof outline*. The proof-parsing and proof-outlining methods are described in section 6.3.

Both successful approaches have been implemented to mark-up sequent calculus proofs for the restricted group of knowledge bases using only classical propositional calculus. This provides a demonstration of the plausibility of automating mark-up to FORA, and is sufficient to suggest that a useful line of future research will be to automate mark-up from an extended range of legacy knowledge bases represented in more complex object-level languages. Section 6.4 describes one such extension by returning to the running example, and describing automated mark-up of a full first order predicate logic representation of the aflatoxin debate, using the proof parsing method of section 6.3. Section 6.5 is a summary.

6.2 Abstracting arguments from knowledge bases

This section explores how to extract arguments from object-level knowledge bases, so

that they can be represented in FORA. First, in section 6.2.1, I argue that mark-up is more than a syntactic process and *must* involve an object-level reasoning mechanism. I do this by presenting a *static mark-up* process which does not involve inference, and show why it must fail to produce a complete representation of the arguments implicit in the knowledge base. Having established that inference is essential to mark-up, in section 6.2.2, I discuss at what stage in the mark-up process inference should happen. One option is to embed the inference mechanism inside the mark-up process. In section 6.2.3, I describe the inference mechanism, a sequent calculus theorem prover, used for these experiments in automating mark-up to FORA. Finally, in section 6.2.4, I describe how mark-up can be achieved by modifying this inference mechanism, and embedding it inside the mark-up process. I conclude, in section 6.2.5, with some problems with this approach. This section leads into section 6.3, in which a more elegant and satisfactory approach to automated mark-up is described.

6.2.1 Why mark-up must involve inference

The simplest possible way of marking-up a knowledge base would seem to be to look at it and notice where relationships are indicated between statements which could be described at the meta-level. In chapter 4, the FORA relations (such as disagreement), were described as representations of object-level notions (such as negation). So, if the knowledge base is represented as a theory of propositional logic, it would seem reasonable that implications could be described as justifications, negations could indicate disagreements and so forth.

The purpose of this section is to show that mark-up is not as simple as this, and that just describing a static knowledge base will not reveal all of the interesting argument structure which it implicitly contains. To reveal the arguments which can be produced from a knowledge base it is necessary to carry out inference and attempt to draw conclusions. To demonstrate this, I will explain what happens when attempting to mark-up a static knowledge base, without invoking an inference mechanism.

Assume that it is possible to look at the structure of the object-level theory (or knowledge base) and match these structures with FORA representations. If so, it must be possible to state syntactic patterns at the object level which correspond to FORA relations. The following very simple algorithm carries out matching on a set of such patterns, or templates.

The algorithm is intended to construct a FORA meta-theory consisting of the set of all meta-level statements generated from the propositional theory. Each meta-level statement is generated from a set of syntactic templates (the markup_links), which relate syntactic structures in the object-level propositional logic to FORA relations.

The syntactic structures are then instantiated with respect to the propositional theory by matching the structures against the theory (by searching for appropriate sub-theories).

Algorithm for static mark-up

Given an object-level theory, T, static mark-up generates the FORA meta-theory, M, which is the set of all relations, R, which are meta-statements describing T.

$$\forall T, \forall M.\ \textit{static-mark-up}(T, M) \quad \leftrightarrow$$
$$(\forall R.\ \textit{meta-statement}(T,R) \rightarrow R \in M.)$$

Given a theory, T, the meta-statement, R describes a part of it, if and only if there is a mark-up template for a set of object-level propositions, O, described at the meta-level as R, such that O matches with a subset of T.

$$\forall T, \forall R.\ \textit{meta-statement}(T,R) \leftrightarrow$$
$$(\exists O.\ \textit{markup\_link}(O, R)\ \&$$
$$\textit{subset\_match}(O, T))$$

A possible set of templates (*markup_links*) for classical first order propositional calculus is as follows. Each link is a predicate taking two arguments - the first being the *set of propositions* which will match (unify) with a subset of the object-level theory, the second being the *FORA relation* which describes the set.

1. There is a disagreement between a proposition and its negation

$$\textit{markup\_link}([P, \neg P], \quad \textit{disagreement}(P, \neg P)).$$

2. Conjunction and disjunction are symmetric

$$\textit{markup\_link}([P\&Q, Q\&P], \quad \textit{equivalent}(P \,\&\, Q, Q \,\&\, P)).$$
$$\textit{markup\_link}([P \vee Q, Q \vee P], \quad \textit{equivalent}(P \vee Q, Q \vee P)).$$

3. De Morgan's Laws

$$\textit{markup\_link}([\neg(P \,\&\, Q), \neg P \vee \neg Q], \quad \textit{equivalent}(\neg(P \,\&\, Q), \neg P \vee \neg Q)).$$
$$\textit{markup\_link}([\neg(P \vee Q), \neg P \,\&\, \neg Q], \quad \textit{equivalent}(\neg(P \vee Q), \neg P \,\&\, \neg Q)).$$

4. There are equivalences between conditionals

$$\textit{markup\_link}([\neg P \vee Q, P \rightarrow Q], \quad \textit{equivalent}(\neg P \vee Q, P \rightarrow Q)).$$
$$\textit{markup\_link}([P \leftrightarrow Q, (P \rightarrow Q), (Q \rightarrow P)],$$
$$\textit{equivalent}(P \leftrightarrow Q, (P \rightarrow Q) \,\&\, (Q \rightarrow P))).$$

5. Proposition sets matching to inference rules, such as *modus ponens*, *modus tollens* and *abduction*, can generate justifications

markup_link([P, P → Q], *justification(Q, [P → Q, P])).*
markup_link([P → Q, ¬Q], *justification(¬P, [P → Q, ¬Q])).*
markup_link([Q, P → Q], *justification(P, [P → Q, Q])).*

6. In order to match to an inference rule such as *modus ponens* being applied with with more complex premises, such as conjunctions or disjunctions, the syntactic matches need to be spelled out.

markup_link([A & B → P, A, B], *justification(P, [A & B → P, A, B])).*
markup_link([A ∨ B → P, A], *justification(P, [A ∨ B → P, A])).*
markup_link([A ∨ B → P, B], *justification(P, [A ∨ B → P, B])).*

7. Finally, here is a syntactic match indicating an application of the *and-elimination* inference rule.

markup_link([P & Q], *justification(P, [P & Q])).*
markup_link([P & Q], *justification(Q, [P & Q])).*

Just by looking at what is needed to write syntactic mark-up links we can start to see the problem. Many of these links are attempts to state the syntactic conditions for the application of inference rules. The root of the problem is the use of *subset_match/2* which carries out first-order term *unification* to match subsets of the object-level theory, whereas what is really required is full logical *equivalence*. However, to establish logical equivalence is in general undecidable, and therefore requires a theorem prover to be invoked as part of the unification process.

An example will illustrate the difficulty. The following propositional theory :

theory([p, q, ¬p, ¬q, p&q, q&p, q∨p, p∨q, p→q, q→r, ¬q∨ r, q→p,
p↔q, p&a&b, p&(a∨b), p&q→w, p∨q→b, (p→ q) →z, ¬(p&q),
a&p&b, ¬p∨¬q]).

generates a set of 33 meta-level statements (3 disagreements, 7 equivalences and 21 justifications). Examination of these shows up a number of problems.

Problems

1. Disagreements should show up between the negation of a proposition P and a statement Q which is equivalent to it, *eg:* we get

disagreement(p&q, ¬(p&q))

but we should also get

disagreement(q&p, ¬(p&q))
and *disagreement(¬p ∨ ¬q, p & q,)*

This can only be achieved by the addition of a template which has a condition attached to it, as follows :

markup_link([P, ¬Q], disagreement(P, ¬Q)) ←
markup_link([P, Q], equivalent(P, Q)).

However, even this conditional template only solves the first of the problems, but not the second. Part of the problem is the lack of symmetry of the disagreement template (ie: the first or second of the arguments may be negated, not just the second), and of the equivalence template. In addition, this added template is problematic as it is really expressing a meta-level relation between disagreements and equivalences so it should be left to the meta-level. The question of symmetry is also handled already at the meta-level.

2. Equivalence checks should be more sophisticated and applied recursively, in order to pick up the equivalence between *p&a&b* and *a&p&b*, etc.

(p& (a&b)) ↔ ((a&b) &p) ↔ (a & (b&p)) ↔ (a& (p&b))

3. Justifications are only partial. For example, only some of the justifications by and-elimination are produced :

justification(p, [p&a&b]) and
justification(a&b, [p&a&b])

are generated, but we should also get

justification(a, [p&a&b])
justification(b, [p&a&b])
justification(p&b, [p&a&b])
justification(p&a, [p&a&b])

There is thus a need for the justification rules to be applied repeatedly. In general, given a set of propositions such as *[r&s, r→t]* it should be possible to generate an argument for *'t'* in FORA, by recognising that through a sequence of inference steps, *'t'* can be deduced. This can only be done by invoking the object-level inference engine or theorem prover, because in FORA, the propositions are all atomic.

These problems lead to the conclusion that it is not enough to have mark-up templates which recognise that a set of propositions in a knowledge base 'look like' they can be used to make a deduction. Inference actually needs to be carried out in order to pick up derivable conclusions and logical equivalences which are not syntactically obvious. Therefore mark-up must involve the object-level theorem prover or inference mechanism. In general the inference needed will be arbitrarily complex. For example, to establish all possible disagreements involves finding all pairs of contradictory formulae derivable from the theory, which is not a decidable problem for the first order case in general, so limits to the inference search space will need to be put in place. The general point here is that completeness of the set of FORA relations generated is limited by the inferential mechanism used at the object-level.

6.2.2 When should the inference happen?

Having established that inference is necessary for mark-up, the next question is whether inference should be applied at the object level before mark-up, during mark-up, or after mark-up (at the meta-level). The latter is not appropriate, because at the meta-level the proposition *'a&b'* is simply a name or atom, and thus the application of and-elimination to it is impossible at the meta-level and would require object-level syntactical knowledge.

The two options left are :
(1) to use a mark-up process which performs inference at the object-level in order to generate, incrementally, all possible candidates for mark-up; or
(2) to generate the closure of the theory with respect to the inference mechanism, or a complete proof of a target proposition, then mark that up after the inference is complete.

Note that both possibilities potentially do not terminate (repeated application of or-introduction, for example, will not terminate in general).

Another example will make clear what the options are. Suppose we have a theory as follows :

$$[a\&b,\ a \rightarrow c].$$

Option 1 involves generating

$$justification(c,\ [a \rightarrow c,\ a\&b])$$

by a mark-up process which itself carries out the necessary and-elimination step before applying the modus-ponens template, and representing it as a justification in FORA. With this option, inference is embedded in the mark-up process.

Alternatively, option 2 would first generate the closure of the theory with respect to the inference mechanism...

$[a\&b,\ a\rightarrow c,\ a,\ b,\ c,\ a\vee b,\ a\vee c,.....]$ [1]

...and then mark-up *afterwards*. A more conservative reading of the second option (see section 6.3) is to target a particular conjecture and use the inference mechanism to generate a proof of that conjecture, and then mark-up that proof.

The second option is discussed in detail in section 6.3. First we will look more closely at what the first option involves. In either case the mark-up process requires an inference engine or theorem prover, so I begin by describing the one I used.

6.2.3 The theorem prover

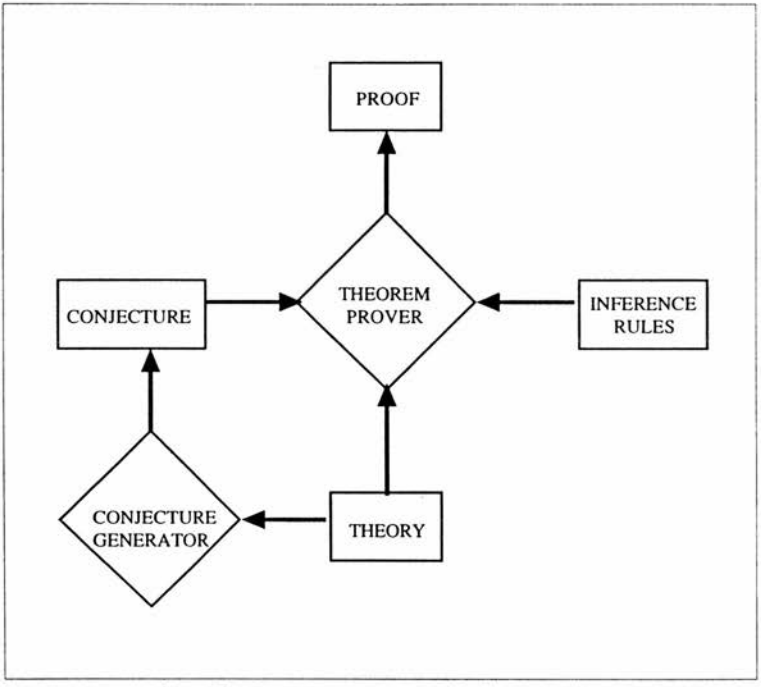


Figure 6.2 : The theorem prover, which applies inference rules to a theory in order to try to generate a proof of a conjecture. The conjecture generator selects propositions named in the theory, which the theorem prover will attempt to prove.

The inference mechanism which I implemented is a propositional logic theorem prover (See figure 6.2). This inference mechanism takes a theory, generates a set of hypotheses (conjectures) to attempt to prove from that theory, and then attempts to

[1] Note that in general this is non-terminating due to the use of or-introduction, so some limit would need to be put on the inference.

prove each of them with the theorem prover, recording the proof if there is one. The set of conjectures which are generated from the theory is the set of literals (atomic propositions) occurring in the theory, plus the set of their negations. The theorem prover can also be invoked to attempt a proof of a conjecture (which can be arbitrarily complex) given by the user, in other words it is not *restricted* only to proving conjectures generated automatically from the object-level theory.

The proofs are generated using sequent calculus inference rules, backward chaining (depth-first, left-to-right) from the goal sequent :

Theory $\vdash$ *Conjecture*

where $\vdash$ means 'entails'. The proof terminates when there are no remaining conditions of inference rules remaining to be satisfied.

Sequent calculus was chosen as the inference technique because it uses a large set of inference rules from which to abstract when marking-up. By contrast, an inference technique such as resolution, which uses only one inference rule, would be expected to pose much less of a challenge when marking-up proofs, because each step in the proof would be handled in the same way.

A second benefit of sequent calculus is that it is straightforward to add new inference rules to the rule-base, including rules which are logically invalid, such as abduction. These can be used to generate plausible (though logically invalid) arguments in FORA.

The third, and most important, benefit of sequent calculus is that its proofs contain a complete record of the proof situation (the state of the theory and conjecture) at each inference rule application. No additional context, history or working memory needs to be recorded, in order to generate a full proof from the theorem prover. Unlike a resolution theorem prover, or a natural deduction theorem prover, there is no need to generate a 'trace' of the proof procedure in order to generate a full record of the proof (including vital information such as the discharge of assumptions, for example). This is explained by Dana Scott in [Scott 81] as follows :

> *"The idea of using sequents is to display on the left of the* $\vdash$ *symbol all the required assumptions. In following the flow of argument, the assumptions will be moved from left to right (or will simply disappear). At every step everything is explicit : the books are always open to public inspection. The cost of this procedure is that your arms get very tired carrying large books around .... The advantage of the sequent scheme is that general arguments about provability are easier to give." (p130)*

An example of an inference rule is:

$$\textit{and-introduction}: \quad \frac{\Gamma \vdash \phi, \Delta \qquad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \phi \,\&\, \psi, \Delta}$$

This rule says that if the theory Γ entails the proposition ϕ (plus the set of propositions Δ), and it also entails the proposition ψ (plus Δ), then Γ must entail the proposition $\phi \,\&\, \psi$ (plus Δ).

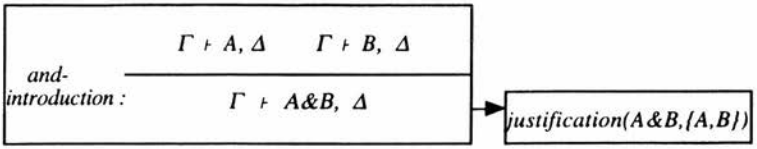
The proofs which are recorded are simply lists of the inference rules applied during the proof. A 'residue' inference rule allows the capture of partial proofs, and thus the ability to record at the meta-level which propositions are not justified and thus are a dead end in a proof tree.

The theorem prover includes a depth-check to prevent proofs of a depth of greater than 25 steps down any branch, and a loop-checker to prevent infinite loops. See section 6.2.1 for comments explaining why a limit to the size of the inference search tree is needed.

6.2.4 Dynamic mark-up

The first of the two mark-up methods using the theorem prover is called 'dynamic mark-up'. The proof procedure is embedded within the mark-up process and drives the generation of FORA relations. The dynamic mark-up module monitors the application of each inference rule by the theorem prover and records 0,1 or 2 meta-level relations for each proof step, as it is taken. Most of these relations are justifications, though negation-introduction and negation-elimination also allow disagreements (between negations) to be recorded.

The FORA relations are added as an additional component in the rules, which ensures instantiation of the proposition names as the propositions are used in the proof. For example, the and-introduction rule shown above would record at the meta-level the fact that $A\&B$ is justified by the set $\{A,B\}$. The and-introduction rule thus becomes as follows, in which an additional component is added by the FORA relation which can be derived from it :



As a proof proceeds all the FORA relations generated by it are recorded. All the propositions mentioned in the relations are altered to make clear that the FORA relation at the meta-level is between the *proposition names*. This is done by surrounding each proposition by triangular brackets, so $A\&B$ becomes $<A\&B>$. This is an important syntactical detail, as it indicates that these are FORA objects which cannot be decomposed, and that they are the basis building blocks for constructing arguments. The point is that FORA does not make any logical distinction between $A \rightarrow B$ and *'Clouds lead to rain'* . In FORA they are both atomic. The final FORA relation recorded by the dynamic mark-up would look like :

justification(<A&B>, [<A>,<B>]).

6.2.5 Problems with dynamic mark-up

The primary problem with this method of mark-up is that it requires alteration to the inference mechanism. Every inference rule is altered by the addition of an extra component representing the FORA relation or relations which relate to the rule application. This therefore requires that the theorem prover is re-implemented in order to reason with these different rules.

Secondly, in dynamic mark-up, knowledge about how rules can be interpreted as argumentation steps is attached to individual inference rules. This is limiting. It would be preferable to represent this knowledge in a more general manner, so that argumentation steps could be deduced from chains of reasoning 'in general' rather than being tied to particular inference rules.

Thirdly, when we produce real arguments, we regularly gloss over several reasoning steps which we could give in more detail if necessary. It would be good if the mark-up process had this sort of flexibility, rather than being merely an echo of each inference step made by the object-level theorem prover.

A fourth problem is that this method of mark-up is entirely dependent on the ability of the modified inference engine to achieve proofs automatically, which is not generally possible, and this limits the potential of mark-up to those cases where a proof is achievable automatically.

Finally, the most problematic thing about dynamic mark up is that the alterations to the inference mechanism would have to be repeated for every different knowledge representation language and new inference engine which needed to be used to produce a FORA knowledge base. Each new legacy knowledge base thus requires, in principal, that its inference engine is rewritten.

In the next section, the other alternative approach to mark-up is examined to show that it is possible to overcome these problems.

6.3 Mark-up by Parsing Proofs

This section describes a second technique which automatically marks-up a proof of a conjecture, constructed by an object-level theorem prover from a theory (or in KBS terminology, an inference chain produced from an inference engine reasoning over a knowledge base). This technique is *proof parsing*. It takes a proof as input and outputs a set of FORA statements which describe the justifications and other argumentation relations used in the proof. These relations are abstractions of the reasoning steps used in the proof.

This technique is an improvement on the earlier attempt to automate mark-up because it does not have the theorem prover embedded inside it. Instead it only uses the output of a theorem prover - the proof - which it parses to deduce the reasoning steps involved in it, resulting in a FORA knowledge base. The proof parser is implemented to parse proofs from the sequent calculus theorem prover as described in section 6.2.3. Proof-parsing is illustrated in figure 6.3.

6.3.1. Parsing Proofs

The individual proof steps are analysed and a record made of the justifications involved in each one. This is done by parsing each of the proof rule applications given in the proof, in turn. A small set of parsing rules analyse the differences between the succedents and the antecedents in the sequents at each stage in the proof in order to extract the nature of the deduction which must have occurred at that point.

The proof-parsing rules are general in form and work at a more abstract level than the syntactical details of the inference rules. This abstraction is achieved by using a separate module of inference rule destructors to determine if the conditions of each parsing rule apply. These destructors are themselves more general-purpose than the dynamic mark-up technique, as they refer only to the relative forms of the sequents used in the inference rules, and do not depend at all on the syntax used to represent conjunction, negation etc. For example, the destructor for checking if the consequent has changed in a rule application applies to all possible sequent calculus

inference rules. Hence if new inference rules are added to the rule base, the destructors can apply equally to them. The rule destructors are the only aspect of this technique which should need to be adapted for a new inference mechanism (such as resolution).

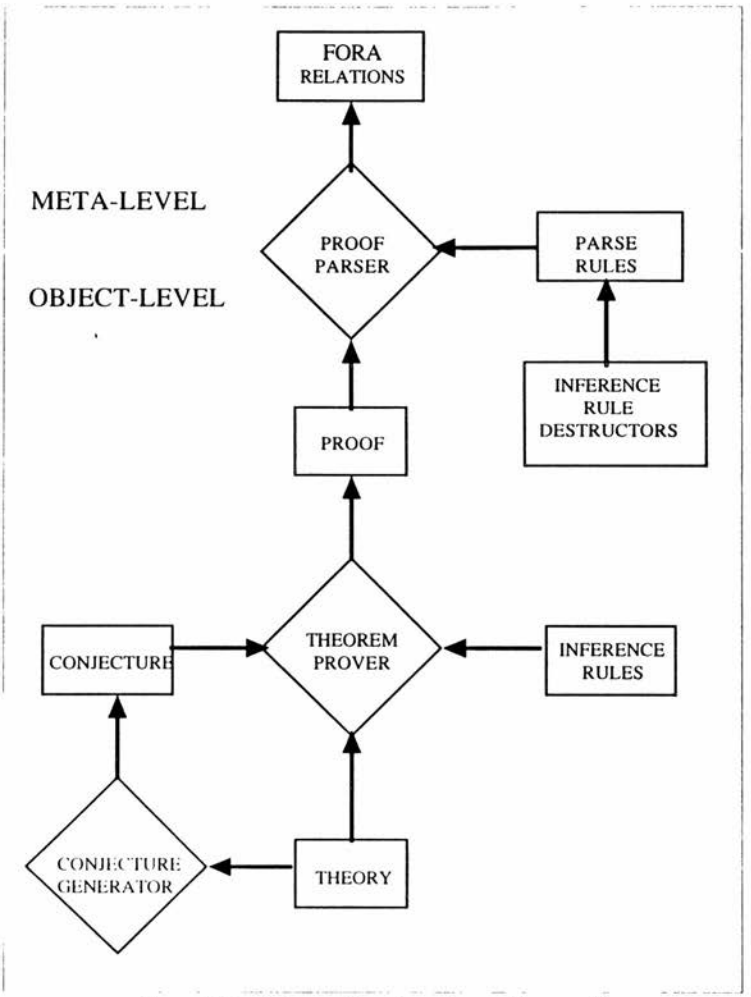


Figure 6.3 : Automated mark-up by parsing proofs. The diamond-shaped boxes represent programs and the rectangular boxes represent structures manipulated or used by the programs. The uppermost section of the diagram represents the meta-level, thus arrows pointing upwards indicate abstraction.

The principal achievement here is the analysis of proof steps without any knowledge of the object-level syntax (*eg:* & and →) or decomposition of compound propositions. In other words, the parser does not know what the rule definitions are, and simply treats each one as a transformation from one sequent to a set of one or more (when the proof branches) sequents.

The proof parsing rules will apply to any proof in which the state of the theory and the conjecture is explicit at each inference step, so they are not restricted to a particular theorem prover. This has been demonstrated with the sequent calculus theorem prover described earlier, augmented from the propositional case to full first order predicate logic. Without any alteration at all to the proof-parsing rules, or sequent destructors, full FOPL proofs can be parsed completely using the same parse rules as required for propositional logic proofs.

The parsing rules are implemented in Prolog, and included as appendix A. Each of these parse rules takes a single inference rule application and generates one or more FORA relations to describe it. Parsing takes as input the complete proof and generates a complete FORA representation of the reasoning steps.

Definition : *Given a proof* Π, *which is a set of inference rule applications,*
$\exists F$, *a set of FORA relations, such that*
parsed_proof(Π, F*)* iff
$\Pi = \varnothing$ *and* $F = \varnothing$
or, $\forall I \in \Pi$, $\exists S$, *parse(I, S) and* $F \supset S$.

Definition : Given an inference rule application , I,
and *a set of proof-parsing rules,* R,
$\exists S$, *a set of FORA relations, such that parse(I, S)* iff
$\exists r \in R$, *and* $r : I ==> S$

The parse rules are as follows. Note that in a sequent, the succedent is the current conjecture, and the antecedent is the current state of the theory. These descriptions are given as if the proof is generated by backward chaining, so the changes to the sequents at each inference step are described as changes moving from the conclusion of the inference figure (below the line) up to the antecedent of the inference figure (above the line). The conventional inference rule names (I use the conventional natural deduction nomenclature) refer to *introductions* and *eliminations* of formulae reading down the proof, or forward chaining. However, a proof can be generated or read in either direction, and I have adopted the approach of describing each proof step upwards because it leads more easily to the interpretation of the proof step as part of an argument in which the conclusion is justified by some reasoning step.

For example, if the conjecture does not change during a proof step, but the theory does, then any addition to the theory must have been deduced from (can be justified by) whatever has been removed from the theory. Likewise, when a conjecture is removed, it must be deducible from (is justified by) whatever conjecture(s) have been added plus whatever propositions have been removed from the theory. The proof parsing rules are presented in detail next.

The proof parsing rules

Rule 1 : If the succedent of the sequent has changed, but the antecedent has not, then the succedents which have been removed must be justified by the succedents which have been added.

Example : *and-introduction* has this form.

$$\textit{and-introduction}: \quad \frac{\Gamma \vdash \phi, \Delta \qquad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \phi \,\&\, \psi, \Delta}$$

Here, reading up the inference figure, the conjunction $A\&B$ has been removed from the conjecture set (succedent) and A and B have been added to the two new succedents. The theory (antecedent) has not changed during this inference step. Hence this rule generates the FORA relation :

justification (A&B, [A,B]).

Rule 2 : If the succedent of the sequent has changed, and no items have been added to the antecedent, then the succedents which have been removed must have been derivable from the new succedents plus any items in the antecedents which have been removed.

Example : *direct* has this form.

$$\textit{direct:} \quad \frac{}{\Gamma, \phi \vdash \phi, \Delta}$$

This rule reads the conjecture directly from the theory and therefore results in a leaf node in the proof tree. The sequent set in the antecedent of the inference figure is empty, therefore both the antecedent and succedent of the sequent have changed, but nothing has been added to the antecedent. The conjecture is thus justified by the entire antecedent set, resulting in :

justification(ϕ, [ϕ | Γ])

Rule 3 : If no succedents have been eliminated, but new succedents have been generated, and the antecedent has changed, then any additions to the antecedent must be justified by the eliminated elements in the antecedent plus the new succedents.

Example : *implication-elimination* has this form.

$$\textit{implication\_elimination} : \frac{\Gamma \vdash \phi, \Delta \qquad \Gamma, \psi \vdash \Delta}{\Gamma, \phi \rightarrow \psi \vdash \Delta}$$

The implication elimination rule branches the proof tree, removing an implication from the theory in both branches. In the left hand branch, the premise of the implication is added as a new conjecture to prove. The right hand branch continues with the same conjecture set but the conclusion of the implication is added to the theory. Hence, no succedents have been eliminated but ϕ (the premise of the implication) has been added to the succedent in the left hand branch. The conclusion, ψ, is an addition to the antecedent, and the implication, $\phi \rightarrow \psi$, has been removed from the antecedent. The third parse rule therefore justifies ψ with $\phi \rightarrow \psi$ and ϕ, giving

justification (ψ, [$\phi \rightarrow \psi$, ϕ])

Rule 4 : If the succedents have changed and there's an addition to, but no removal from, the antecedent of the sequent, then the old succedents must have been justified by a further justification - of the new succedent by the additions to the antecedent, ie: this is a meta-meta-level relation.

Example : *implication-introduction* has this form.

$$\textit{implication\_introduction} : \frac{\Gamma, \phi \vdash \psi, \Delta}{\Gamma \vdash \phi \rightarrow \psi, \Delta}$$

This inference rule handles the situation when an implication is one of the conjectures by generating a sub-proof in which the conclusion of the implication is added to the conjecture set and the proof attempted with the premise of the implication as a new assumption in the theory. The parse rule applies because the succedent of the sequent has changed with the removal of the implication and the addition of the conclusion of the implication. Nothing is removed from the antecedent of the sequent, and the premise of the implication is added. The implication is thus justified by the fact that its conclusion is justified by its premise, leading to the following meta-meta-level statement :

justification($\phi \rightarrow \psi$, [justification(ψ, [ϕ])])

Rule 5 : If the succedents have not changed, but the antecedent has changed and the proof has not branched, then any new items in the antecedent must be derivable from the items which have been removed from it.

Example : *and-elimination* has this form.

$$\textit{and-elimination}: \quad \frac{\Gamma,\ \phi,\ \psi \vdash\ \Delta}{\Gamma, \phi \,\&\, \psi \vdash\ \Delta}$$

Here the conjunction is removed from the theory, and replaced by the two conjuncts. The succedent of the sequent is unchanged, the proof does not branch, and therefore the conjunction which is removed must be justified by the two conjuncts which are added, giving the following justification :

justification($\phi \,\&\, \psi$, [ϕ, ψ])

Rule 6 : If the succedents have not changed in the proof step, and the proof branches, with the same items removed from the antecedent in both branches but different additions in each branch, then the elements in the succedent each have a justification consisting of the thing removed from the antecedent, together with two meta-level justifications representing each branch of the proof, ie: this rule says that the hypothesis is justified by generation of two sub proofs, or alternatively, that this is the point in the proof where assumptions are discharged.

Example : *or-elimination* has this form in general.

$$\textit{or-elimination}: \quad \frac{\Gamma, \phi \vdash\ \Delta \qquad \Gamma, \psi \vdash\ \Delta}{\Gamma,\ \phi \vee \psi \vdash\ \Delta}$$

This inference rule allows the conjecture set to be deduced from a theory, Γ, plus a disjunction if the same conjecture set can be deduced from the theory plus either side of the disjunction. In other words it generates two sub-proofs for the conjecture set. The parse rule therefore must generate meta-meta-level relations justifying each conjecture by the disjunction, plus statements that the conjecture is justified by each side of the disjunction. So suppose that the conjecture δ is an element of Δ, then this parse rule generates the following justification for it :

justification(δ, [$\phi \vee \psi$, justification(δ, [ϕ | Γ]), justification(δ, [ψ | Γ])])

Rule 7 : This rule is similar to rule 6, except that in the left-hand-branch of the proof tree, no new derived propositions are generated and added to the theory, and therefore the only interesting justification occurs in the right-hand-branch of the proof tree. In other words, the hypothesis is justified by generation of a sub-proof in the

right-hand branch of the proof.

Example : *or-elimination* sometimes has this form. Specifically it has this form when the left-hand disjunction is already a member of the theory. Using the same example as for rule 6, this rule applies when ϕ is already a member of Γ, and the resulting justification is thus :

justification(δ, [φ ∨ ψ, justification(δ, [ψ | Γ])])

Rule 8 : This rule is the counter-part to rule 7, in which the hypothesis is justified by generation of a sub-proof in the left-hand-branch of the proof tree.

Example : *or-elimination* sometimes has this form. Specifically it has this form when the left-hand disjunction is already a member of the theory. Using the same example as for rule 6, this rule applies when ψ is already a member of Γ, and the resulting justification is thus :

justification(δ, [φ ∨ ψ, justification(δ, [φ | Γ])])

The final three rules are only invoked to parse or-elimination. The other rules have more general applicability. For example, rule 1 parses and-introduction, or-introduction, and equivalence-introduction. Rule 5 parses and-elimination, equivalence-elimination and all-elimination. Rule 3 can also parse the rule for abduction, as despite being logically invalid, it has the same sequent structure as implication-elimination.

For an example of proof parsing explained in detail, see section 6.4.3, in which a full first order logic proof from the aflatoxin example is parsed. Here is a very simple example of the output of a simple propositional logic proof parsed by the parse rules.

Conjecture : d
Theory : a&b, b→c, c→d

Proof :
rule(and_elim, [[a, b, b→c, c→d] ⊢ [d]], [a&b, b→c, c→d] ⊢ [d])
rule(imp_elim, [[a, b, c→d] ⊢ [b], [c, a, b, c→d] ⊢ [d]], [a, b, b→c, c→d] ⊢ [d])
rule(direct, [], [a, b, c→d] ⊢ [b])
rule(imp_elim, [[c, a, b] ⊢ [c], [d, c, a, b] ⊢ [d]], [c, a, b, c→d] ⊢ [d])
rule(direct, [], [c, a, b] ⊢ [c])
rule(direct, [], [d, c, a, b] ⊢ [d])

Meta-level relations generated by the parser :
justification(d, [c, c→d], imp_elim)
justification(c, [b, b→c], imp_elim)

justification(b, [a&b], and_elim)
justification(a, [a&b], and_elim)

Look at the first rule in the proof :

rule(and_elim, [[a, b, b→c, c→d] ⊢ [d]], [a&b, b→c, c→d] ⊢ [d])

This inference rule is parsed by recognising that *and-elimination* involves no branching, no change to the succedent, and a removal and two additions to the antecedent. Parse rule 5 is therefore applicable. This results in two justifications, of the two additions (a and b), by the item removed *(a&b)*. Thus each side of the conjunction is justified by the conjoined statement.

justification(b, [a&b], and_elim)
justification(a, [a&b], and_elim)

Now look at the second rule in the proof :

rule(imp_elim,[[a,b, c→d] ⊢ [b], [c,a,b,c→d] ⊢ [d]], [a,b, b→c, c→d] ⊢ [d])

Implication-elimination is a branching rule, in which there is a new succedent and both removals and additions to the antecedent. Parse rule 3 is therefore applicable. The original succedent (d) is retained in the right-hand-branch, and the left-hand-branch has a new succedent (the premise of the implication, b). The implication statement ($b \rightarrow c$) is removed from the antecedent in both branches. In the right-hand-branch the conclusion of the implication (c) is added to the antecedent. The resulting justification is that this conclusion (c) is justified by the new succedent (b) plus the item removed from the antecedent ($b \rightarrow c$).

justification(c, [b, b→c], imp_elim)

It should now be clear that proofs are marked-up in a significantly more abstract manner than was possible using the dynamic mark-up process.

Section 6.2.5 stated five problems with dynamic mark-up. The first was that it involved alterations to the object-level inference mechanism. This has been solved by working entirely with the output of the inference process (the proof). I have still assumed that the object-level inference mechanism is capable of producing a proof tree.

The second problem was that the mark-up process was not looking at the pattern of reasoning produced by the inference mechanism, but was actually using hard-coded knowledge about how to interpret each particular inference rule at the meta-level. This problem has also been addressed by parsing the structural changes to the sequents at each step in the proof, without reference to which particular inference rule has produced these changes.

The third problem was that it is necessary to be able to gloss over details in the proof when describing it as a general argument. This can be achieved entirely at the meta-level in FORA and is the subject of the next section.

The fourth problem was the limitation of requiring that proofs are achievable automatically during mark-up. The proof-parsing method has also addressed this problem by starting from a completed proof, with no restriction as to how it has been derived. Proof parsing will parse proofs which could have been created by hand, or automatically or partially automatically with some user involvement.

The final problem was that mark-up would be different for each new representation language and inference mechanism at the object-level. Proof parsing has also tackled this problem by reasoning about the form of the inference steps rather than their content, and section 6.4 demonstrates that the object-level inference system can be altered (from propositional to full first order predicate logic) without requiring any changes at all to the parse rules. Assessment of how widely applicable this approach is would be an interesting area of further work.

6.3.2. Merging relations

It is possible to carry the abstraction process even further, by merging the FORA relations which result from the parsing process to produce more sketchy outlines of the proof. Figure 6.4 illustrates this process.

The idea here is that a proof, or chain of inference, involves steps which are not all equally central to the argument. Some proof steps carry out syntactic manipulation which may seem not really to progress the proof a great deal, whilst at other points the proof seems really to have 'moved on'. For example, if the conjunction $A\&B\&C$ is in the theory, and so is $B \rightarrow D$, deduction of D will involve an *implication-elimination* step and also two *and-elimination* steps. The proof makes significant progress as a result of the three rule applications taken together, so they could usefully be merged into a summary of this stage of the proof.

When summarising the central argument in a proof or inference chain, there are therefore likely to be steps which can be glossed over, and details of the object-level syntactical manipulation which can be ignored. I am not claiming here that FORA can possibly make the sorts of judgements required to decide when a proof step is significant and when it is not - apart from anything else, in FORA each statement is just a proposition *name* and the syntax is not available for reasoning about. However, FORA does provide some useful machinery for combining several proof steps into one and thus giving a more succinct description of the reasoning involved.

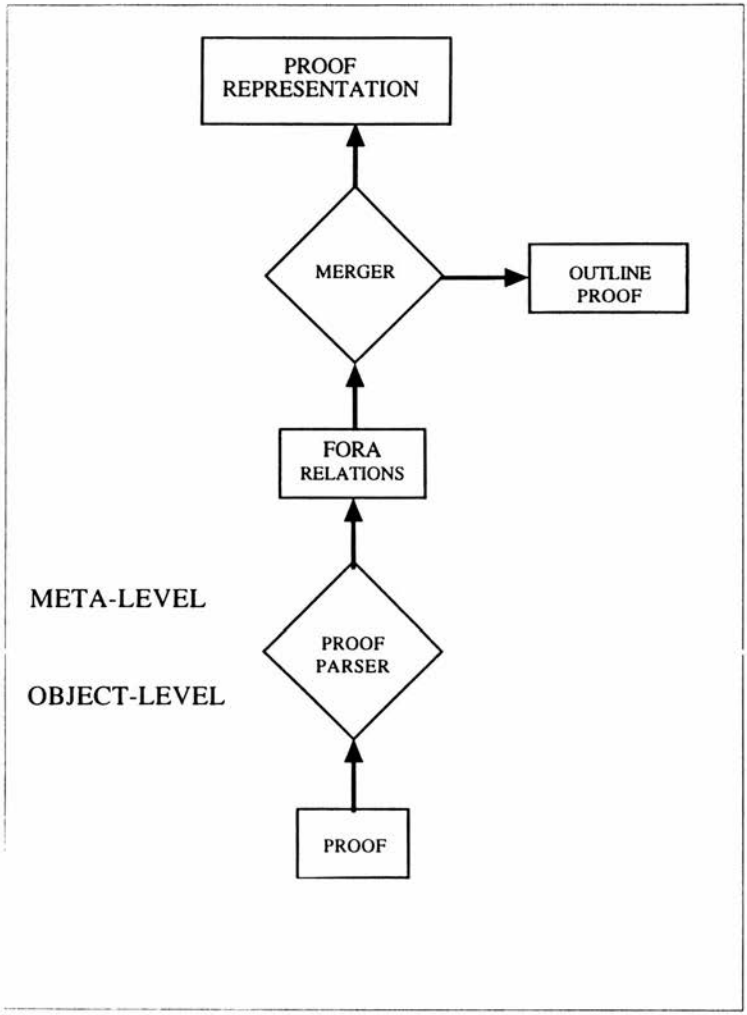


Figure 6.4 : Further meta-level analysis of a proof by merging and outlining.

For example, take the two inference steps parsed in section 6.3.1 :

justification(c, [b, b→c], imp_elim)

justification(b, [a&b], and_elim)

Of these, the deduction of b from $a\&b$ seems fairly obvious, whilst the application of modus ponens by the elimination of the implication seems rather more substantive. Also from the point of view of producing an argument for c, the two steps can be neatly combined to give the following justification which captures the gist of the argument.

justification(c, [b→c, a&b])

In merging, the justification relations are analysed and more abstract relations are deduced by merging two or more relations into one. The merging algorithm is very simple. If Q is one of the supports for P in P's justifications, and there is a justification for Q too, then merging results in a new justification for P, which is the union of Q's supports with the rest of P's supports.

Definition : merge(justification(P, S1), justification(Q, S2)) gives
justification(P, S3) iff

$$S1 = (Q \cup S) \ \& \ S3 = (S \cup S2)$$

For example :

justification(a, [b,c,d]) and
justification(c, [e,f,g])

would be merged to give

justification(a, [b,e,f,g,d]).

In this way, the intermediate details in a proof are ignored.

Merging is carried out exhaustively on the set of justifications resulting from parsing a proof, to give the complete FORA *proof_representation.*

Definition : Given a proof, Π, and a set of FORA relations F,
such that parsed_proof(Π, F),
$\exists R.$ *proof_representation(Π, R)* iff
$\forall J.\ (J \in F) \rightarrow (J \in R)$ and
$\forall J\ .\ (\exists J1 \in R$ *and* $\exists J2 \in R$ *and merge(J1, J2) = J)* $\rightarrow J \in R$

One of the merged justifications in the proof representation is particularly significant. It is called the *outline proof.* It is the justification of the conclusion of the proof by listing all of the assumptions used in it (*ie:* all the leaf nodes of the proof tree, which are the statements in the initial theory, or axioms, involved in the proof). This outline proof states which subset of the underlying theory the conclusion is dependent upon. It is important because it is the justification which results from removing all of the derived or intermediate conclusions in the proof. Merging is abstraction, blurring out one node in the proof tree at a time, while the proof outline is the end result of

merging. It is the relation which captures the entire proof in a single justification step - the proved conjecture is justified by the initial assumptions.

Definition : Given a theory, T, a conjecture, C, and a proof, Π, of C from T, such that proof_representation(Π, R), then
∃P. outline _proof(Π, P), iff
$P \in R \ \& \ P = justification(C, S) \ \& \ subset(S, T)$.

In the example from section 6.3.1, the following three relations are generated by merging.

justification(c, [b→c, a&b])
justification(d, [c→d, b, b→c])
justification(d, [c→d, b→c, a&b])

The third of these justifications is the outline proof, because all three of the supports are elements of the original theory, and no additional derived or intermediate conclusions are included, so the support set is a subset of the theory. FORA's output is as follows :

Subset of theory needed for proof :
c→d
b→c
a&b

This section has described how FORA can take a parsed proof and, working entirely at the meta-level, continue to provide other useful abstractions of the proof structure, using simple merging and outlining techniques.

6.4 Illustration of proof-parsing in terms of the aflatoxin debate

In this section I illustrate automated mark-up by showing how the argument for the FDA's policy can be automatically abstracted from a first order predicate calculus proof. This illustration has several stages.

Firstly, a first order predicate calculus theory is given which is the knowledge base to be marked up. Secondly, the conjecture, which states that the FDA policy should set the maximum allowed level of aflatoxins at 20 parts-per-billion, is proved from the theory. Thirdly, the proof of the FDA policy conjecture is parsed by the proof-parser, resulting in a marked-up proof-representation of the subset of the knowledge base used in the proof, consisting of a set of FORA relations. Finally, the merger and proof outliner extract the key steps in the argument.

6.4.1 A first order predicate logic knowledge base about aflatoxins

This theory is the formalised version of the FDA policy argument, as given by Fox in [Fox 94], in a slightly altered syntax (in particular with the addition of quantifiers). Some comments are included in the theory - these begin with a percentage (%) sign. In Fox's representation, each statement in the theory has a name, and these are given before each statement, indicated by the double percentage sign (%%).

% First there are five ground assumptions about aflatoxins

%% data1
current_assessment(aflatoxins, cancer, humans)

%% causal1
causes(aflatoxins, cancer, animals)

%% risk1
indicates_risk(chemicals, animals, humans)

%% compounds1
kind_of(aflatoxins, chemicals)

%% aflatoxins1
minimum_detectable_level(aflatoxins, '20ppb')

% Next there are several implications in which variables are used,
% which illustrate the inclusion of quantifiers in the FOPL theory

%% extrap1
$\forall$ *agent,* $\forall$ *species1,* $\forall$ *species2,* $\forall$ *class,*
($\exists$ *path, current_assessment(agent, path, species2)) &*
kind_of(agent, class) &
indicates_risk(class, species1, species2))
$\rightarrow$ *can_extrapolate(agent, species1, species2)*

%% extrap2
$\forall$ *agent,* $\forall$ *path,* $\forall$ *species1,* $\forall$ *species2,*
(current_assessment(agent, path, species2) &
causes(agent, path, species1) &
can_extrapolate(agent, species1, species2))
$\rightarrow$ *causes(agent, path, species2)*

%% safe1
$\forall$ *agent,* $\forall$ *species, current_assessment(agent, cancer, species)*
$\rightarrow$ *no_safe_level(agent, cancer, species)*

%% minim1
$\forall$ *compound,* $\forall$ *path,* $\forall$ *species,* $\forall$ *min,*
(no_safe_level(compound, path, species) &
minimum_detectable_level(compound, min))
$\rightarrow$ *maximum_acceptable_level(compound, path, species, min)*

%% fda1
∀agent, ∀path, ∀level,
(causes(agent, path, humans) &
maximum_acceptable_level(agent, path, humans, level))
→policy_level(agent, level)

6.4.2 Proving the FDA policy conjecture

The following conjecture :

policy_level(aflatoxins, '20ppb')

is proved using the sequent calculus theorem prover described in section 6.2.3. For the full proof, which runs to several pages, see appendix B.

In order to prove this conjecture, the theorem prover needed to be upgraded from a propositional theorem prover to a full first order calculus theorem prover. This was done by adding four inference rules for handling quantified expressions :

(i) introduction of the universal quantifier : *all-intro*
(ii) elimination of the universal quantifier : *all-elimination*
(iii) introduction of the existential quantifier : *exists-introduction*
(iv) elimination of the existential quantifier : *exist-elim.*

In fact only two of these (*all-elimination* and *exists-introduction*) are needed in the proof. The object-level proof consists of 40 inference steps. This includes 12 applications of the *direct* rule, 17 applications of *all-elimination* (one for each universal quantifier in the theory), and the remainder are all applications of *exists-introduction, and-introduction* and *implication-elimination.*

In addition, some variable handling was needed in order to guide the instantiation of variables by these four rules. Two types of instantiation are possible - either by matching terms on similar terms in the theory, and deducing plausible constant values, or by asking the user to give an instantiation. This avoids the system generating implausibly large search trees by generating partial proofs with inappropriate variable instantiations.

A summary of the proof is as follows :

1. *all-elimination* is repeatedly applied to the *extrap1* implication in the theory until all the universal quantifiers are replaced by constants.

2. *implication-elimination* removes the implication, and branches the proof tree, so

that one branch now has the conjunction of *extrap1's* antecedents as its goal to prove (the succedent of the sequent).

3. On that branch, *and-introduction* branches the proof tree again to separate the two parts of the conjunction.

4. The left-hand branch's goal (succedent) is the existentially quantified

∃path, current_assessment(aflatoxins, path, humans)

so the *exists-introduction* rule is applied and instantiates the existentially quantified variable 'path' with 'cancer'. (It does this by matching on the theory). The resulting instantiated succedent is found by an application of the *direct* rule.

5. The proof returns to the right-hand-side of the conjunction branch from Step 3. One more application of *and-introduction* produces two further branches, whose succedents are

kind_of(aflatoxins, chemicals).

indicates_risk(chemicals, animals, humans).

Both of these are found by application of the *direct* rule as they are facts in the theory.

6. The left-hand-branch resulting from the *implication-elimination* application (Step 2) is now complete, so the right-hand-branch continues, with

can_extrapolate(aflatoxins, animals, humans)

in the antecedent of the sequent, the *extrap1* implication removed from the antecedent, and the FDA policy conjecture still as the goal proposition (succedent) of the sequent. (Elsewhere I describe it as an 'intermediate conclusion' which is added to the theory during the proof but is not part of it at the start).

7. The next phase of the proof is similar to steps 1-6, as the *extrap2* implication is handled. *All-elimination* is repeatedly applied to the *extrap2* implication in the theory until all the universal quantifiers are replaced by constants. Then *implication-elimination* branches the proof tree and *and-introduction* branches it again. There is no equivalent of step 4 this time as the existential quantifier is not used. A second *and-introduction* and three applications of *direct* complete this branch of the proof. The right-hand-branch continues, with the *extrap2* implication removed from the antecedent and

causes(aflatoxins, cancer, humans)

added to the antecedent.

8. A similar process applies to decomposing the *safe1* implication to add

no_safe_level(aflatoxins, cancer, humans)

to the antecedent.

9. The same process occurs with *minim1* to derive
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)

10. Finally, the *fda1* rule is used, adding
policy_level(aflatoxins, 20ppb)
to the antecedent. As it is the succedent too it can be derived using *direct*, thus completing the proof.

The proof is output in the form shown in appendix B. This output includes some annotation and layout information. Internally the proof is just the list of the inference steps. This list is next passed to the proof-parser, as explained in the next section, in order to generate the argument structure from the proof. No additional information is required by the proof parser.

6.4.3 Parsing the proof

By applying the parse rules to each step in the proof in turn, the set of FORA relations shown in appendix C are derived. Note that for clarity they have an additional argument stating which type of inference rule they resulted from.

The parse rules used are those described in section 6.3.1, and the proof is parsed recursively, one inference rule at a time. The proof involves five different inference rules, so five different types of parsing occur. Each inference rule type is always parsed by the same parse-rule. Here I shall describe the effect of parsing each type of inference rule used in the proof of the FDA policy conjecture.

1. *all-elimination :* This rule is parsed by the fifth parse-rule (see appendix A for the full Prolog version of the parse rules). The parse succeeds because :
- there is no hypothesis change (*ie:* the succedent of the sequent is the same above and below the line, or in other words, before and after the inference step);
- the theory does change (*ie:* the antecedent of the sequent is altered by the inference step);
- the proof tree does not branch at this step.

The justification which results is the new addition to the antecedent, justified by the term(s) removed from the antecedent. In this case, the term resulting from instantiating the universally quantified variable is added. The universally quantified term is removed. Therefore the result is that the instantiated term is supported by the universally quantified term. For example, in the inference rule application shown in figure 6.5, the variable *agent* is instantiated to *aflatoxins*. The term which alters in the inference step is emboldened. The parse rule picks out the two highlighted terms and produces the following justification.

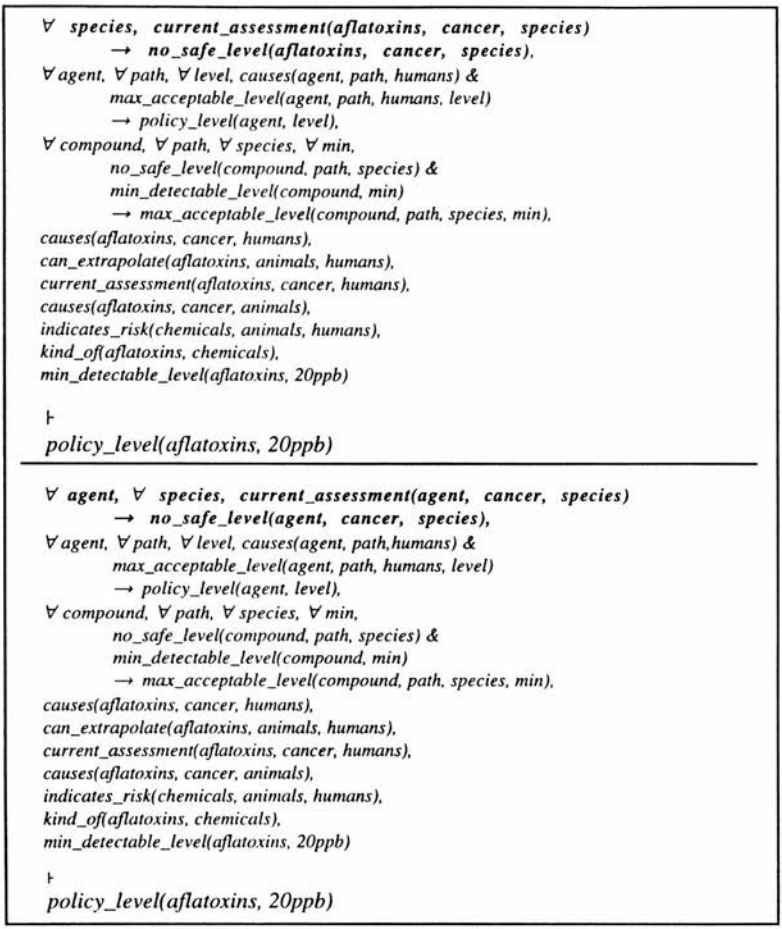


Figure 6.5: Application of inference rule : all_elimination

justification(all(species, current_assessment(aflatoxins, cancer, species)
→ no_safe_level(aflatoxins, cancer, species)),
[all(agent, all(species, current_assessment(agent, cancer, species)
→ no_safe_level(agent, cancer, species)))], all_elim)

It is important to note that the parse rule uses general structural changes brought about by the inference step, rather than using knowledge about the *particular* inference rules.

2. *implication-elimination :* Implication-elimination rule instances are parsed by the third parse rule (again see appendix A for the details of the parse rule). The parse succeeds because :

- there is a new hypothesis (succedent of the sequent) added but the prior hypothesis is also retained, *ie:* the succedent is not removed entirely (this is because the proof branches, with the original succedent on one branch, and a new one on the other branch);
- the theory (antecedent of the sequent) changes, with both additions and removals.

Any addition to the theory is justified by the union of the removals and the new hypothesis. In this case, the addition to the antecedent in the right-hand-branch is the conclusion of the eliminated implication. The new hypothesis in the left-hand-branch is the premise of the implication. The implication is removed from the antecedent in the right-hand-branch. Therefore the conclusion is justified by the premise, and the implication, resulting in a justification which looks just like an application of *modus ponens*. Again it must be noted that this is achieved solely by looking at the structural changes in the sequents before and after the inference step, and *not* with any knowledge of implication elimination in particular.

An example, showing parsing of an application of implication-elimination, is given in figure 6.6.

The highlighted changes result in the following justification :

justification(causes(aflatoxins, cancer, humans),
[current_assessment(aflatoxins, cancer, humans) &
causes(aflatoxins, cancer, animals) &
can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans) &
causes(aflatoxins, cancer, animals) &
can_extrapolate(aflatoxins, animals, humans)
→ causes(aflatoxins, cancer, humans)])

rule : implication elimination

[can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
∀ agent, ∀ species,
current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species),
∀ compound, ∀ path, ∀ species, ∀ min,
no_safe_level(compound, path, species) &
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min),
∀ agent, ∀ path, ∀ level,
causes(agent, path, humans) &
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level)]

*|- **current_assessment(aflatoxins, cancer, humans) & causes(aflatoxins, cancer, animals) & can_extrapolate(aflatoxins, animals, humans)***

causes(aflatoxins, cancer,humans),
can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
∀ agent, ∀ species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
∀ compound, ∀ path, ∀ species, ∀ min,
no_safe_level(compound, path, species) &
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min))))),
∀ agent, ∀ path, ∀ level, causes(agent, path,humans) &
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level)))))]

|- policy_level(aflatoxins, 20ppb)

---

current_assessment(aflatoxins, cancer, humans) & causes(aflatoxins, cancer, animals) & can_extrapolate(aflatoxins, animals, humans) ->causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
∀ agent, ∀ species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
∀compound, ∀ path, ∀ species, ∀ min,
no_safe_level(compound, path, species) &
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min))))), all(agent,
∀ path, ∀ level, causes(agent, path, humans) &
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level)))))]

|- policy_level(aflatoxins, 20ppb)

By now it should be becoming clear that the argument structure which is extracted from each inference step is a useful clarification of the significance of each step in the proof. Parsing picks out the pertinent change at each step, and ignores the rest. The achievement here is that this is done at the meta-level in a way which is entirely independent of the object-level syntax.

3. *direct :* All instances of the *direct* inference rule are parsed by the second parsing rule. Parsing succeeds because :

- There is a change to the hypothesis (succedent of the sequent); and
- There is a removal from, but no addition to, the theory (antecedent).

The justification which results is that the old succedent is justified by the union of the new succedent and the removed antecedents. In the case of *direct,* both the new succedent and new antecedent are empty. Hence the original succedent (which is read-off directly from the antecedent) is justified by the antecedent.

Figure 6.7 shows as an example the final application of *direct* in the proof of the FDA policy conjecture. Note that there is nothing 'above the line'.

The following justification results.

justification(policy_level(aflatoxins, 20ppb),
[policy_level(aflatoxins, 20ppb),
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb)])

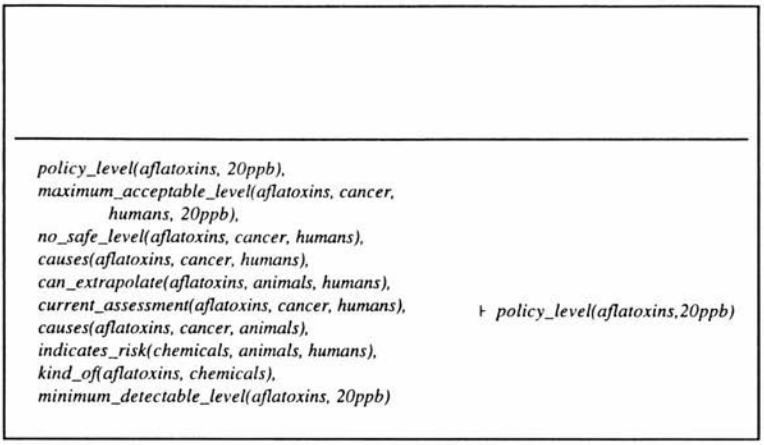


Figure 6.7: Application of inference rule : direct

4. Finally, both *and-introduction* and *exist-introduction* are parsed by the first parsing rule. This serves to demonstrate that the parsing is not inference-rule specific, as both these rules have the same structural characteristics recognised by this single parsing rule. The parse succeeds in both cases because :

- The hypothesis (succedent) is altered by the step - both removing a hypothesis and adding new one(s) (note that in and-introduction the inference step branches the proof, and in exists-introduction it does not);
- The theory (antecedent) is not altered by the inference step - the changes are entirely to the succedent of the sequent.

The resulting justification is that the succedent(s) which were removed, must have been justified by the new succedent(s). So, for *exist-elim*, the existentially quantified term is justified by the term with its variable instantiated to some constant. For *and-introduction*, the conjunction is justified by the two conjuncts. Figures 6.8 and 6.9 give illustrations of both cases.

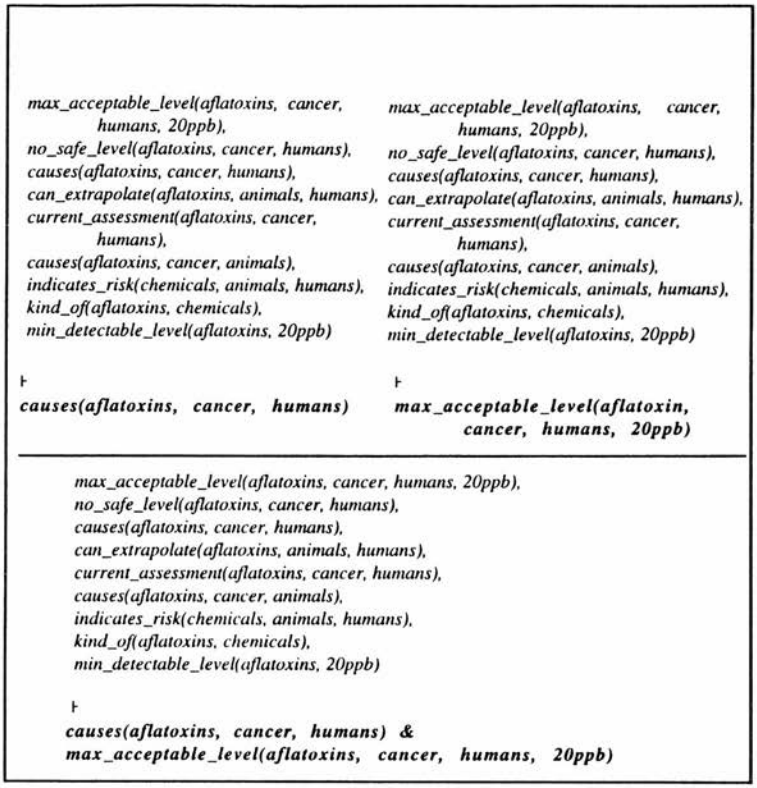


Figure 6.8: Application of inference rule : and_introduction

current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
min_detectable_level(aflatoxins, 20ppb),
$\forall$ *agent,* $\forall$ *path,* $\forall$ *species1,* $\forall$ *species2, current_assessment(agent, path, species2)*
& causes(agent, path, species1) & can_extrapolate(agent, species1, species2)
$\rightarrow$ *causes(agent, path, species2),*
$\forall$ *agent,* $\forall$ *species, current_assessment(agent, cancer, species)*
$\rightarrow$ *no_safe_level(agent, cancer, species),*
$\forall$ *compound,* $\forall$ *path,* $\forall$ *species,* $\forall$ *min, no_safe_level(compound, path, species)*
& min_detectable_level(compound, min)
$\rightarrow$ *max_acceptable_level(compound, path, species, min),*
$\forall$ *agent,* $\forall$ *path,* $\forall$ *level, causes(agent, path, humans)*
& max_acceptable_level(agent, path, humans, level)
$\rightarrow$ *policy_level(agent, level))))*

$\vdash$ ***current_assessment(aflatoxins, cancer, humans)***

---

current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
min_detectable_level(aflatoxins, 20ppb),
$\forall$ *agent,* $\forall$ *path,* $\forall$ *species1,* $\forall$ *species2, current_assessment(agent, path, species2)*
& causes(agent, path, species1) & can_extrapolate(agent, species1, species2)
$\rightarrow$ *causes(agent, path, species2),*
$\forall$ *agent,* $\forall$ *species, current_assessment(agent, cancer, species)*
$\rightarrow$ *no_safe_level(agent, cancer, species),*
$\forall$ *compound,* $\forall$ *path,* $\forall$ *species,* $\forall$ *min, no_safe_level(compound, path, species)*
& min_detectable_level(compound, min)
$\rightarrow$ *max_acceptable_level(compound, path, species, min),*
$\forall$ *agent,* $\forall$ *path,* $\forall$ *level, causes(agent, path, humans)*
& max_acceptable_level(agent, path, humans, level)
$\rightarrow$ *policy_level(agent, level))))*

$\vdash$ $\exists$ ***path. current_assessment(aflatoxins, path, humans)***

Figure 6.9: Application of inference rule : exists_introduction

In figure 6.8, the following justification is derived from the succedents of the sequent prior to inference, and of both branches after the inference rule application (above the line) :

justification(causes(aflatoxins, cancer, humans) &
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
[causes(aflatoxins, cancer, humans),
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)])

The same parse rule parses the non-branching, *exists-introduction* inference rule application in figure 6.9. The following simple justification is derived :

justification(∃path. current_assessment(aflatoxins, path, humans)),
[current_assessment(aflatoxins, cancer, humans)])

Having abstracted the main steps of the proof, FORA can be used to carry the abstraction process even further, as illustrated in the next section.

6.4.4. Further abstraction in FORA, merging argument steps and proof-outlining

The set of relations produced by the parse-rules give the argument structure of every step in the proof. This section shows how FORA can merge the steps in the argument, producing an outline proof.

First FORA merges the justification steps just derived by proof parsing. Two justifications of the following forms :

justification(P, [....Q.....])
justification(Q, [....])

can be merged, to give a new justification for P whose support is the union of its original support (minus Q), and the support for Q. The result of merging the justifications derived by parsing the proof of the FDA's policy is given in appendix D. A summary of the merged rules and the outline proof is given here.

At the start of the merging process there is exactly one justification per inference rule application in the proof. The merging process achieves proof abstraction by eliminating the intermediate propositions derived by application of inference rules. Two justifications which represent two inference rule applications are merged, resulting in the intermediate conclusion being abstracted away. This increases clarity. Five examples are given to illustrate this.

(i) Merging combines the repeated applications of *all-elimination* which instantiate the universally quantified variables in the rules, to give a single justification of the fully instantiated rule instance from the universally quantified version, abstracting from all the intermediate stages when the rule instance is partially instantiated. For example, here is the merged justification for the rule called *extrap2*.

justification(current_assessment(aflatoxins, cancer, humans) &
causes(aflatoxins, cancer, animals) &
can_extrapolate(aflatoxins, animals, humans)
→ causes(aflatoxins, cancer, humans),
[∀agent, ∀path, ∀species1, ∀species2,
current_assessment(agent, path, species2) &
causes(agent, path, species1) &
can_extrapolate(agent, species1, species2)
→ causes(agent, path, species2)]))

(ii) Merging also combines the results of merging the *all-elimination* steps with an application of *implication-elimination* which allows the conclusion of an implication to be deduced from its premise(s). For example, here is the result of merging the instantiation of the rule *safe1* and the elimination of the implication.

justification(no_safe_level(aflatoxins, cancer, humans),
[current_assessment(aflatoxins, cancer, humans),
∀agent, ∀species, current_assessment(agent, cancer, species)
→ no_safe_level(agent, cancer, species)])

(iii) Merging also combines the steps involved in introducing conjunctions, and existential quantifiers. For example, here is the result of merging the justifications from the two instances of *and-introduction*, and the *exists-introduction*, to achieve the combined conjunction, which is then used as the premise of the *extrap1* rule.

justification(exists(path, current_assessment(aflatoxins, path, humans)) &
kind_of(aflatoxins, chemicals) &
indicates_risk(chemicals, animals, humans),
[current_assessment(aflatoxins, cancer, humans),
kind_of(aflatoxins, chemicals),
indicates_risk(chemicals, animals, humans)])

(iv) A more complex example similar to the previous one involves merging an instance of *and-introduction* with an instance of *implication-elimination* to give the premise of the *minim1* rule by application of the *safe1* rule.

justification(no_safe_level(aflatoxins, cancer, humans) &
minimum_detectable_level(aflatoxins, 20ppb),
[minimum_detectable_level(aflatoxins, 20ppb),
current_assessment(aflatoxins, cancer, humans),
∀agent, ∀species, current_assessment(agent, cancer, species)
→ no_safe_level(agent, cancer, species)])

(v) As a final example, merging also combines the results of all the *all-elimination, and-introduction, exist-introduction* and *implication-elimination* inference steps which are needed to apply both the *extrap1* and *extrap2* rules and all the relevant facts, from the theory, in order to deduce that aflatoxins cause cancer in humans.

justification(causes(aflatoxins, cancer, humans),
[∀agent, ∀path, ∀species1, ∀species2,
current_assessment(agent, path, species2) &
causes(agent, path, species1) &
can_extrapolate(agent, species1, species2)
→ causes(agent, path, species2),
causes(aflatoxins, cancer, animals),
∀agent, ∀species1, ∀species2, ∀class, ∃path,
current_assessment(agent, path, species2)) &
kind_of(agent, class) &
indicates_risk(class, species1, species2)
→ can_extrapolate(agent, species1, species2),
current_assessment(aflatoxins, cancer, humans),
kind_of(aflatoxins, chemicals),
indicates_risk(chemicals, animals, humans)])

The outline proof which is generated from the proof is the single justification which sums up the deduction of the conjecture from the theory, omitting *all* of the intermediate inference steps. In this case the entire theory is used in the proof, so the justification is of the policy level conclusion, from the set of all statements in the theory. The outline proof is extracted from the complete set of merged rules, by seeking a justification of the conjecture, and then checking that its support set is a subset of the theory (*ie:* does not include any intermediate derived propositions). This outline proof can also be described as justifying the root node of the proof tree (the proved conjecture) by the complete set of leaf nodes (assumptions) in the proof tree. Here is the outline proof of the policy level conjecture.

justification(policy_level(aflatoxins, 20ppb),
[∀agent, ∀path, ∀level, causes(agent, path, humans) &
maximum_acceptable_level(agent, path, humans, level)

→ policy_level(agent, level),
∀ agent, ∀ path, ∀ species1, ∀ species2,
current_assessment(agent, path, species2) &
causes(agent, path, species1) &
can_extrapolate(agent, species1, species2)
→ causes(agent, path, species2),
causes(aflatoxins, cancer, animals),
∀ agent, ∀ species1, ∀ species2, ∀ class, ∃path,
current_assessment(agent, path, species2)) &
kind_of(agent, class) &
indicates_risk(class, species1, species2)
→ can_extrapolate(agent, species1, species2),
kind_of(aflatoxins, chemicals),
indicates_risk(chemicals, animals, humans),
minimum_detectable_level(aflatoxins, 20ppb),
∀ compound, ∀ path, ∀ species, ∀ min,
no_safe_level(compound, path, species) &
minimum_detectable_level(compound, min)
→ maximum_acceptable_level(compound, path, species, min),
current_assessment(aflatoxins, cancer, humans),
∀ agent, ∀ species, current_assessment(agent, cancer, species)
→ no_safe_level(agent, cancer, species))])

It is important to note that the merging and proof outlining processes are carried out entirely at the meta-level in FORA. This means that these processes treat each of the propositions in the justifications as *proposition names* and do not involve any decomposition of them at all. So, for example, if the justified proposition is an implication, the merging and proof outlining processes do not need to know anything about the syntax of the implication. The explanations of the merging steps just given have talked in terms of how different inference rule applications are combined, but from the point of view of FORA all that is happening is that two justifications of arbitrary propositions are combined. The fact that this combination process is actually a meaningful integration of inference steps adds weight to the thesis that abstract representation of proofs as *argument structures* is both possible and useful.

The significant point here is that what appears as a meaningful integration of inference steps is achieved by a simple merging algorithm applied at the meta-level. This suggests that we have indeed abstracted out the significant structural aspect of inference at the object-level and represented it more simply as an argument.

6.5 Summary

This chapter has described how to automate mark up of existing formal knowledge bases to FORA. In section 6.1, I explained why this is significant by arguing that it is useful to reason about the arguments contained implicitly in legacy knowledge bases. In order for this to be practicable it is necessary to be able to mark-up existing knowledge bases *automatically.*

In section 6.2, I explained why automated mark-up of formal knowledge bases (or theories of logic) requires inference and described dynamic mark-up, in which abstraction is carried out during inference by modification of the inference mechanism. There are inherent problems with having to carry out this kind of modification and so in section 6.3 an alternative approach to automated mark up was described which involves parsing completed proofs or inference chains independently of the inference mechanism itself. Additional abstraction at the meta-level leading to the extraction of a proof outline was also described.

Section 6.4 illustrated the ideas in this chapter with respect to the aflatoxin debate, and showed how proof parsing is an adaptable and effective method for automating mark-up to FORA and abstracting arguments from chains of formal inference.

Having established that it is possible to automate the mapping from object-level formal knowledge bases to FORA's meta-level representation, I turn now to the inverse mapping from FORA to the object-level. The next chapter investigates how to help a user to construct an object-level knowledge base to represent FORA argument structures.

Chapter 7

Supporting Mark-down from FORA

This chapter argues that FORA is useful because it supports the construction and maintenance of object-level knowledge bases. Some tools for this are presented and they are illustrated in terms of the aflatoxin debate.

7.1 Introduction

As introduced in chapter 3, [Fox 94] shows how to construct a knowledge base representing two of the conflicting views about the cancer risk of aflatoxins. The approach which Fox takes to constructing the knowledge base has two stages.

1. First, the arguments both for and against the FDA's policy are articulated at a high level of abstraction, using English language sentences and diagrams representing the relationships between these sentences.

2. Secondly, this high level representation is used to drive the formalisation of the content of the arguments into a formal language (the Logic of Argumentation, LA) which can be reasoned with using the Argumentation Theorem Prover (ATP) [Krause et al 95].

This chapter describes an alternative approach to this task. There is a tool within the FORA framework called *mark-down*, which takes FORA meta-level argument descriptions and uses them to guide the generation of formal object-level knowledge bases which can be used to actually carry out the patterns of reasoning. This chapter shows how articulating the structure of arguments at the meta-level allows FORA to support object-level formalisation decisions, and enables some of the formalisation process to be automated.

The object-level formal language chosen here is first order predicate logic (FOPL), rather than the logic of argumentation (LA) as used by Fox. First order predicate logic is chosen as it is a highly expressive language which subsumes many useful knowledge representation and logic programming languages. A proof of concept showing mark-down from FORA to FOPL to be possible therefore supports the suggestion that mark-down is feasible for a broad family of formal representation languages, including LA (which is an extension of part of FOPL involving labelling propositions with representations of their proofs, or arguments, in a similar manner

to the mark-up process described in Chapter 6).

Section 7.2 deals with *mark-down* to the object-level and describes the tool supporting this task, section 7.2.1 gives one possible formalisation of the argument for the FDA policy in the object-level logic, and sections 7.2.2 and 7.2.3 describe how *mark-down* supports multiple alternative formalisations. Once a knowledge base such as this has been constructed, it is useful to evaluate it, and in particular to check that certain required chains of inferences (proofs) are possible. Section 7.3 describes an *inference checker* tool for doing just this. Finally section 7.4 draws together the ideas of arguing (from chapter 5) and mark-down (from section 7.2) to produce a tool which *criticises* the object level knowledge base by pointing out vulnerable parts of the argument from which it derives. This demonstrates that putting effort into a more formal abstract representation of the arguments at the beginning pays off, not only by supporting knowledge base construction, but also by enabling knowledge base evaluation and maintenance.

7.2 Mark-down

Having articulated the arguments at the meta-level, the FORA structures provide support for further formalisation, or mark-down, to an object-level representation. Each justification relation, for example, can be viewed as an indication of an inference step which should be made possible at the object-level. The object-level is assumed to require a language which supports a greater level of granularity of formalisation. In particular, most interesting object-level languages involve internal structure of propositions, in the form of rules, or predicate logic statements. The FORA relations indicate patterns of deduction or syntactically recognisable contradiction which should be possible in the object-level knowledge base, but they do not provide any indication of the internal structure of the propositions.

The *mark-down* program supports a user in constructing elements of their object-level knowledge base, by incrementally formalising the knowledge expressed in the arguments, one relation at a time. The user needs to supply information to the program, because mark-down results in a net increase in the level of detail of the knowledge base. Unlike mark-up, therefore, it cannot in general be automated. The abstract FORA relations between proposition names need to be filled out by adding information about how the propositions will be represented, and the nature of the inferences to be made possible. The mark-down program provides book-keeping facilities and applies logical constraints to assist the user with the task of representing the argument in predicate logic. It handles the placing and scope of connectives, matching of predicates, and, to some extent, variable handling. The user is required to select schemata from a list of inference descriptions. The user also provides the

initial representation of each proposition as a predicate with arguments, but once this is provided it is reused whenever it occurs in the reasoning, greatly reducing the risk of syntax errors.

Example

In the following simple example, the justifications forming the argument for the FDA policy are formalised, all in the same way, as implications. This results in a representation similar to Fox's, though the issue of certainty factors is ignored. The benefit is that an explicit link is kept back to the FORA representation, which can be useful as shown in sections 7.3 & 7.4. This example marks-down the following five justifications.

```
justification('The FDA policy level for aflatoxins should be 20ppb',
        ['The maximum acceptable level of aflatoxins is 20ppb',
        'Aflatoxins cause cancer in humans']).
justification('The maximum acceptable level of aflatoxins is 20ppb',
        ['There is no safe level of aflatoxins',
        'The minimum detectable level of aflatoxins is 20ppb']).
justification('There is no safe level of aflatoxins',
        ['Aflatoxins cause cancer in humans']).
justification('Aflatoxins cause cancer in humans',
        ['Aflatoxins cause cancer in non-human animals',
        'Extrapolation of the cancer risk of aflatoxins from animals to humans
                is reliable']).
justification('Extrapolation of the cancer risk of aflatoxins from animals
                to humans is reliable',
        ['Aflatoxins are a kind of chemical',
        'Animals are good indicators of the cancer risk of chemicals
                to humans']).
```

Each proposition name is given an object level instantiation, represented using the two-place predicate *inst/2* . They are acquired from the user.

```
inst('The FDA policy level for aflatoxins should be 20ppb',
        fda_policy_level(aflatoxins, (20,ppb))).
inst('The maximum acceptable level of aflatoxins is 20ppb',
        max_acceptable_level(aflatoxins, (20,ppb))).
inst('Aflatoxins cause cancer in humans',
        cause(aflatoxins, cancer, humans)).
inst('There is no safe level of aflatoxins',
        no_safe_level(aflatoxins, cancer, humans)).
inst('The minimum detectable level of aflatoxins is 20ppb',
        min_detectable_level(aflatoxins, (20,ppb))).
inst('Aflatoxins cause cancer in non-human animals',
        cause(aflatoxins, cancer, animals)).
inst('Extrapolation of the cancer risk of aflatoxins from animals to humans
                is reliable',
        can_extrapolate(cancer_risk, aflatoxins, animals, humans)).
inst('Aflatoxins are a kind of chemical',
        kind_of(aflatoxins, chemical)).
inst('Animals are good indicators of the cancer risk of chemicals to humans',
```

indicator(cancer_risk, chemicals, animals, humans)).
inst('The FDA policy level for aflatoxins should not be 20ppb',
not fda_policy_level(aflatoxins, (20,ppb))).

This knowledge base of Horn clauses is the result of formalising all the justification relations in the same way, as implications.

kind_of(aflatoxins, chemical)
& indicator(cancer_risk, chemicals, animals, humans)
→ can_extrapolate(cancer_risk, aflatoxins, animals, humans).
cause(aflatoxins, cancer, animals)
& can_extrapolate(cancer_risk, aflatoxins, animals, humans)
→ cause(aflatoxins, cancer, humans).
cause(aflatoxins, cancer, humans)
→ no_safe_level(aflatoxins, cancer, humans).
no_safe_level(aflatoxins, cancer, humans)
& min_detectable_level(aflatoxins, (20,ppb))
→ max_acceptable_level(aflatoxins, (20,ppb)).
max_acceptable_level(aflatoxins, (20,ppb))
& cause(aflatoxins, cancer, humans)
→ fda_policy_level(aflatoxins, (20,ppb)).

7.2.1 Mark-down of a relation

This section explains how the example object-level theory just shown is generated using the mark-down program. The mark-down program is a *schema-application* program, similar to the schema application programs used in the construction of simulation programs in [Haggith 88], [Haggith 90a], [Robertson *et al* 91]. The general idea is that knowledge about how to formalise the meta-level relations in various ways is represented as objects or schemata. For each relation, given a particular object-level language such as FOPL, the schemata provide a variety of options which the user can choose between to formalise the relation. The user selects a relation, which allows the program to narrow down the range of schemata available. The user is then asked to choose a schema. This is made possible as one of the contents of each schema is a textual description of what it represents. Another part of the schema consists of Prolog code for generating one or a set of object-level propositions. This code is run once the user has requested a schema, generating the appropriate object-level form and adding it to the object-level knowledge base.

Schemata have the following form :

schema(Name, Relation, Object-level-form, Generation-code)

The *Name* is the natural language description of the schema which the user selects from. The *Relation* is the FORA relation to which it applies. The *Object-level-form* is the resulting structure which will be added to the object-level knowledge base if schema matching is successful. The *Generation-code* is Prolog code used to carry

out the necessary instantiation of the object-level structure.

Schema matching has three steps.

1. Match the meta-level relation to the second schema item.

2. Call the Prolog goals in the fourth schema item.

3. Check that this has successfully instantiated the third schema item and if so, add the resulting proposition(s) to the object-level knowledge base.

For example, suppose the user chooses to formalise the following meta-level relation :

```
justification('The FDA policy level for aflatoxins should be 20ppb',
        ['The maximum acceptable level of aflatoxins is 20ppb', '
        Aflatoxins cause cancer in humans']).
```

One of the schemata for justification is as follows :

```
schema( 'Justification as implication',
        justification(LHS, RHS),
        [Q -> P],
        (P = inst(LHS),
                findall(inst(X), on(X, RHS), S),
                conjunction(S, Q))).
```

This is the schema which will formalise *justification(P, [Q,R])* as *Q & R -> P*. In other words it treats the justification as being a simple implication.

This schema consists of four items :

1. The textual description of the formalisation it enables - in this case *'Justification as implication'*.

2. The meta-level representation of the relation - in this case *justification(LHS, RHS)*, which matches with the relation chosen by the user.

3. The set of formalised propositions which will be added to the object-level knowledge base - in this case *[Q -> P]*.

4. A set of Prolog goals which when run will carry out the appropriate instantiations. In this case these are :

```
(P = inst(LHS),
findall(inst(X), on(X, RHS), S),
conjunction(S, Q)).
```

The first line of these instantiates the statement justified (either by looking it up in the set of instantiations or asking the user how they wish to formalise it). The second line does the same for each of the justifications, using the Prolog predicate *findall/3*, to repeatedly call the goal *inst(X)* for each member of the set *RHS* which is the set of meta-level justifications (the membership relation is called 'on'). The resulting set of instantiated propositions is the set S. The final goal, *conjunction(S, Q)*, converts the set of instantiated propositions, S, to the conjunction of these propositions, Q, which in turn matches with the left hand side of the object-level implication.

In the case of the example, schema matching goes as follows :

1. First, the second item of the schema matches to the meta-level relation, so the following variable matching occurs :

LHS = 'The FDA policy level for aflatoxins should be 20ppb'
RHS = ['The maximum acceptable level of aflatoxins is 20ppb',
'Aflatoxins cause cancer in humans']

2. Next, the Prolog goals in the fourth schema item are called, prompting the user to give instantiations for the three propositions used in the relation. Assume that the user gives the following three instantiations for the three propositions :

```
inst('The FDA policy level for aflatoxins should be 20ppb',
  fda_policy_level(aflatoxins, (20,ppb))).
inst('The maximum acceptable level of aflatoxins is 20ppb',
  max_acceptable_level(aflatoxins, (20,ppb))).
inst('Aflatoxins cause cancer in humans',
  cause(aflatoxins, cancer, humans)).
```

This means that the schema variables are matched as follows :

```
P = fda_policy_level(aflatoxins, (20,ppb))
S = [max_acceptable_level(aflatoxins, (20,ppb)),
        cause(aflatoxins, cancer, humans) ]
Q = max_acceptable_level(aflatoxins, (20,ppb)) &
        cause(aflatoxins, cancer, humans)
```

3. As a side effect of this variable matching, the object-level formalisation (the third item in the schema) is now fully instantiated to the following :

```
[max_acceptable_level(aflatoxins, (20,ppb)) &
  cause(aflatoxins, cancer, humans)
→fda_policy_level(aflatoxins, (20,ppb))]
```

so the following object-level proposition is added to the object-level knowledge base :

max_acceptable_level(aflatoxins, (20,ppb)) &
cause(aflatoxins, cancer, humans)
→ fda_policy_level(aflatoxins, (20,ppb)).

This is the simplest possible case for formalising a justification. The justification maps directly to an implication, and each of the meta-level proposition names are instantiated directly. There is no redundant information at the meta-level, and no variables are used in the formalisation (all arguments to predicates are constants). The next section will examine some more complex examples.

7.2.2 Supporting a range of alternative representations

The benefit of using schemata to represent knowledge about formalisation is that many alternative formalisations can be provided to the user for any one meta-level relation. To illustrate this, some alternative methods for providing object-level representations of the meta-level relations are now given. This will allow some exploration of issues such as the existence of redundant information at the meta-level, variable handling at the object-level and introduction of quantification over variables at the object-level.

Removing redundancy and controlling inference at the object-level

In some cases, some of the meta-level proposition names may be interpreted as unnecessary in the object-level knowledge base. An example is the following justification, which in the initial formalisation shown above was formalised as an implication precisely as explained in section 2.1.

justification('Aflatoxins cause cancer in humans',
['Aflatoxins cause cancer in non-human animals',
'Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable']).

However, there is an alternative interpretation of this justification, which is that it indicates a direct implication from *'Aflatoxins cause cancer in non-human animals'* to *'Aflatoxins cause cancer in humans'*. Under this interpretation, the extra proposition *'Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable'* is stated just to explain the implication, or to provide what Toulmin [Toulmin 58] calls a 'warrant' for the implication. Interpreted like this the final proposition is redundant at the object-level. Note that in the meta-level knowledge base, the extrapolation statement is itself justified and this justification was also formalised as an implication in the mark-down at the start of section 2. In the following example we assume that the user is being selective about what they include in the object-level knowledge base, and choosing not to include this justification, leaving it there simply as a meta-level

commentary on why the extrapolation is legitimate. The redundancy which is left out of the object level knowledge base is thus both the extrapolation statement, and the justification of this statement.

When a knowledge engineer builds a knowledge base they need to decide not only what knowledge to include in it, but also decide what sort of inferences will be possible. Decisions about inference constrain how the knowledge should be formalised. To illustrate this, an alternative to formalising the justification above as a simple implication, or rule, is described here.

Consider what is required in the object-level knowledge base in order to enable *'Aflatoxins cause cancer in humans'* to be inferred from *'Aflatoxins cause cancer in non-human animals'*, using an inference rule such as *modus ponens*. The implication alone is not sufficient, as modus ponens requires both the implication, and the left-hand-side of the implication to hold, in order for the right-hand-side to be deduced. So, say the instantiations of the two propositions are as follows :

```
inst('Aflatoxins cause cancer in humans',
  cause(aflatoxins, cancer, humans)).
inst('Aflatoxins cause cancer in non-human animals',
  cause(aflatoxins, cancer, animals)).
```

The object-level knowledge base is required to contain both

cause(aflatoxins, cancer, animals) → *cause(aflatoxins, cancer, humans)*

and

cause(aflatoxins, cancer, animals)

so that modus ponens can be used to deduce

cause(aflatoxins, cancer, humans).

The schema enabling this formalisation is the following :

```
schema( 'Justification as modus ponens (general form)',
        justification(LHS, RHS),
        [Q → P| S],
        (P = inst(LHS),
              user_removes_redundancy(RHS, Rest),
              findall(inst(X), on(X, Rest), S),
              conjunction(S,Q))).
```

As before, the conclusion of the implication is the instantiation of the left hand side of the justification. But in this case, the user is asked which, if any, of the statements on the right-hand-side of the justification are redundant, and only the remaining statements are instantiated and used as the conditions of the implication. Note also that the third item of the schema, which represents the object-level formalisation, contains both the implication and also the list of instantiated condition statements, in

order to enable modus ponens to deduce the conclusion.

Continuing the example, if the user selected this schema to instantiate the justification above, the schema matching would proceed as follows :

1. First the second item of the schema would match to the justification giving these variable instantiations :

LHS = 'Aflatoxins cause cancer in humans'
RHS = ['Aflatoxins cause cancer in non-human animals',
'Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable']

2. Next, the Prolog goals in the fourth schema item are called. The left-hand-side is instantiated giving :

P = cause(aflatoxins, cancer, humans)

and then the user is asked to point out redundant statements in the right-hand-side. Assume that they select the statement about extrapolation as redundant in the object-level inference. Thus

Rest = ['Aflatoxins cause cancer in non-human animals']

S = [cause(aflatoxins, cancer, animals)]

Q = cause(aflatoxins, cancer, animals).

3. The third schema item *([Q → P| S])* is now fully instantiated and so the statements

cause(aflatoxins, cancer, animals)
→ cause(aflatoxins, cancer, humans)
cause(aflatoxins, cancer, animals)

are added to the object-level knowledge base.

This example has illustrated two things. Firstly, it has shown how information at the meta-level can be omitted during formalisation when it is considered redundant in terms of the object-level inference system. Secondly, it has shown how schemata can be used to enable users to decide during formalisation how inferences shall occur at the object-level, and thus to ensure that the object-level knowledge base contains all that it needs to enable the inference to go through. In section 7.3, a tool is described which enables this decision to be postponed until later if necessary, and to verify that *chains* of inferences are possible as a result of a set of inference *steps* which have been formalised using individual schemata.

7.2.3 Handling variables and quantification at the object-level

In this section another alternative formalisation is demonstrated to illustrate the use of

variables at the object-level. The examples so far have all produced object-level formalisations which are *ground, ie:* all the arguments of the object-level predicates are constants. However in full first order predicate logic, arguments can also be variables, and these variables can be quantified over. There are inference rules which handle reasoning with quantified variables appropriately. The following example shows how a schema can introduce a quantifier and enable inference with the resulting quantified statement. This example is illustrative of what is possible, and further schemata will need to be devised to enable the complete range of uses of quantifiers within full first order predicate logic to be handled at the object-level.

Suppose that we want to use a variable to represent the level of aflatoxins in the following justification :

justification('The maximum acceptable level of aflatoxins is 20ppb',
['There is no safe level of aflatoxins',
'The minimum detectable level of aflatoxins is 20ppb']).

By using the same variable for the level of aflatoxins both times it occurs, (instead of actually stating its level to be 20ppb as we did in the initial formalisation), the maximum acceptable level will be *deducible* at the object-level, from the knowledge of the minimum detectable level by *unification* of the variables during inference at the object-level. To do this, the following proposition needs to be added to the object-level knowledge base.

$\forall$ X. *no_safe_level(aflatoxins)*
& min_detectable_level(aflatoxins, X)
$\rightarrow$ *max_acceptable_level(aflatoxins, X).*

This states that, for any level, X, if it is the minimum detectable level of aflatoxins, and there is no safe level, then that level is the maximum acceptable level. The level, X, is universally quantified.

Assume the following instantiations of propositions (and note that the variables here have been given instantiations by the user which look like two different variable names, L and L1) :

inst('The maximum acceptable level of aflatoxins is 20ppb',
max_acceptable_level(aflatoxins, 'L')).
inst('There is no safe level of aflatoxins',
no_safe_level(aflatoxins)).
inst('The minimum detectable level of aflatoxins is 20ppb',
min_detectable_level(aflatoxins, 'L1')).

As an aside, it is worth noting that whenever the user gives an instantiation of a proposition, the mark-down system simply ignores the user's attempts to differentiate

between constants and variables, and treats every argument as a constant initially (which is why the variable L is recorded by the system as the constant 'L'). The system relies entirely on the mark-down schemata to decide which arguments of the object-level predicates need to be variables and to ensure that they will unify. Thus the user could in fact have used constants such as '20', 'level' or anything else and the system can still replace them with appropriate variables, and likewise the user's use of different variable names is also immaterial.

The schema which enables the justification of the maximum acceptable level to be formalised as an implication with a universally quantified variable, is as follows :

```
schema( 'Justification as implication, with universally quantified variable',
  justification(LHS, RHS),
  [forall(X, GenPred → GenConc)],
  (inst_prop_set([LHS |RHS], [inst(LHS, P) |RHSinsts]),
        findall(Y, on(inst(_,Y), RHSinsts), S),
        user_selects_variables(X, [P|S], [GenConc|GenS]),
        conjunction(GenS, GenPred))).
```

This schema is slightly more complex than those seen previously. As before, the user selects this schema by name. Then schema matching first matches the second schema item, and then calls the four Prolog goals in the fourth item.

The first goal finds the instantiations of the propositions used in the justification by calling *inst_prop_set/2*, which looks up, or asks the user for, instantiations of the propositions in FOPL. Then *user_selects_variables/3* is called, which is an interactive dialogue. The user is asked which arguments are to contain the quantified variables, in each of the propositions which will appear in the object-level implication. For each proposition, all of the arguments are given on a menu and the user can select any or none of them. So, for example, in the case of the conclusion of the implication :

```
P = max_acceptable_level(aflatoxins, 'L')
```

the user is presented with a menu containing two items : *aflatoxins* and *'L'*. They select *'L'* as the quantified variable (and note that the fact that this 'looks like' a Prolog variable is entirely irrelevant, and it could just as easily be the constant 'level' or '20ppb'). Both of the propositions in the condition of the implication are treated likewise. In the case of the proposition :

```
no_safe_level(aflatoxins)
```

the user states that no argument is quantified over (*ie:* that *aflatoxins* is a constant). In

the case of the minimum detectable level, the user selects *L* as the quantified variable. The system replaces each of the chosen arguments with the *same* variable name (*X*), which is the variable name used in the third schema item representing the object-level form. This third item represents the universal quantifier using the predicate *forall/2*, which takes two arguments: the variable quantified over and the object level expression which is the quantifier's scope - in this case the implication.

The final goal of the schema matching code forms the conjunction of the conditions (exactly as in section 2.1) and then matching succeeds as the third schema item (the representation of the object-level statement) is now fully instantiated. The following is added to the object-level knowledge base :

$$\forall X.\ no\_safe\_level(aflatoxins) \\ \quad \& \ min\_detectable\_level(aflatoxins, X) \\ \rightarrow max\_acceptable\_level(aflatoxins, X).$$

This section has demonstrated how the schemata can support a user in the consistent use of variables and quantifiers in their object-level knowledge base. This is achieved by embedding instantiation of propositions inside the schema matching process, and reasoning about the internal structure of the propositions, interacting with the user, in order to place variables appropriately so that they will unify. A simple example was given which shows how universal quantification can be combined with implication.

The quantification schema is more complex than the previous schemata. When a schema produces ground formulae at the object-level (as in sections 2.1 & 2.2) the instantiation of its internal propositions can be deferred until after schema-matching. However when, as in this case, the schema produces non-ground propositions at the object-level, it is necessary first to instantiate its constituent propositions because to satisfy the other schema-matching goals the internal structure of the propositions has to be manipulated. To be precise, the arguments of the predicates at the object-level need to be known so that the variables can be made to unify.

To the user there is no apparent difference in behaviour - in both cases the user first selects a schema, then gives the instantiations of the propositions at the object-level, and then the formalised statement is added to the object-level knowledge base. The difference is internal. In the ground case schema matching succeeds and then the constituent propositions are instantiated. In the non-ground case, instantiation becomes part of the schema-matching process. It could be claimed that instantiation of propositions should be carried out within the schema-matching process in both ground and non-ground cases, however it has been interesting to note that this is only necessary in the non-ground case and that these two aspects of formalisation can be uncoupled when the resulting object-level knowledge base is ground.

The same technique can be used to include quantifiers and variables in other formalisations like those shown in section 7.2.2 which omit redundant information, or enable modus ponens by including rule conditions in the object-level knowledge base. Similarly, in the case above, it is straightforward to enable the quantification to be over a variable called 'Chemical' used in place of 'aflatoxins', which would result in a rule enabling deduction of the maximum acceptable level of *any* chemical substance, if it has no safe level and its minimum detectable level is known. The schema for including two or more universally quantified variables is more complex, but follows the same lines as shown for one.

7.2.4 Discussion

As noted earlier, more work is required to produce a library of schemata which can enable formalisation of relations as any *arbitrary* formula in full first order predicate logic, for example there is no schema for quantifying over arbitrary numbers of variables, nor for combining existential and universal quantifiers, and many common inference rules are not yet implemented as schemata.

However, the schema-based mark-down tool does demonstrate, as a proof of concept, that it is possible:

a) to represent knowledge of *how to do formalisation* as general schemata, which provide mappings between meta-level relations and segments of a knowledge base encoded in FOPL;

b) to use these schemata to support a user in generating an object-level knowledge base, starting from abstract meta-level relations in the FORA framework;

c) to provide multiple, alternative, object-level formalisations from a single meta-level knowledge base;

d) to address various important issues in the formalisation of knowledge, in particular the omission of redundant information, control of object-level inference, and handling of variables and quantification.

In the next section some tools are described which can support *maintenance* and *evaluation* of knowledge bases once they have been formalised from arguments, using the schema-based mark-down. This will illustrate further the benefit of representing knowledge as abstract relations in the meta-level argumentation framework, and in particular the flexibility it allows in changing our minds about which arguments to formalise, and how we should formalise them.

7.3 Object-level inference verification.

Having used schema-based mark-down to generate some object-level knowledge base structures, it is next necessary to verify that the argument which has been used for generation is indeed fully captured at the object-level. This involves checking that the 'line of reasoning' expressed in the argument corresponds to an inference chain at the object-level. To verify this, there is a tool called an *inference checker* which first establishes the conclusion of the argument as an inference goal, and then attempts to infer it using the object-level inference mechanism, noting any points at which inference fails.

Definition : Given an argument $P <= S$, *a set of instantiations, I, and an object-level knowledge base O,*
inference_check(P) returns true iff
$inst(P, P1) \in I$, *and* $O \vdash P1$,
otherwise inference_check(P) returns P2, where
P2 is generated as a failed subgoal in the attempted proof of P1

To illustrate the inference checker, recall the object-level knowledge base given at the start of section 7.2, consisting of the following set of rules.

kind_of(aflatoxins, chemical)
& indicator(cancer_risk, chemicals, animals, humans)
→ can_extrapolate(cancer_risk, aflatoxins, animals, humans).
cause(aflatoxins, cancer, animals)
& can_extrapolate(cancer_risk, aflatoxins, animals, humans)
→ cause(aflatoxins, cancer, humans).
cause(aflatoxins, cancer, humans)
→ no_safe_level(aflatoxins, cancer, humans).
no_safe_level(aflatoxins, cancer, humans)
& min_detectable_level(aflatoxins, (20,ppb))
→ max_acceptable_level(aflatoxins, (20,ppb)).
max_acceptable level(aflatoxins, (20,ppb))
& cause(aflatoxins, cancer, humans)
→ fda_policy_level(aflatoxins, (20,ppb)).

These were formalised with the intention of enabling the deduction of

fda_policy_level(aflatoxins, (20,ppb)).

The *inference checker* program enables the user to test out this hypothesis, and records where, if anywhere, the inference fails. In this case it fails at four proof tree leaves, the assumptions for which are not stated explicitly in the knowledge base. These are the following. They need to be asserted into the knowledge base to enable the inference to go through.

cause(aflatoxins, cancer, animals).
kind_of(aflatoxins, chemical).
indicator(cancer_risk, chemicals, animals, humans).
min_detectable_level(aflatoxins, (20,ppb)).

The syntax of these propositions is already known to the inference checker program (via the records kept during mark-down, which record the object-level formalisations of each FORA proposition as it is formalised) so they can be added automatically if the user desires. Alternatively, the user can return to the mark-down program to enable further justifications to be formalised, or they can return to the meta-level to mark-up additional argument structure.

In the alternative formalisation illustrated in section 7.2.2, the justification of the extrapolation statement was omitted and the schema was used to guarantee that modus ponens will enable the inference of

cause(aflatoxins, cancer, humans).

This means that the first of the four object-level propositions is already in the knowledge base and the third is not used in the inference. The inference checker in this case only suggests the second and fourth propositions be added to the knowledge base.

In the case of section 7.2.3, assuming that the only different schema used was the one illustrated (introducing a universal quantifier), all four of these statements would be suggested by the inference checker. However, the final statement giving the minimum detectable level, would be suggested with a variable instead of the 20ppb value, and if the user gives a particular value this will enable the universally quantified variable to be instantiated during object-level inference.

The inference checker thus uses information acquired, and retained, during mark-down to enable the user to evaluate and add to their formalised knowledge base. The inference checker requires access to the object-level inference mechanism, and this mechanism needs to be able to record explicitly where inference fails. The further work of generating a larger library of mark-down schemata will require a parallel in extending the inference checker to handle a wider variety of inference mechanisms.

The inference checker demonstrates that the effort required in representing arguments at the meta-level in FORA, is repaid when these arguments are viewed as requirements for reasoning at the object-level. Using schema-based mark-down from FORA, verification that these requirements have been met can be automated.

7.4 Criticising the object-level knowledge base

The purpose of representing more than one side of the argument at the meta-level is to enable more than one point of view to be taken into account when constructing the object-level formalisation. Hence, a further tool, the *critic* program, has been implemented to highlight points in the object level knowledge base which are vulnerable to counter-argument or disagreement. It does this by reasoning at the meta-level about the statements in the formalised argument which are disputed, and then maps down to the object level to determine the object level items which could be questioned. It can use the schemata for mark down of relations, to suggest possible formalisations of statements which would alter the deductions possible at the object level.

The critic first asks the user to select a proposition at the meta-level which they want the critic to use as the basis of its attack. All of the argumentation structures which represent opposition (*ie:* all of the structures described in section 5.3.2 of this document, including counter-arguments, rebuttals etc) are then constructed, and the disagreement relation at their core is extracted. If one side of this disagreement occurs instantiated in the object-level knowledge base, then the critic uses a schema for disagreement to suggest a new object-level statement. All such statements are generated and presented to the user.

Definition : For a proposition P in a FORA knowledge base F,
and a set of object-level instantiations ,O,
∃ C. criticises(P, C) iff
∃ S such that S is an oppositional argumentation structure for P
and disagree(Q, R) ∈ S and disagree(Q, R) ∈ F,
and inst(Q, Q1) ∈ O
and schema(_, disagree(Q, R), C, _).

In this next example, the object-level knowledge base containing the five implications is criticised. From the FORA knowledge base it is possible to generate various counter-arguments to the argument in favour of the FDA policy, and these possibilities are suggested to the user, along with negated statements of the offending weak points in the argument. (This is because negation is currently FORA's only schema for formalising disagreement.) The user could at this point either adopt some of the suggestions, thus altering the inferences possible at the object-level (which could be evaluated with the *inference checker* program), or alternatively, the user may wish to take account of the weaknesses of the argument, and supply additional arguments supporting them at the meta-level.

Note that some new evidence has been added to the meta-level in FORA, namely the

proposition: *'1ppb is the minimum detectable level for aflatoxins'*. In addition, the new relation:

disagree(The minimum detectable level of aflatoxins is 20ppb,
1ppb is the minimum detectable level of aflatoxins)

has also been added, to suggest where this new evidence should be taken into account. This new evidence was introduced in chapter 3, as support for the suggestion that the FDA policy level is not, as the agricultural lobby claim, too low, but may in fact not be low enough. The critic uses this new relation as the basis of its suggestion that the minimum detectable level statement could be altered. The critic's suggestion of simply adding the negation of the current level statement is not as sophisticated as one might wish but it will at least alert the knowledge engineer to the possibility that this information undermines the conclusion currently drawable from the knowledge base. This suggestion is the third one generated in the critic's output shown below.

SOME SUGGESTIONS FOR ATTACKING :
The FDA policy level for aflatoxins should be 20ppb
IN THE OBJECT-LEVEL KNOWLEDGE BASE.

You could mark-down this relation :
disagreement(The FDA policy level for aflatoxins should be 20ppb,
The FDA policy level for aflatoxins should not be 20ppb)
For example, by adding this to the object-level knowledge base :
not fda_policy_level(aflatoxins, (20,ppb))

You could mark-down this relation :
disagreement(The maximum acceptable level of aflatoxins is 20ppb,
20ppb is not the maximum acceptable level of aflatoxins)
For example, by adding this to the object-level knowledge base :
not max_acceptable_level(aflatoxins, (20,ppb))

You could mark-down this relation :
disagreement(The minimum detectable level of aflatoxins is 20ppb,
1ppb is the minimum detectable level of aflatoxins)
For example, by adding this to the object-level knowledge base :
not min_detectable_level(aflatoxins, (20,ppb))

You could mark-down this relation :
disagreement(There is no safe level of aflatoxins,
There is a lack of scientific evidence showing no safe level of aflatoxins)
For example, by adding this to the object-level knowledge base :
not no_safe_level(aflatoxins, cancer, humans)

You could mark-down this relation :
disagreement(Aflatoxins cause cancer in humans,
There is a lack of scientific evidence showing aflatoxins cause
cancer in humans)
For example, by adding this to the object-level knowledge base :
not cause(aflatoxins, cancer, humans)

You could mark-down this relation :

disagreement(Extrapolation of the cancer risk of aflatoxins from animals to humans is reliable,
Extrapolation of cancer risk of aflatoxins from animals to humans is not reliable)
For example, by adding this to the object-level knowledge base :
not can_extrapolate(cancer_risk, aflatoxins, animals, humans)

The second last of these suggestions is interesting as it demonstrates an important aspect of Fox's initial analysis of the arguments in [Fox 94]. The counter-argument to the argument in favour of the FDA policy challenges the assumption that aflatoxins cause cancer in humans, due to a lack of corroborating scientific evidence for this claim. In the representation shown here, this challenge has been articulated as a proposition ('There is a lack of scientific evidence showing aflatoxins cause cancer in humans') which disagrees with the claim 'Aflatoxins cause cancer in humans', and so the attack program has suggested the negation of this claim as a possible formalisation.

In FORA, the lack of corroboration can also be interpreted entirely at the meta-level, by noting a lack of supporting argument for 'Aflatoxins cause cancer in humans', other than the argument which involves extrapolation from animal cancer-risk. This lack of corroboration was also detected by the *arguer* program (see section 5.4.2) which suggested ways that the user could add extra support to the claim that aflatoxins cause cancer, at the meta-level, before carrying out further formalisation.

7.5 Summary

In this chapter some benefits of using FORA have been demonstrated. In particular, some examples have been presented of software tools which make use of FORA argument representations to support a user in the formalisation task, particularly when faced with formalising conflicting points of view. These tools demonstrate that arguing at the meta-level can facilitate formalisation (or mark-down) to an object-level language, verification of inference using the resulting knowledge base and finally criticism of this knowledge base. This final example, in particular, demonstrates the gain from integrating abstract argumentation into the formalisation process, by highlighting particular object-level statements which may be vulnerable to dispute or which may need revising in the light of new evidence.

This chapter has concluded the presentation of evidence in support of my thesis that FORA is a useful framework for supporting knowledge engineering in controversial domains. The next chapter summarises this thesis and presents some avenues of future research.

Chapter 8

Conclusions and Further Work

8.1 Summary

This document has described a computational framework called FORA for articulating arguments at a high level of abstraction and exploring the resulting structures. Arguments and debate involve reasoning *about* the relationships between the statements and chains of inference we use, and so they involve *meta-level* knowledge.

Chapter 2 reviewed the literature on argumentation and disagreement, and in chapter 3, I concluded that there is a need for an argumentation system handling multiple points of view, which has the following characteristics :

1. The system should not focus on conflict resolution, but must provide facilities for exploration of the content and structure of a debate.

2. It should enable the many useful argumentation structures from the literature (such as counter-arguments, corroborations, rebuttals and undercutting) to be expressed.

3. It should be based on a formal language which discourages proliferation of primitive relations without clear meaning.

4. Most importantly, the arguments should be represented at the meta-level in a way which is independent of any particular object-level language, but which can be formally interpreted in terms of object-level logics.

FORA meets all four of these criteria and thus demonstrates that such a system is *possible*. No attempt is made in FORA to resolve inconsistencies, instead FORA includes a language for articulating disagreements and arguments at the meta-level, and allowing users to explore them. The primitives in the language can be combined to express standard argumentation structures, and they are limited in number and restricted in meaning. They are also independent of any particular object-level interpretation, and they can be interpreted as representing links between pieces of natural language text, or between elements of a formal logic.

FORA's language was described in chapter 4, together with some tools and techniques for acquiring knowledge from many points of view and abstracting

arguments and controversial statements from the resulting texts.

The *usefulness* of FORA was argued in the remaining chapters of this document. In chapter 5, its abilities to express many standard argumentation structures was demonstrated, and some tools were described for helping a user to explore these structures. In chapter 6, methods were described for constructing FORA knowledge bases by automatic abstraction (*mark-up*) from formal object-level theories or legacy knowledge bases. The usefulness of the FORA approach was also illustrated in chapter 7, by describing how instantiation (*mark-down*) of the meta-level argument structures can support construction, evaluation and criticism of object-level knowledge bases.

In chapter 5, the following functionality was identified as important for an argumentation system to provide :

1. to express arguments;

2. to explore existing arguments;

3. to relate the arguments from one source to those of another;

4. to add new arguments and to challenge existing arguments;

5. to represent the arguments of an existing knowledge source;

6. to criticise views of a knowledge source which are particularly controversial;

7. to construct or modify a knowledge source;

8. to check that a knowledge source actually reasons according to an argument;

9. to identify how new evidence might require changes to a knowledge source.

FORA provides, in prototype form, all of this functionality. Points 1-4, which are all meta-level functionalities, were described in Chapter 5. Points 5 and 6 involve handling existing object-level knowledge sources and these were the subject of Chapter 6. Points 7-9 involve constructing and modifying object-level knowledge sources and they were discussed in Chapter 7.

Throughout the document all tools and techniques have been illustrated in terms of a running example which I refer to as 'the aflatoxin debate', based on a real world controversy about an important health risk assessment. This has provided a

preliminary evaluation of FORA's effectiveness in handling real-world argumentation, and has enabled me to demonstrate some of the advantages of a meta-level approach to argument representation.

8.2 Further Work

The future directions of this research include the following possibilities :

1. *Application of FORA to other argumentation examples.* This would demonstrate further the ability of the FORA framework to handle complex arguments, and in particular to address the needs of users of knowledge from multiple viewpoints. Because of the generality of the system it is interesting to speculate on its application to other uses of argumentation. Two examples of situations where argumentation is used are decision making and education.

It is possible to imagine a decision-maker weighing up the arguments for and against options, using a computer system based on FORA representations of these arguments. The FORA system at the moment does not include functionality specifically geared to such decision makers (this would include much more sophisticated user-interface functionality such as graphical representation of argument patterns, and possibly also support for conflict resolution techniques, if required). A current research project coordinated by the author is looking at the requirements of environmental managers for functionality of this sort, but it is important to emphasise that though interesting, such functionality is not within the scope of this document.

Argumentation is also useful in education (see for example [Smith *et al* 81], [Suthers *et al* 95]), both in understanding topics which are controversial where a student needs to understand how competing views relate to each other, and also in developing students' abilities to produce and develop arguments and enter into debates. Section 5.5 described an application of the FORA language to a tutoring situation in which the computer plays the role of a devil's advocate to a student who is required to understand issues in controversial domains. Development of other tools to support the development of students' argumentation skills is another area of current and future research.

2. *Use of meta-meta-level relationships.* The FORA relations are all between proposition names, but there is a clear interest in stating relationships between relations too, *ie:* reasoning about the meta-level. For example, we may disagree that a reasoning step is really a justification of its conclusion, or that two propositions are equivalent, or we may wish to claim that two justification steps are equivalent. In addition, some of the proof steps parsed in chapter 6 demonstrated a need for justifications in which one of the supports is itself a justification. The formal and

practical characteristics of meta-meta level reasoning such as this would be an interesting topic of further study. It would be particularly interesting to explore using FORA to argue about its own arguments, thus providing the extra meta level by applying FORA reflectively to itself.

3. *Representation of debating moves or strategies.* In Chapter 5, the FORA language was used to represent many standard argumentation structures from the literature. This gives a declarative representation of the structure of a FORA knowledge base. Two lines of research present themselves here. Firstly, there is a need to examine the extent to which the set of argumentation structures can be considered 'complete'. I have argued that there are an arbitrarily large number of ways of combining the basic FORA relations, but it may be that there is an effective limit to the set of such structures which are meaningful or useful. Secondly, it would also be interesting to look at a procedural approach to these argumentation structures, in which patterns of FORA relations are used to define strategies for exploring knowledge bases or for simulating debate. Patterns of repeated argumentation moves may be a way of characterising the ways in which certain types of people act in a debating situation (*eg:* always disagreeing with the most recent proposition could be a characterisation of 'squabbling'). The devil's advocate strategy is an example of this kind of extension to FORA. A closely related idea is the notion of implementing such strategies as argumentation *agents.* The field of Distributed Artificial Intelligence (DAI) is a rich source of ideas for novel applications of argumentation techniques, and although in-depth coverage of DAI has been beyond the scope of this document, it is clear that conflict between agents or distributed AI systems is viewed by some researchers as an interesting opportunity for reasoning and debate (see, for example, [Galliers 90]).

4. *Generalising mark-up.* Automation of mark-up, described in Chapter 6, has been shown to be possible for propositional and first order predicate logics using a sequent calculus theorem prover. The question which remains is - how general purpose is this technique? What is involved in automating mark-up to FORA from a theorem prover using a temporal logic, a paraconsistent logic, or a frame-based expert system? In addition, it would be interesting to provide methods for parsing proof steps to generate the other FORA relations, *ie:* elaborations and equivalences and in particular, to use elaborations to capture the reasoning carried out in object-level variable handling.

5. *Improving mark-down.* The semi-automated mark-down described in Chapter 7 is limited by a small set of available schemata for mapping between FORA and the object-level, so an obvious extension is to extend the schema library. It would also be feasible to introduce a more sophisticated dialogue mechanism to help the user to specify their object-level knowledge base. This process is similar to the 'idealisation' process involved in building simulation models of real world systems, so dialogue

systems which support this process (see, for example, [Haggith 90b]) may be of relevance here.

8.3 Conclusions

Aristotle devised logic more than two thousand years ago, as a way of articulating and clarifying the debates held in the public arenas of Greek society. Logic was intended as a useful tool to support the practical business of arguing, clarifying difficult decisions and untangling conflicts. The existence of multiple inconsistent viewpoints was assumed to be the starting point from which logical reasoning ensued. The philosopher Stephen Toulmin has argued that during the past century, logic (and the computer systems based on it) has been 'mathematised' and has become distanced from its application to practical reasoning about conflicts. Disagreement, inconsistency and contradiction are now generally treated as problems to be avoided, removed or resolved.

I have argued the thesis that disagreements between multiple viewpoints are, as Toulmin claims, opportunities for exploration and reasoning, and that it is both possible and useful to provide computer-based support for handling disagreements without resolving them. I have further argued that the most appropriate basis for such support is by helping people to articulate *arguments* for points of view which disagree, and I have described a framework called FORA for doing this.

Informal argument representations are problematic because it is not possible easily to automate reasoning with them, so a formal representation language has been defined in FORA. Most formal argumentation systems are object level extensions of one particular logic, but I have argued that this is limiting, and instead advocated taking a more abstract approach to argument representation. FORA representations treat arguments as meta-level structures which describe the relationships between claims expressed as propositions in any object-level language (including natural language text). The usefulness of this framework has been demonstrated by showing how FORA can support several knowledge engineering tasks, such as knowledge base construction, maintenance and criticism.

Bibliography

Alvarado, S.G. *Understanding editorial text: a computer model of argument comprehension.* PhD Thesis, Technical report CSD-890045, Computer Science Department, University of California, Los Angeles. July 1989.

Aristotle. *The complete works of Aristotle : the revised Oxford translation.* Barnes, J. (ed). Princeton University Press. 1984

Baker, G.P. and Hacker, P.M.S. *Language, Sense and Nonsense.* Basil Blackwell, Oxford. 1984.

Beck, H. and Watson, D. Analysis of Agricultural Extension Documents. *AI Applications.* Vol 6, No. 3. 1992.

Bench-Capon, T.J.M., Lowes, D. and McEnery, A.M. Argument-based explanation of logic programs. *Knowledge Based Systems* . Vol 4, No 3. September 1991.

Boose, J. Rapid acquisition and combination of knowledge from multiple experts in the same domain. *Future Computing Systems.* Vol 1. 1986.

Brachman, R.J. and Levesque, H.J. *Readings in Knowledge Representation.* Morgan Kaufmann, 1985.

Brewka, G. *Nonmonotonic Reasoning : Logical Foundations of Commonsense.* Cambridge University Press. 1991.

Buchanan, B. and Shortliffe, E.H. *Rule-based Expert Systems : The MYCIN experiments of the Stanford Heuristic Programming Project.* Addison-Wesley. 1984.

Carnielli, W.A. and Marques, M.L. Reasoning under Inconsistent Knowledge. Internal report 90-15/R, Institut de Researche en Informatique de Toulouse (IRIT), France. 1990.

Casson, A. and Stone, D. An Expertext System for Building Standards. *Proceedings of workshop on Computers and Building Standards.* Montreal. 1992.

Cohen, P.R. *Heuristic Reasoning about Uncertainty : An Artificial Intelligence Approach.* Pitman Publishing Ltd / Morgan Kaufmann. 1985.

Conklin, J. and Begeman, M.L. gIBIS : A Hypertext Tool for Exploratory Policy Discussion. *ACM Transactions on Office Information Systems*. Vol 6, No 4. October 1988.

da Costa, N.C.A. On the Theory of Inconsistent Formal Systems. *Notre Dame Journal of Formal Logic.* Vol 15, No 4, pp 497-510. 1974.

da Costa, N.C.A. Automatic Theorem Proving in Paraconsistent Logics : Theory and Implementation. *Proceedings of CADE-10.* M.E.Stickel (ed). Springer-Verlag. 1990.

Daradoumis, T. Managing Interruptions in Tutorial Dialogues by means of an extended RST-based model. *Proceedings of the Conference on AI & Education, AI&ED-93.* Edinburgh. 1993.

Daradoumis, T. Building a Dynamic RST-based Grammar of Coherent Interactive Dialogues. *Proceedings of the Fourth European Workshop on Natural Language Generation.* 1993.

de Kleer, J. An Assumption Based TMS. *Artificial Intelligence.* Vol 28. 1986.

Delbecq, A.L., Van de Ven, A.H. and Gustafson, D.H. *Group Techniques for Program Planning : a guide to nominal group and Delphi processes.* Management Application Series, xv. Scott Foresman. 1975.

Deville, Y. *Logic Programming.* Addison-Wesley Publishing Company. 1990.

Doyle, J. A truth maintenance system. *Artificial Intelligence.* Vol 12. 1979.

Doyle, J. Reason maintenance and belief revision : Foundations versus Coherence theories. In [Gärdenfors 92].

Dryzek, J. *Conflict and Choice in Resource Management : The Case of Alaska.* Westview Press. 1983.

Dubois, D. and Prade, H. Reasoning with inconsistent information in a possibilistic setting. IRIT Research Report (IRIT/90-2/R), Université Paul Sabatier, Toulouse, France. 1990.

Dummett, M. *Elements of Intuitionism.* Oxford University Press. 1977.

Easterbrook, S. Handling conflict between domain descriptions with computer-supported negotiation. *Journal of Knowledge Acquisition.* Vol 3, No 3. P225-290. 1991.

Eaton, D.L. and Groopman, J.D. (eds) *The toxicology of aflatoxins : human health, vetinary, and agricultural significance.* Academic Press. 1994.

Elvang-Gøransson, M. & Hunter, A. Acceptable inferences from inconsistent information. Working Paper WPCS-93-13, Centre for Cognitive Informatics, Roskilde University, Roskilde, Denmark. 1993.

Elvang-Gøransson, M., Krause, P. and Fox, J. Dialectic reasoning with inconsistent information. *Proceedings of Uncertainty in Artificial Intelligence.* 1993.

Ericsson, K.A. and Simon, H.A. *Protocol Analysis : verbal reports as data.* Bradford Books, MIT Press. 1984.

Etherington, D. *Reasoning with Incomplete Information.* Pitman Publishing, London. 1988.

Evans, J.D.G. *Aristotle's concept of dialectic.* Cambridge University Press. 1977.

Fawcett, R. and Davies, B. Monologue as a Turn in Dialogue: Towards an Integration of Exchange Structure and RST. *Aspects of Automated Natural Language Generation.* Dale, R., Hovy, E., Rösner, D. and Stock, O. (eds). Springer Verlag. 1992.

Finkelstein, A., Gabbay, D., Hunter, A., Kramer, J., and Nuseibeh, B. Inconsistency Handling in Multi-Perspective Specifications. *Proceedings of the Fourth European Software Engineering Conference (ESEC 93).* Springer-Verlag. 1993.

Fisher, A. *The Logic of Real Arguments.* Cambridge University Press. 1988.

Fox, J. An example of the use of formal argument in assessing cancer risk: the cases for and against FDA policy on aflatoxins. Technical report (unpublished), Imperial Cancer Research Fund, London. 1994.

Fox, J. and Clarke, M. Towards a formalisation of arguments in decision making. *Proceedings of AAAI Spring Symposium on Argumentation and Belief.* Stanford University. March 1991.

Fox, J., Krause, P. and Ambler, S. Arguments, contradictions and practical reasoning. *Proceedings of the European Conference on AI, ECAI-92.* 1992.

Fox, J. and Krause, P. Qualitative frameworks for decision support : lessons from medicine. *The Knowledge Engineering Review.* Vol 7:1. 1992.

Fox, J. Das, S. and Elsdon, D. Decision making and planning by autonomous systems : theory, technology and applications. *Proceedings of the Workshop on Decision Theory for Distributed AI Applications, ECAI-94.* Amsterdam. 1994.

Freeman, K. and Farley, A. Argumentation in weak theory domains. *Proceedings of the Fifth Irish Conference on Artificial Intelligence and Cognitive Science.* K. Ryan & R. Sutcliffe (eds). Springer Verlag. 1992. Also available as Research report CIS-TR-92-10, University of Oregon, Eugene. April 1992.

Freeman, K. and Farley, A. Toward Formalizing Dialectical Argumentation. Research report CIS-TR-93-13, University of Oregon, Eugene. May 1993.

Frege, G. *The Foundations of Arithmetic.* Basil Blackwell, Oxford. 1950.

Gabbay, D. Labelled Deductive Systems : part 1. Research Report, CIS-Bericht-90-22, University of Munich. December 1990.

Gabbay, D. and Hunter, A. Making Inconsistency Respectable: A Logical Framework for Inconsistency in Reasoning: Part 1 - A Position Paper. *Fundamentals of Artificial Intelligence Research.* P. Jorrand and J. Kelemen (eds). Lecture Notes in Computer Science 535, Springer Verlag. 1991.

Gabbay, D. and Hunter, A. Making Inconsistency Respectable: Part 2 - Meta-level handling of inconsistency. *Proceedings of ECSQUARU conference 1993.* Lecture Notes in Computer Science 747, Springer Verlag. 1993.

Gabbay, D. and Hunter, A. Restricted Access Logics for Inconsistent Information. *Proceedings of ECSQUARU conference 1993.* Lecture Notes in Computer Science 747, Springer Verlag. 1993.

Gaines, B.R. and Boose, J.H. (eds). *Knowledge acquisition for knowledge based systems.* Academic Press. 1988.

Galliers, J.R. The positive role of conflict in cooperative multi-agent systems. *Decentralised AI.* Demazeau, Y. and Müller, J-P. (eds). Elsevier Science Publishers. 1990.

Galliers, J.R. Autonomous belief revision and communication. In [Gärdenfors 92].

Gärdenfors, P. *Knowledge in Flux.* MIT Press. 1988.

Gärdenfors, P. *Belief Revision.* Cambridge University Press. 1992.

Genesereth, M.R. and Nilsson, N.J. *Logical Foundations of Artificial Intelligence.* Morgan Kaufmann. 1987.

Ginsberg, M. (ed). *Readings in nonmonotonic reasoning.* Morgan Kaufmann. 1987.

Ginsberg, M. *Essentials of Artificial Intelligence.* Morgan Kaufmann. 1993.

Haack, S. *Philosophy of Logics*, Cambridge University Press. 1978.

Haggith, M. HIPPIE : Using Program Schemata to Generate Dialogue for Ecological Modelling. MSc Thesis, Department of AI, University of Edinburgh. 1988.

Haggith, M. Interactive Program Construction. *Proceedings of UK IT 90.* Southampton, UK. March 1990.

Haggith, M. Sheltering Under a Tree : An Artificial Intelligence Perspective on Idealisation in Ecological Modelling. *Proceedings of Pacific Rim International Conference on Artificial Intelligence (PRICAI-90).* Nagoya, Japan. November 1990.

Haggith, M. A meta-level framework for exploration of multiple knowledge bases. *Proceedings of the Conference on Cooperating Knowledge Based Systems (CKBS-94).* University of Keele. June 1994.

Haggith, M. A meta-level framework for exploring conflicting knowledge bases. *Hybrid problems, hybrid solutions.* J. Hallam (ed). IOS Press. 1995.

Haggith, M. Argumentation in Natural Resource Management. *Proceedings of the workshop on AI and the Environment, IJCAI-95.* Montreal. August 1995.

Hamilton, A.G. *Logic for Mathematicians.* Cambridge University Press. 1978.

Hansson, S.O. A dyadic representation of belief. In [Gärdenfors 92].

Harman, G. *Change in View : Principles of Reasoning.* MIT Press. 1986.

Hart, A. *Knowledge acquisition for expert systems.* Kogan Page, London. 1986.

Heyting, A. The Intuitionistic Foundations of Mathematics. *Erkenntnis.* Vol 2. 1931. Also in *Philosophy of Mathematics : Selected Readings.* Benacerraf P. and Putnam, H. (eds). 2nd Edition, Cambridge University Press. 1983.

Hodges, W. *Logic.* Penguin Books. 1977.

Huang, J., Jennings, N.R. and Fox, J. Cooperation in Distributed Medical Care. *Proceedings of the Second International Conference on Cooperative Information Systems (CoopIS-94).* Toronto, Canada, May 1994.

International Crops Research Institute for the Semi-Arid Tropics. Summary and Recommendations of the International Workshop on Aflatoxin Contamination of Groundnut. ICRISAT, Andrha Pradesh, India. 1987.

Jackson, P. *Introduction to Expert Systems.* 2nd Edition, Addison-Wesley. 1990.

Kant, I. *Critique of pure reason* . Translated by Kemp Smith, N. 2nd Edition,

Macmillan, London. 1933.

Kellogg, W. *Response to Sceptics of Global Warming.* American Meteorological Society Bulletin. Vol 74, No 4. 1991.

Kidd, A.L. (ed). *Knowledge acquisition for expert systems: a practical handbook.* Plenum, New York. 1987.

Kleene, S.C. *Introduction to metamathematics.* Van Nostrand. 1952.

Kleene, S.C. *Mathematical Logic.* John Wiley & Sons. 1967.

Knott, A. and Dale, R. Using Linguistic Phenomena to motivate a set of Rhetorical Relations. Research report HCRC/RP-39. Human Communication Research Centre, University of Edinburgh. 1992

Kowalski, R. A. and Sergot, M. J. The use of logical models in legal problem solving. *Ratio Juris.* Vol 3, No 2. 1990.

Krause, P., Fox, J. and Judson, P. An argumentation-based approach to risk assessment. *IMA Journal of Mathematics Applied in Business and Industry (1993/4).* Vol 5, pp249-263. Oxford University Press. 1994.

Krause, P., Ambler, S., Elvang-Gøransson, M. and Fox, J. A logic of argumentation for reasoning under uncertainty. *Computational Intelligence.* Vol 11, No 1. 1995.

Krause, P., Fox, J. and Judson, P. Is there are role for qualitative risk assessment? *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence, UAI-95.* Montreal, Canada. 1995.

Kuhn, T.S. *The Structure of Scientific Revolutions.* Chicago University Press. 1962.

Kunz, W. and Rittel, H.W.J. Issues as Elements of Information Systems. Working paper 131, Institute fur Grundlagen der Planung. Universitat Stuttgart. July 1970.

Lemmon, E.J. *Beginning Logic.* Van Nostrand Reinhold Ltd. 1965.

Lindley, D.V. *Making Decisions.* (2nd ed) Wiley, 1985.

Lloyd, J. *Foundations of Logic Programming.* Springer Verlag. 1984.

Loui, R.P. Process and Prolicy : Resource-Bounded Non-Demonstrative Reasoning. Technical Report, WUCS-92-43, Department of Computer Science, Washington University. October 1992.

Loui, R.P., Norman, J., Olson, J. and Merrill, A. A design for Reasoning with Policies, Precedents and Rationales. *Proceedings of the Fourth International Conference on Artificial Intelligence and Law.* Amsterdam, The Netherlands. June 1993.

Lukasiewicz, J. On 3-valued logic. *Polish Logic.* McCall, S. (ed). Oxford Unversity Press. 1967.

MAFF, UK Ministry of Agriculture Fisheries and Food, Agricultural Development and Advisory Service. Aflatoxin in animal feed. MAFF, leaflet 817. 1982.

Mann, W.C. and Thompson, S.A. Rhetorical Structure Theory : A framework for

the analysis of texts. Research report, ISI/RS-87-185, University of Southern California. April 1987.

Mann, W.C. and Thompson, S.A. Antithesis : A Study in Clause Combining and Discourse Structure. Research report, ISI/RS-87-171, University of Southern California. April 1987.

Mann, W.C., Matthiessen, C.M.I.M. and Thompson, S.A. Rhetorical Structure Theory and text analysis. Research report, ISI/RR-89-242, University of Southern California. November 1989.

Martin, R.L. (ed). *Recent essays on truth and the liar paradox.* Clarendon Press, Oxford. 1984

Martin-Löf, P. Constructive Mathematics and Computer Programming. *Logic, Methodology and Philosophy of Science, proceedings of the 6th International Congress, Hannover 1979.* North-Holland Publishing Company, Amsterdam. 1982.

McAllester, D. A. An outlook on truth maintenance. AI memo (MIT) No 551. MIT, Artificial Intelligence Laboratory, Cambridge, Mass. 1980

McCarthy, J. Circumscription - A Form on Non-Monotonic Reasoning. *Artificial Intelligence* . Vol 13. 1980.

McCarthy, J. and Hayes, P.J. Some Philosophical Problems from the standpoint of Artificial Intelligence. *Machine Intelligence 4.* B. Meltzer and D. Michie (eds). Edinburgh University Press. 1969.

McDonald, D. Aflatoxins : poisonous substances that can be present in Nigerian groundnuts. Institute of Agricultural Research, Zaria. 1976.

Meiland, J.W. and Krausz, M. (eds). *Relativism, cognitive and moral.* University of Notre Dame Press. 1982.

Mohi Eldin, H. Detection of aflatoxins in food and feeds with particular reference to molasses and milk. MSc thesis. University of Edinburgh. 1988.

Moore, C.J. and Miles, J.C. Knowledge elicitation using more than one expert to cover the same domain. *Artificial Intelligence Review.* Vol 5, p255 271. 1991.

Naqvi, S.A. and Rossi, F. Reasoning in Inconsistent Databases. *Logic Programming: Proceedings of the North American Conference.* S. Debray and M. Hermenegildo (eds). MIT Press. 1990.

Nebel, B. Syntax-based approaches to belief revision. In [Gärdenfors 92].

Ng, K.W. Knowledge Acquisition from Multiple Experts using the Delphi Method. MSc Thesis, Department. of AI, University of Edinburgh. 1990.

Politakis, P.G and Weiss, S.M. Designing consistent knowledge bases : an approach to expert knowledge acquisition. Research report CBM-TR-113. Department of Computer Science, Rutgers University, New Brunswick, New Jersey. 1980.

Priest, G., Routley, R. and Norman, J. (eds). *Paraconsistent Logic : Essays on the Inconsistent.* Philosophia Verlag. 1989.

Pylyshyn, Z. Complexity and the study of Artificial and Human Intelligence.

Philosophical Perspectives on Artificial Intelligence. Martin Ringle, (ed). Harvester Press. 1978.

Quine, W.V. *Philosophy of Logic.* Prentice-Hall. 1970.

Quine, W.V. and Ullian, J.S. *The Web of Belief.* 2nd edition, Random House, NewYork. 1978.

Raphael, B. The Frame Problem in Problem Solving Systems. *Artificial Intelligence and Heuristic Programming.* Findler, N.V. and Meltzer, B.(eds). Edinburgh University Press 1971.

The RED (Rigorously Engineered Decisions) Development Group (headed by John Fox, ICRF) Expert systems for Safety Critical Applications : theory, technology and applications. *Proceedings of IEE Colloquium on Designing Safety Critical Systems.* London. April 1994.

Reiter, R. A logic for default reasoning. *Artificial Intelligence.* Vol 13. 1980.

Reiter, R. On Reasoning by Default. In [Brachman & Levesque 85].

Rescher, N. and Brandom, R. *The Logic of Inconsistency.* American Philosophical Quarterly Library of Philosophy, Vol 5. Basil Blackwell, Oxford. 1979.

Retalis, S. OLIA : An Intelligent Tutoring System for Controversial Domains. MSc Thesis. Dept. of AI, University of Edinburgh. 1995.

Retalis, S., Pain, H. and Haggith, M. Arguing with the Devil : Teaching in Controversial Domains. *Proceedings of ITS-96: Intelligent Tutoring Systems.* Montreal. 1996.

Robertson, D., Bundy, A., Muetzelfeldt, R., Haggith, M. and Uschold, M. *Eco-Logic: Logic based approaches to ecological modelling.* MIT Press. 1991.

Robertson, D. and Goldsborough, D. Representing the structure of Reserve Selection Arguments Using Logic Programs. *Proceedings of the Eighth Annual Symposium on Geographical Information systems.* Sawayama, G. (ed). 1994.

Robinson, J.A. A Machine-Oriented Logic Based on the Resolution Principle. *Journal of the Association for Computing Machinery.* Vol 12, No 1. 1965.

Rodricks, J.V. *Calculated Risks : the toxicity and human health risks of chemicals in our environment.* Cambridge University Press. 1992.

Rosen, M. *Hegel's dialectic and its criticism.* Cambridge University Press. 1982.

Russell, B. *A History of Western Philosophy* . Allen and Unwin. 1946.

Russell, B. The Philosophy of Logical Atomism. *Bertrand Russell : Logic and Knowledge, essays 1901-50.* R.C.Marsh, (ed). Allen and Unwin. 1956.

Sartor, G. A Simple Computational Model for Nonmonotonic and Adversarial Legal Reasoning. *Proceedings of the Fourth International Conference on Artificial Intelligence and Law.* Amsterdam, The Netherlands. June 1993.

Shaw, M.L.G. and Gaines, B.R. A methodology for analyzing terminological and conceptual differences in language use across communities. *Proceedings of the First*

Quantitative Linguistics Conference, QUALICO. Trier, Germany. September 1991.

Scott, D. *Notes on the Formalisation of Logic, parts III & IV.* Study Aids Monograph No 3. Sub-faculty of Philosophy, University of Oxford. July 1981.

Sillince, J.A.A. Multi-agent conflict resolution: a computational framework for an intelligent argumentation program. *Knowledge Based Systems.* Vol 7, No 2, p75-90. 1994.

Smith, K., Johnson, D.W. and Johnson, R.T. Can conflict be constructive? Controversy versus concurence seeking in learning groups. *Journal of Educational Psychology.* Vol 73, No 5, p651-663. 1981.

Suthers, D., Weiner, A., Connelly, J. and Paolucci, M. Belvedere: Engaging Students in Critical Discussion of Science and Public Policy Issues. *Proceedings of 7th World Conference on AI in Education (AI-ED 95).* Washington DC. August 1995.

Toulmin, S. *The Uses of Argument.* Cambridge University Press, Cambridge. 1958.

Trice, A. and Davis, R. Consensus knowledge acquisition. AI Memo (MIT) No 1183. Artificial Intelligence Laboratory, MIT, Cambridge, Mass. 1989.

Turner, R. *Logics for Artificial Intelligence.* Ellis Horwood Ltd. 1984.

UK Government. Aflatoxins in nuts, nut products, dried figs and dried fig products: Regulations, 1992. HMSO, London. 1992.

Whitby, B. *Artificial Intelligence : a handbook of professionalism.* Ellis Horwood. 1988.

Wielinga, B. (ed). *Current trends in knowledge acquisition.* IOS Press, Amsterdam. 1990.

Wittgenstein, L. *Tractatus Logico-Philosophicus.* Routledge & Kegan Paul Ltd. 1922.

Wittgenstein, L. *Philosophical Investigations.* 2nd Edition, Basil Blackwell, Oxford. 1958.

Wittgenstein, L. *Remarks on the Foundations of Mathematics.* 2nd Edition, Basil Blackwell, Oxford. 1967.

Yang, S-A., Robertson, D. and Lee, J. KICS: A Knowledge-Intensive Case-Based Reasoning System for Building Regulations and Case Histories. *Proceedings of 4th International Conference on AI and Law.* 1993.

Appendix A

Proof parsing rules

This appendix contains the Prolog representation of the proof parsing rules described in section 6.3.1.

```
/* Rule 1 : If the hypotheses have changed, but the theory has not, then any old hypotheses which
have been removed must be derivable from the new ones, according to the inference rule applied. */

rule_to_rels(Rule, Justifications) :-
        hypothesis_change(Rule,OldHyps, NewHyps),
        no_theory_change(Rule),
        rule_name(Rule, Name),
        setof(justification(Hyp, NewHyps, Name),
            member(Hyp, OldHyps),
            Justifications).

/* Rule 2 : If the hypotheses have changed, and items have been removed from the theory, then the
hypotheses which have been removed, must have been derivable from the new hypotheses plus those
items in the theory which have been removed. */

rule_to_rels(Rule, Justifications) :-
        hypothesis_change(Rule, Hyps, New),
        theory_change(Rule, [], Removals), % additions should be empty
        union(New, Removals, Supports),
        rule_name(Rule, Name),
        setof(justification(Hyp, Supports, Name),
            member(Hyp, Hyps),
            Justifications).

/* Rule 3 : If no hypotheses have been eliminated, but new hypotheses have been generated, and the
theory has changed, then any additions to the theory must be justified by the eliminated elements in
the theory plus the new hypotheses. */

rule_to_rels(Rule, Justifications) :-
        hypothesis_change(Rule, [], New),   % no hypothesis eliminated
        theory_change(Rule, Additions, Removals),
        union(New, Removals, Supports),
        rule_name(Rule, Name),
        setof(justification(Add, Supports, Name),
            member(Add, Additions),
            Justifications).

/* Rule 4 : If the hypotheses change and there's an addition, but no removal, from the theory, then the
old hypotheses must have been justified by a further justification - of the new hypothesis by the
additions to the theory. ie: this is a meta-meta-level relation */

rule_to_rels(Rule, Justifications) :-
        hypothesis_change(Rule, Old, [New]), % just one new hypothesis
        theory_change(Rule, Additions, []),
        rule_name(Rule, Name),
        setof(justification(H, [justification(New, Additions)], Name),
                member(H, Old),      Justifications).
```

```
/* Rule 5 : If the hypotheses have not changed, but the theory has changed and has not branched, any
new items in the theory must be derivable from the items which have been removed. */

rule_to_rels(Rule, Justifications) :-
        \+ hypothesis_change(Rule, _, _),
        theory_change(Rule, Additions, Removals),
        \+ branching_proof(Rule),
        rule_name(Rule, Name),
        setof(justification(Add, Removals, Name),
            member(Add, Additions),
             Justifications).

/* Rule 6 : If the hypotheses have not changed in the proof step, and the proof branches, with the
same items removed from the theory in both branches but different additions in each branch,
then the hypotheses each have a justification consisting of the thing removed  from the theory,
together with two meta-level justifications representing each branch of the proof. ie: this rule says that
the hypothesis is justified by generation of two sub proofs, or alternatively, that this is the point in
the proof where assumptions are discharged. */

rule_to_rels(Rule, Justifications) :-
        \+ hypothesis_change(Rule, _, _),
        branching_proof(Rule),
        hypotheses(Rule, Hyps),
        theory_change_left_branch(Rule, Adds1, Removals),
        theory_change_right_branch(Rule, Adds2, Removals),
                                        % same set as L-branch
        \+ (Hyps = Adds1 ; Hyps = Adds2), !,
        rule_name(Rule, Name),
        setof(justification(Hyp,
                        [justification(Hyp, Adds1), justification(Hyp, Adds2)|Removals],
                        Name),
                member(Hyp, Hyps),
                Justifications).

/* Rule 7 : This rule is similar to rule 6, except that in the left-hand-branch of the proof tree, no new
derived propositions are generated and added to the theory, and therefore the only interesting
justification occurs in the right-hand-branch of the proof tree. In other words, the hypothesis is
justified by generation of a sub-proof in the right-hand branch of the proof. */

rule_to_rels(Rule, Justifications) :-
        \+ hypothesis_change(Rule, _, _),
        hypotheses(Rule, Hyps),
        theory_change_left_branch(Rule, Hyps, Removals),
                                % L-branch has no interesting additions
        theory_change_right_branch(Rule, Adds, Removals),
                                % note same removal set as L-branch
        rule_name(Rule, Name),
        setof(justification(Hyp,
                        [justification(Hyp, Adds)|Removals], Name),
                member(Hyp, Hyps),
                Justifications).
```

```
/* Rule 8 : This rule is the counter part to rule 7, in which the hypothesis is  justified by generation
of a sub-proof in the left-hand-branch of the proof tree. */

rule_to_rels(Rule, Justifications) :-
        \+ hypothesis_change(Rule, _, _),
        hypotheses(Rule, Hyps),
        theory_change_left_branch(Rule, Adds, Removals),
                                % R-branch has no interesting additions
        theory_change_right_branch(Rule, Hyps, Removals),
                                % note same removal set as L-branch
        rule_name(Rule, Name),
        setof(justification(Hyp,
                [justification(Hyp, Adds)|Removals], Name),
                                        member(Hyp, Hyps),
                                        Justifications).
```

Appendix B

Proof of the FDA policy conjecture

This appendix contains the proof described in section 6.4.2, in the syntax as produced by the Prolog implementation of the theorem prover. (It has been slightly reformatted to increase readability).

```
level : 0
rule : rule(all_elim,
Above line :
[all(species1, all(species2, all(class, exists(path, current_assessment(aflatoxins, path, species2)) &
        kind_of(aflatoxins, class)&indicates_risk(class, species1, species2)
        ->can_extrapolate(aflatoxins, species1, species2)))),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) &
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path, species2))&
        kind_of(agent, class)&indicates_risk(class, species1, species2)
        ->can_extrapolate(agent, species1, species2))))),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]
```

```
level : 1
rule : rule(all_elim,
Above line :
[all(species2, all(class, exists(path, current_assessment(aflatoxins, path, species2))&
        kind_of(aflatoxins, class)&indicates_risk(class, animals, species2)
        ->can_extrapolate(aflatoxins, animals, species2))),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(species1, all(species2, all(class, exists(path, current_assessment(aflatoxins, path, species2))&
        kind_of(aflatoxins, class)&indicates_risk(class, species1, species2)
        ->can_extrapolate(aflatoxins, species1, species2)))),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 2
rule : rule(all_elim,
Above line :
[all(class, exists(path, current_assessment(aflatoxins, path, humans))&
        kind_of(aflatoxins, class)&indicates_risk(class, animals, humans)
        ->can_extrapolate(aflatoxins, animals, humans)),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
```

```
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(species2, all(class, exists(path, current_assessment(aflatoxins, path, species2))&
        kind_of(aflatoxins, class)&indicates_risk(class, animals, species2)
        ->can_extrapolate(aflatoxins, animals, species2))),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]


level : 3
rule : rule(all_elim,
Above line :
[exists(path, current_assessment(aflatoxins, path, humans))&
        kind_of(aflatoxins, chemicals)&indicates_risk(chemicals, animals, humans)
        ->can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
```

```
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(class, exists(path, current_assessment(aflatoxins, path, humans))&
        kind_of(aflatoxins, class)&indicates_risk(class, animals, humans)
        ->can_extrapolate(aflatoxins, animals, humans)),
current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 4
rule : rule(imp_elim,
Above line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [exists(path, current_assessment(aflatoxins, path, humans))&
        kind_of(aflatoxins, chemicals)&indicates_risk(chemicals, animals, humans)]

[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
```

```
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level)))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[exists(path, current_assessment(aflatoxins, path, humans))&
        kind_of(aflatoxins, chemicals)&indicates_risk(chemicals, animals, humans)
        ->can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level)))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 5
rule : rule(and_intro,
Above line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level)))))]
|- [exists(path, current_assessment(aflatoxins, path, humans))]

[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level)))))]
```

|- [kind_of(aflatoxins, chemicals)&indicates_risk(chemicals, animals, humans)]

Below line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [exists(path, current_assessment(aflatoxins, path, humans))&kind_of(aflatoxins, chemicals)
&indicates_risk(chemicals, animals, humans)]

level : 6
rule : rule(exist_intro,
Above line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [current_assessment(aflatoxins, cancer, humans)]

Below line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [exists(path, current_assessment(aflatoxins, path, humans))]

level : 7

rule : rule(direct,
Above line :

Below line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [current_assessment(aflatoxins, cancer, humans)]

level : 6
rule : rule(and_intro,
Above line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [kind_of(aflatoxins, chemicals)]

[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [indicates_risk(chemicals, animals, humans)]

Below line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),

```
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [kind_of(aflatoxins, chemicals)&indicates_risk(chemicals, animals, humans)]

level : 7
rule : rule(direct,
Above line :

Below line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [kind_of(aflatoxins, chemicals)]

level : 7
rule : rule(direct,
Above line :

Below line :
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [indicates_risk(chemicals, animals, humans)]
```

```
level : 5
rule : rule(all_elim,
Above line :
[all(path, all(species1, all(species2, current_assessment(aflatoxins, path, species2)&
        causes(aflatoxins, path, species1)&can_extrapolate(aflatoxins, species1, species2)
        ->causes(aflatoxins, path, species2)))),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2)&
        causes(agent, path, species1)&can_extrapolate(agent, species1, species2)
        ->causes(agent, path, species2))))),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 6
rule : rule(all_elim,
Above line :
[all(species1, all(species2, current_assessment(aflatoxins, cancer, species2)&
        causes(aflatoxins, cancer, species1)&can_extrapolate(aflatoxins, species1, species2)
        ->causes(aflatoxins, cancer, species2))),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(path, all(species1, all(species2, current_assessment(aflatoxins, path, species2)&
```

```
        causes(aflatoxins, path, species1)&can_extrapolate(aflatoxins, species1, species2)
        ->causes(aflatoxins, path, species2)))),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 7
rule : rule(all_elim,
Above line :
[all(species2, current_assessment(aflatoxins, cancer, species2)&
        causes(aflatoxins, cancer, animals)&can_extrapolate(aflatoxins, animals, species2)
        ->causes(aflatoxins, cancer, species2)),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(species1, all(species2, current_assessment(aflatoxins, cancer, species2)&
        causes(aflatoxins, cancer, species1)&can_extrapolate(aflatoxins, species1, species2)
        ->causes(aflatoxins, cancer, species2))),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 8
rule : rule(all_elim,
Above line :
[current_assessment(aflatoxins, cancer, humans)&
        causes(aflatoxins, cancer, animals)&can_extrapolate(aflatoxins, animals, humans)
        ->causes(aflatoxins, cancer, humans),
```

```
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)->
        maximum_acceptable_level(compound, path, species, min)))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(species2, current_assessment(aflatoxins, cancer, species2)&
        causes(aflatoxins, cancer, animals)&can_extrapolate(aflatoxins, animals, species2)
        ->causes(aflatoxins, cancer, species2)), can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min)))),
 all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 9
rule : rule(imp_elim,
Above line :
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min)))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [current_assessment(aflatoxins, cancer, humans)&
        causes(aflatoxins, cancer, animals)&can_extrapolate(aflatoxins, animals, humans)]

[causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min)))),
all(agent, all(path, all(level, causes(agent, path, humans)&
```

```
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level)))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[current_assessment(aflatoxins, cancer, humans)&causes(aflatoxins, cancer, animals)&
        can_extrapolate(aflatoxins, animals, humans)
        ->causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 10
rule : rule(and_intro,
Above line :
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins,
chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [current_assessment(aflatoxins, cancer, humans)]

[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins,
chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)->policy_level(agent, level))))]
|- [causes(aflatoxins, cancer, animals)&can_extrapolate(aflatoxins, animals, humans)]

Below line :
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
```

all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [current_assessment(aflatoxins, cancer, humans)&causes(aflatoxins, cancer, animals)&
can_extrapolate(aflatoxins, animals, humans)]

level : 11
rule : rule(direct,
Above line :

Below line :
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min)))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [current_assessment(aflatoxins, cancer, humans)]

level : 11
rule : rule(and_intro,
Above line :
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min)))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [causes(aflatoxins, cancer, animals)]

[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min
->maximum_acceptable_level(compound, path, species, min)))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [can_extrapolate(aflatoxins, animals, humans)]

Below line :
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins,
chemicals), minimum_detectable_level(aflatoxins, 20ppb),

all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min)))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [causes(aflatoxins, cancer, animals)&can_extrapolate(aflatoxins, animals, humans)]

level : 12
rule : rule(direct,
Above line :

Below line :
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxjns, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min)))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [causes(aflatoxins, cancer, animals)]

level : 12
rule : rule(direct,
Above line :

Below line :
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins,
chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min)))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [can_extrapolate(aflatoxins, animals, humans)]

level : 10
rule : rule(all_elim,
Above line :
[all(species, current_assessment(aflatoxins, cancer, species)
->no_safe_level(aflatoxins, cancer, species)),
causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),

minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[causes(aflatoxins, cancer, humans), can_extrapolatc(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(species, current_assessment(agent, cancer, species)
        ->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 11
rule : rule(all_elim,
Above line :
[current_assessment(aflatoxins, cancer, humans)
        ->no_safe_level(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(species, current_assessment(aflatoxins, cancer, species)
        ->no_safe_level(aflatoxins, cancer, species)),
causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 12

```
rule : rule(imp_elim,
Above line :
[causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb), all(compound, all(path, all(species, all(min,
no_safe_level(compound, path, species)&minimum_detectable_level(compound, min)-
>maximum_acceptable_level(compound, path, species, min))))), all(agent, all(path, all(level,
causes(agent, path, humans)&maximum_acceptable_level(agent, path, humans, level)-
>policy_level(agent, level))))]
|- [current_assessment(aflatoxins, cancer, humans)]

[no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[current_assessment(aflatoxins, cancer, humans)
        ->no_safe_level(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 13
rule : rule(direct,
Above line :

Below line :
[causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [current_assessment(aflatoxins, cancer, humans)]

level : 13
rule : rule(all_elim,
```

```
Above line :
[all(path, all(species, all(min, no_safe_level(aflatoxins, path, species)&
        minimum_detectable_level(aflatoxins, min)
        ->maximum_acceptable_level(aflatoxins, path, species, min)))),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species)&
        minimum_detectable_level(compound, min)
        ->maximum_acceptable_level(compound, path, species, min))))),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 14
rule : rule(all_elim,
Above line :
[all(species, all(min, no_safe_level(aflatoxins, cancer, species)&
        minimum_detectable_level(aflatoxins, min)
        ->maximum_acceptable_level(aflatoxins, cancer, species, min))),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(path, all(species, all(min, no_safe_level(aflatoxins, path, species)&
        minimum_detectable_level(aflatoxins, min)
        ->maximum_acceptable_level(aflatoxins, path, species, min)))),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 15
rule : rule(all_elim,
Above line :
```

[all(min, no_safe_level(aflatoxins, cancer, humans)&
minimum_detectable_level(aflatoxins, min)
->maximum_acceptable_level(aflatoxins, cancer, humans, min)),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(species, all(min, no_safe_level(aflatoxins, cancer, species)&
minimum_detectable_level(aflatoxins, min)
->maximum_acceptable_level(aflatoxins, cancer, species, min))),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 16
rule : rule(all_elim,
Above line :
[no_safe_level(aflatoxins, cancer, humans)& minimum_detectable_level(aflatoxins, 20ppb)
->maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(min, no_safe_level(aflatoxins, cancer, humans)&
minimum_detectable_level(aflatoxins, min)
->maximum_acceptable_level(aflatoxins, cancer, humans, min)),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 17
rule : rule(imp_elim,
Above line :
[no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),

causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))] |- [no_safe_level(aflatoxins, cancer,
humans)&minimum_detectable_level(aflatoxins, 20ppb)]

[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[no_safe_level(aflatoxins, cancer, humans)&
minimum_detectable_level(aflatoxins, 20ppb)
->maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 18
rule : rule(and_intro,
Above line :
[no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [no_safe_level(aflatoxins, cancer, humans)]

[no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))]
|- [minimum_detectable_level(aflatoxins, 20ppb)]

Below line :
[no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&

```
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level)))))]
|- [no_safe_level(aflatoxins, cancer, humans)&minimum_detectable_level(aflatoxins, 20ppb)]

level : 19
rule : rule(direct,
Above line :

Below line :
[no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level)))))]
|- [no_safe_level(aflatoxins, cancer, humans)]

level : 19
rule : rule(direct,
Above line :

Below line :
[no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level)))))]
|- [minimum_detectable_level(aflatoxins, 20ppb)]

level : 18
rule : rule(all_elim,
Above line :
[all(path, all(level, causes(aflatoxins, path, humans)&
        maximum_acceptable_level(aflatoxins, path, humans, level)
        ->policy_level(aflatoxins, level))),
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb),
all(agent, all(path, all(level, causes(agent, path, humans)&
        maximum_acceptable_level(agent, path, humans, level)
        ->policy_level(agent, level)))))]
|- [policy_level(aflatoxins, 20ppb)]

level : 19
```

```
rule : rule(all_elim,
Above line :
[all(level, causes(aflatoxins, cancer, humans)&
        maximum_acceptable_level(aflatoxins, cancer, humans, level)
        ->policy_level(aflatoxins, level)),
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(path, all(level, causes(aflatoxins, path, humans)&
        maximum_acceptable_level(aflatoxins, path, humans, level)
        ->policy_level(aflatoxins, level))),
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [policy_level(aflatoxins, 20ppb)]

level : 20
rule : rule(all_elim,
Above line :
[causes(aflatoxins, cancer, humans)&maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)
        ->policy_level(aflatoxins, 20ppb),
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[all(level, causes(aflatoxins, cancer, humans)&
        maximum_acceptable_level(aflatoxins, cancer, humans, level)
        ->policy_level(aflatoxins, level)),
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [policy_level(aflatoxins, 20ppb)]

level : 21
rule : rule(imp_elim,
Above line :
[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [causes(aflatoxins, cancer, humans)&
        maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)]
```

[policy_level(aflatoxins, 20ppb), maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [policy_level(aflatoxins, 20ppb)]

Below line :
[causes(aflatoxins, cancer, humans)&maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)
->policy_level(aflatoxins, 20ppb),
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [policy_level(aflatoxins, 20ppb)]

level : 22
rule : rule(and_intro,
Above line :
[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [causes(aflatoxins, cancer, humans)]

[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)]

Below line :
[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [causes(aflatoxins, cancer, humans)&
maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)]

level : 23
rule : rule(direct,
Above line :

Below line :
[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb), no_safe_level(aflatoxins, cancer,
humans), causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans),
current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb)]
|- [causes(aflatoxins, cancer, humans)]

level : 23
rule : rule(direct,

Above line :

Below line :
[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb), no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)]

level : 22
rule : rule(direct,
Above line :

Below line :
[policy_level(aflatoxins, 20ppb), maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb), no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)]
|- [policy_level(aflatoxins, 20ppb)]

Appendix C

The parsed proof of the FDA policy conjecture

The result of parsing the proof (described in section 6.4.3) is the following set of justifications. The third argument notes the inference rule from the proof from which the justification was abstracted.

```
justification(all(species1, all(species2, all(class, exists(path, current_assessment(aflatoxins, path,
species2)) & kind_of(aflatoxins, class) & indicates_risk(class, species1, species2)
->can_extrapolate(aflatoxins, species1, species2)))),
        [all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path,
species2)) & kind_of(agent, class) & indicates_risk(class, species1, species2)
->can_extrapolate(agent, species1, species2))))],
        all_elim)

justification(all(species2, all(class, exists(path, current_assessment(aflatoxins, path, species2)) &
kind_of(aflatoxins, class) & indicates_risk(class, animals, species2)
->can_extrapolate(aflatoxins, animals, species2))),
        [all(species1, all(species2, all(class, exists(path, current_assessment(aflatoxins, path,
species2)) & kind_of(aflatoxins, class) & indicates_risk(class, species1, species2)
->can_extrapolate(aflatoxins, species1, species2))))],
        all_elim)

justification(all(class, exists(path, current_assessment(aflatoxins, path, humans)) &
kind_of(aflatoxins, class) & indicates_risk(class, animals, humans)
->can_extrapolate(aflatoxins, animals, humans)),
        [all(species2, all(class, exists(path, current_assessment(aflatoxins, path, species2)) &
kind_of(aflatoxins, class) & indicates_risk(class, animals, species2)
->can_extrapolate(aflatoxins, animals, species2)))],
        all_elim)

justification(exists(path, current_assessment(aflatoxins, path, humans)) & kind_of(aflatoxins,
chemicals) & indicates_risk(chemicals, animals, humans)
->can_extrapolate(aflatoxins, animals, humans),
        [all(class, exists(path, current_assessment(aflatoxins, path, humans)) & kind_of(aflatoxins,
class) & indicates_risk(class, animals, humans)
->can_extrapolate(aflatoxins, animals, humans))],
        all_elim)

justification(can_extrapolate(aflatoxins, animals, humans),
        [exists(path, current_assessment(aflatoxins, path, humans)) & kind_of(aflatoxins, chemicals)
& indicates_risk(chemicals, animals, humans), exists(path, current_assessment(aflatoxins, path,
humans)) & kind_of(aflatoxins, chemicals) & indicates_risk(chemicals, animals, humans)
->can_extrapolate(aflatoxins, animals, humans)],
        imp_elim)

justification(exists(path, current_assessment(aflatoxins, path, humans)) & kind_of(aflatoxins,
chemicals) & indicates_risk(chemicals, animals, humans),
        [exists(path, current_assessment(aflatoxins, path, humans)), kind_of(aflatoxins, chemicals) &
indicates_risk(chemicals, animals, humans)],
        and_intro)
```

```
justification(exists(path, current_assessment(aflatoxins, path, humans)),
        [current_assessment(aflatoxins, cancer, humans)],
        exist_intro)

justification(current_assessment(aflatoxins, cancer, humans),
        [current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb), all(agent, all(path, all(species1, all(species2,
current_assessment(agent, path, species2) & causes(agent, path, species1) & can_extrapolate(agent,
species1, species2)->causes(agent, path, species2))))), all(agent, all(species, current_assessment(agent,
cancer, species)->no_safe_level(agent, cancer, species))), all(compound, all(path, all(species, all(min,
no_safe_level(compound, path, species) & minimum_detectable_level(compound, min)-
>maximum_acceptable_level(compound, path, species, min))))), all(agent, all(path, all(level,
causes(agent, path, humans) & maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))],
        direct)

justification(kind_of(aflatoxins, chemicals) & indicates_risk(chemicals, animals, humans),
        [kind_of(aflatoxins, chemicals), indicates_risk(chemicals, animals, humans)],
        and_intro)

justification(kind_of(aflatoxins, chemicals),
        [current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb), all(agent, all(path, all(species1, all(species2,
current_assessment(agent, path, species2) & causes(agent, path, species1) & can_extrapolate(agent,
species1, species2)->causes(agent, path, species2))))), all(agent, all(species, current_assessment(agent,
cancer, species)->no_safe_level(agent, cancer, species))), all(compound, all(path, all(species, all(min,
no_safe_level(compound, path, species) & minimum_detectable_level(compound, min)-
>maximum_acceptable_level(compound, path, species, min))))), all(agent, all(path, all(level,
causes(agent, path, humans) & maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))],
        direct)

justification(indicates_risk(chemicals, animals, humans),
        [current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals),
indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals),
minimum_detectable_level(aflatoxins, 20ppb), all(agent, all(path, all(species1, all(species2,
current_assessment(agent, path, species2) & causes(agent, path, species1) & can_extrapolate(agent,
species1, species2)->causes(agent, path, species2))))), all(agent, all(species, current_assessment(agent,
cancer, species)->no_safe_level(agent, cancer, species))), all(compound, all(path, all(species, all(min,
no_safe_level(compound, path, species) & minimum_detectable_level(compound, min)-
>maximum_acceptable_level(compound, path, species, min))))), all(agent, all(path, all(level,
causes(agent, path, humans) & maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))],
        direct)

justification(all(path, all(species1, all(species2, current_assessment(aflatoxins, path, species2) &
causes(aflatoxins, path, species1) & can_extrapolate(aflatoxins, species1, species2)
->causes(aflatoxins, path, species2)))),
        [all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2) &
causes(agent, path, species1) & can_extrapolate(agent, species1, species2)->causes(agent, path,
species2))))))],
        all_elim)

justification(all(species1, all(species2, current_assessment(aflatoxins, cancer, species2) &
```

causes(aflatoxins, cancer, species1) & can_extrapolate(aflatoxins, species1, species2)
->causes(aflatoxins, cancer, species2))),
[all(path, all(species1, all(species2, current_assessment(aflatoxins, path, species2) &
causes(aflatoxins, path, species1) & can_extrapolate(aflatoxins, species1, species2)->causes(aflatoxins,
path, species2)))))],
all_elim)

justification(all(species2, current_assessment(aflatoxins, cancer, species2) & causes(aflatoxins, cancer,
animals) & can_extrapolate(aflatoxins, animals, species2)->causes(aflatoxins, cancer, species2)),
[all(species1, all(species2, current_assessment(aflatoxins, cancer, species2) &
causes(aflatoxins, cancer, species1) & can_extrapolate(aflatoxins, species1, species2)
->causes(aflatoxins, cancer, species2)))],
all_elim)

justification(current_assessment(aflatoxins, cancer, humans) & causes(aflatoxins, cancer, animals) &
can_extrapolate(aflatoxins, animals, humans)->causes(aflatoxins, cancer, humans),
[all(species2, current_assessment(aflatoxins, cancer, species2) & causes(aflatoxins, cancer,
animals) & can_extrapolate(aflatoxins, animals, species2)->causes(aflatoxins, cancer, species2))],
all_elim)

justification(causes(aflatoxins, cancer, humans),
[current_assessment(aflatoxins, cancer, humans) & causes(aflatoxins, cancer, animals) &
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans) &
causes(aflatoxins, cancer, animals) & can_extrapolate(aflatoxins, animals, humans)->causes(aflatoxins,
cancer, humans)],
imp_elim)

justification(current_assessment(aflatoxins, cancer, humans) & causes(aflatoxins, cancer, animals) &
can_extrapolate(aflatoxins, animals, humans),
[current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals) &
can_extrapolate(aflatoxins, animals, humans)],
and_intro)

justification(current_assessment(aflatoxins, cancer, humans),
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer,
humans), causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb), all(agent, all(species,
current_assessment(agent, cancer, species)->no_safe_level(agent, cancer, species))),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) &
minimum_detectable_level(compound, min) ->maximum_acceptable_level(compound, path, species,
min))))), all(agent, all(path, all(level, causes(agent, path, humans) &
maximum_acceptable_level(agent, path, humans, level)->policy_level(agent, level))))],
direct)

justification(causes(aflatoxins, cancer, animals) & can_extrapolate(aflatoxins, animals, humans),
[causes(aflatoxins, cancer, animals), can_extrapolate(aflatoxins, animals, humans)],
and_intro)

justification(causes(aflatoxins, cancer, animals),
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer,
humans), causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans),
kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb), all(agent, all(species,
current_assessment(agent, cancer, species)->no_safe_level(agent, cancer, species))), all(compound,
all(path, all(species, all(min, no_safe_level(compound, path, species) &
minimum_detectable_level(compound, min)->maximum_acceptable_level(compound, path, species,
min))))), all(agent, all(path, all(level, causes(agent, path, humans) &
maximum_acceptable_level(agent, path, humans, level)->policy_level(agent, level))))],

direct)

justification(can_extrapolate(aflatoxins, animals, humans),
[can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb), all(agent, all(species, current_assessment(agent, cancer, species)->no_safe_level(agent, cancer, species))), all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) & minimum_detectable_level(compound, min) ->maximum_acceptable_level(compound, path, species, min))))), all(agent, all(path, all(level, causes(agent, path, humans) & maximum_acceptable_level(agent, path, humans, level) ->policy_level(agent, level))))],
direct)

justification(all(species, current_assessment(aflatoxins, cancer, species)
->no_safe_level(aflatoxins, cancer, species)),
[all(agent, all(species, current_assessment(agent, cancer, species)
->no_safe_level(agent, cancer, species)))],
all_elim)

justification(current_assessment(aflatoxins, cancer, humans)
->no_safe_level(aflatoxins, cancer, humans),
[all(species, current_assessment(aflatoxins, cancer, species)
->no_safe_level(aflatoxins, cancer, species))],
all_elim)

justification(no_safe_level(aflatoxins, cancer, humans),
[current_assessment(aflatoxins, cancer, humans), current_assessment(aflatoxins, cancer, humans)->no_safe_level(aflatoxins, cancer, humans)],
imp_elim)

justification(current_assessment(aflatoxins, cancer, humans),
[causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb), all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) & minimum_detectable_level(compound, min) ->maximum_acceptable_level(compound, path, species, min))))), all(agent, all(path, all(level, causes(agent, path, humans) & maximum_acceptable_level(agent, path, humans, level) ->policy_level(agent, level))))],
direct)

justification(all(path, all(species, all(min, no_safe_level(aflatoxins, path, species) & minimum_detectable_level(aflatoxins, min)
->maximum_acceptable_level(aflatoxins, path, species, min)))),
[all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) & minimum_detectable_level(compound, min)
->maximum_acceptable_level(compound, path, species, min))))],
all_elim)

justification(all(species, all(min, no_safe_level(aflatoxins, cancer, species) & minimum_detectable_level(aflatoxins, min)
->maximum_acceptable_level(aflatoxins, cancer, species, min))),
[all(path, all(species, all(min, no_safe_level(aflatoxins, path, species) & minimum_detectable_level(aflatoxins, min)
->maximum_acceptable_level(aflatoxins, path, species, min))))],
all_elim)

```
justification(all(min, no_safe_level(aflatoxins, cancer, humans) &
minimum_detectable_level(aflatoxins, min)
->maximum_acceptable_level(aflatoxins, cancer, humans, min)),
        [all(species, all(min, no_safe_level(aflatoxins, cancer, species) &
minimum_detectable_level(aflatoxins, min)
->maximum_acceptable_level(aflatoxins, cancer, species, min)))],
        all_elim)

justification(no_safe_level(aflatoxins, cancer, humans) & minimum_detectable_level(aflatoxins,
20ppb)->maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
        [all(min, no_safe_level(aflatoxins, cancer, humans) & minimum_detectable_level(aflatoxins,
min)->maximum_acceptable_level(aflatoxins, cancer, humans, min))],
        all_elim)

justification(maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
        [no_safe_level(aflatoxins, cancer, humans) & minimum_detectable_level(aflatoxins, 20ppb),
no_safe_level(aflatoxins, cancer, humans) & minimum_detectable_level(aflatoxins, 20ppb)
->maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)],
        imp_elim)

justification(no_safe_level(aflatoxins, cancer, humans) & minimum_detectable_level(aflatoxins,
20ppb),
        [no_safe_level(aflatoxins, cancer, humans), minimum_detectable_level(aflatoxins, 20ppb)],
        and_intro)

justification(no_safe_level(aflatoxins, cancer, humans),
        [no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins,
chemicals), minimum_detectable_level(aflatoxins, 20ppb), all(agent, all(path, all(level, causes(agent,
path, humans) & maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))],
        direct)

justification(minimum_detectable_level(aflatoxins, 20ppb),
        [no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans),
can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans),
causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins,
chemicals), minimum_detectable_level(aflatoxins, 20ppb), all(agent, all(path, all(level, causes(agent,
path, humans) & maximum_acceptable_level(agent, path, humans, level)
->policy_level(agent, level))))],
        direct)

justification(all(path, all(level, causes(aflatoxins, path, humans) &
maximum_acceptable_level(aflatoxins, path, humans, level)->policy_level(aflatoxins, level))),
        [all(agent, all(path, all(level, causes(agent, path, humans) &
maximum_acceptable_level(agent, path, humans, level)->policy_level(agent, level))))],
        all_elim)

justification(all(level, causes(aflatoxins, cancer, humans) & maximum_acceptable_level(aflatoxins,
cancer, humans, level)->policy_level(aflatoxins, level)),
        [all(path, all(level, causes(aflatoxins, path, humans) &
maximum_acceptable_level(aflatoxins, path, humans, level)->policy_level(aflatoxins, level)))],
        all_elim)]

justification(causes(aflatoxins, cancer, humans) & maximum_acceptable_level(aflatoxins, cancer,
```

humans, 20ppb)->policy_level(aflatoxins, 20ppb),
[all(level, causes(aflatoxins, cancer, humans) & maximum_acceptable_level(aflatoxins, cancer, humans, level)->policy_level(aflatoxins, level))],
all_elim)

justification(policy_level(aflatoxins, 20ppb),
[causes(aflatoxins, cancer, humans) & maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb), causes(aflatoxins, cancer, humans) & maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb) ->policy_level(aflatoxins, 20ppb)],
imp_elim)

justification(causes(aflatoxins, cancer, humans) & maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
[causes(aflatoxins, cancer, humans), maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb)],
and_intro)

justification(causes(aflatoxins, cancer, humans),
[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb), no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)],
direct)

justification(maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
[maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb), no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)],
direct)

justification(policy_level(aflatoxins, 20ppb),
[policy_level(aflatoxins, 20ppb), maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb), no_safe_level(aflatoxins, cancer, humans), causes(aflatoxins, cancer, humans), can_extrapolate(aflatoxins, animals, humans), current_assessment(aflatoxins, cancer, humans), causes(aflatoxins, cancer, animals), indicates_risk(chemicals, animals, humans), kind_of(aflatoxins, chemicals), minimum_detectable_level(aflatoxins, 20ppb)],
direct)

Appendix D

Merged justifications

The result of exhaustively *merging* (described in section 6.4.4) the justifications given in Appendix C is is the following set of justifications.

justification(causes(aflatoxins, cancer, humans) & maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
[all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2) & causes(agent, path, species1) & can_extrapolate(agent, species1, species2)->causes(agent, path, species2))))),
causes(aflatoxins, cancer, animals),
all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path, species2)) & kind_of(agent, class) & indicates_risk(class, species1, species2)->can_extrapolate(agent, species1, species2))))),
kind_of(aflatoxins, chemicals), indicates_risk(chemicals, animals, humans),
minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) & minimum_detectable_level(compound, min) ->maximum_acceptable_level(compound, path, species, min))))), current_assessment(aflatoxins, cancer, humans),
all(agent, all(species, current_assessment(agent, cancer, species) ->no_safe_level(agent, cancer, species)))],
merged)

justification(policy_level(aflatoxins, 20ppb),
[all(agent, all(path, all(level, causes(agent, path, humans) & maximum_acceptable_level(agent, path, humans, level)->policy_level(agent, level)))),
all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2) & causes(agent, path, species1) & can_extrapolate(agent, species1, species2) ->causes(agent, path, species2))))),
causes(aflatoxins, cancer, animals),
all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path, species2)) & kind_of(agent, class) & indicates_risk(class, species1, species2) ->can_extrapolate(agent, species1, species2))))),
kind_of(aflatoxins, chemicals), indicates_risk(chemicals, animals, humans),
minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) & minimum_detectable_level(compound, min) ->maximum_acceptable_level(compound, path, species, min))))), current_assessment(aflatoxins, cancer, humans),
all(agent, all(species, current_assessment(agent, cancer, species) ->no_safe_level(agent, cancer, species)))],
merged)

justification(causes(aflatoxins, cancer, humans) & maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb) ->policy_level(aflatoxins, 20ppb),
[all(agent, all(path, all(level, causes(agent, path, humans) & maximum_acceptable_level(agent, path, humans, level) ->policy_level(agent, level))))],
merged)

justification(all(level, causes(aflatoxins, cancer, humans) & maximum_acceptable_level(aflatoxins, cancer, humans, level) ->policy_level(aflatoxins, level)),
[all(agent, all(path, all(level, causes(agent, path, humans) & maximum_acceptable_level(agent, path, humans, level) ->policy_level(agent, level))))],
merged)

justification(no_safe_level(aflatoxins, cancer, humans) & minimum_detectable_level(aflatoxins, 20ppb),
[minimum_detectable_level(aflatoxins, 20ppb), current_assessment(aflatoxins, cancer, humans), all(agent, all(species, current_assessment(agent, cancer, species)->no_safe_level(agent, cancer, species)))],
merged)

justification(maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
[minimum_detectable_level(aflatoxins, 20ppb),
all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) & minimum_detectable_level(compound, min) ->maximum_acceptable_level(compound, path, species, min))))), current_assessment(aflatoxins, cancer, humans),
all(agent, all(species, current_assessment(agent, cancer, species)->no_safe_level(agent, cancer, species)))],
merged)

justification(no_safe_level(aflatoxins, cancer, humans) & minimum_detectable_level(aflatoxins, 20ppb)
->maximum_acceptable_level(aflatoxins, cancer, humans, 20ppb),
[all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) & minimum_detectable_level(compound, min) ->maximum_acceptable_level(compound, path, species, min))))],
merged)

justification(all(min, no_safe_level(aflatoxins, cancer, humans) & minimum_detectable_level(aflatoxins, min)
->maximum_acceptable_level(aflatoxins, cancer, humans, min)),
[all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) & minimum_detectable_level(compound, min) ->maximum_acceptable_level(compound, path, species, min))))],
merged)

justification(all(species, all(min, no_safe_level(aflatoxins, cancer, species) & minimum_detectable_level(aflatoxins, min) ->maximum_acceptable_level(aflatoxins, cancer, species, min))),
[all(compound, all(path, all(species, all(min, no_safe_level(compound, path, species) & minimum_detectable_level(compound, min) ->maximum_acceptable_level(compound, path, species, min))))],
merged)

justification(no_safe_level(aflatoxins, cancer, humans),
[current_assessment(aflatoxins, cancer, humans), all(agent, all(species, current_assessment(agent, cancer, species) ->no_safe_level(agent, cancer, species)))],
merged)

justification(current_assessment(aflatoxins, cancer, humans)->no_safe_level(aflatoxins, cancer, humans),
[all(agent, all(species, current_assessment(agent, cancer, species)->no_safe_level(agent, cancer, species)))],
merged)

justification(causes(aflatoxins, cancer, animals) & can_extrapolate(aflatoxins, animals, humans),
[causes(aflatoxins, cancer, animals), all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path, species2)) & kind_of(agent, class) & indicates_risk(class, species1, species2) ->can_extrapolate(agent, species1, species2))))), current_assessment(aflatoxins,

```
cancer, humans),
kind_of(aflatoxins, chemicals), indicates_risk(chemicals, animals, humans)],
        merged)

justification(current_assessment(aflatoxins, cancer, humans) & causes(aflatoxins, cancer, animals) &
can_extrapolate(aflatoxins, animals, humans),
        [causes(aflatoxins, cancer, animals),
all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path, species2)) &
kind_of(agent, class) & indicates_risk(class, species1, species2)
->can_extrapolate(agent, species1, species2))))), current_assessment(aflatoxins, cancer, humans),
kind_of(aflatoxins, chemicals), indicates_risk(chemicals, animals, humans)],
        merged)

justification(causes(aflatoxins, cancer, humans),
        [all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2) &
causes(agent, path, species1) & can_extrapolate(agent, species1, species2) ->causes(agent, path,
species2))))),
causes(aflatoxins, cancer, animals),
all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path, species2)) &
kind_of(agent, class) & indicates_risk(class, species1, species2)
->can_extrapolate(agent, species1, species2))))), current_assessment(aflatoxins, cancer, humans),
kind_of(aflatoxins, chemicals), indicates_risk(chemicals, animals, humans)],
        merged)

justification(current_assessment(aflatoxins, cancer, humans) & causes(aflatoxins, cancer, animals) &
can_extrapolate(aflatoxins, animals, humans) ->causes(aflatoxins, cancer, humans),
        [all(agent, all(path, all(species1, all(species2,
current_assessment(agent, path, species2) & causes(agent, path, species1) &
can_extrapolate(agent, species1, species2) ->causes(agent, path, species2)))))],
        merged)

justification(all(species2, current_assessment(aflatoxins, cancer, species2) & causes(aflatoxins, cancer,
animals) & can_extrapolate(aflatoxins, animals, species2) ->causes(aflatoxins, cancer, species2)),
        [all(agent, all(path, all(species1, all(species2,
current_assessment(agent, path, species2) & causes(agent, path, species1) & can_extrapolate(agent,
species1, species2) ->causes(agent, path, species2)))))],
        merged)

justification(all(species1, all(species2, current_assessment(aflatoxins, cancer, species2) &
causes(aflatoxins, cancer, species1) & can_extrapolate(aflatoxins, species1, species2) -
>causes(aflatoxins, cancer, species2))),
        [all(agent, all(path, all(species1, all(species2, current_assessment(agent, path, species2) &
causes(agent, path, species1) & can_extrapolate(agent, species1, species2) ->causes(agent, path,
species2))))))],
        merged)

justification(exists(path, current_assessment(aflatoxins, path, humans)) & kind_of(aflatoxins,
chemicals) & indicates_risk(chemicals, animals, humans),
        [current_assessment(aflatoxins, cancer, humans), kind_of(aflatoxins, chemicals),
indicates_risk(chemicals, animals, humans)],
        merged)

justification(can_extrapolate(aflatoxins, animals, humans),
        [all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path,
species2)) & kind_of(agent, class) & indicates_risk(class, species1, species2)->can_extrapolate(agent,
species1, species2))))),current_assessment(aflatoxins, cancer, humans), kind_of(aflatoxins, chemicals),
indicates_risk(chemicals, animals, humans)],
```

merged)

justification(exists(path, current_assessment(aflatoxins, path, humans)) & kind_of(aflatoxins, chemicals) & indicates_risk(chemicals, animals, humans)->can_extrapolate(aflatoxins, animals, humans),
[all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path, species2)) & kind_of(agent, class) & indicates_risk(class, species1, species2)->can_extrapolate(agent, species1, species2))))],
merged)

justification(all(class, exists(path, current_assessment(aflatoxins, path, humans)) & kind_of(aflatoxins, class) & indicates_risk(class, animals, humans) ->can_extrapolate(aflatoxins, animals, humans)),
[all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path, species2)) &
kind_of(agent, class) & indicates_risk(class, species1, species2) ->can_extrapolate(agent, species1, species2))))],
merged)

justification(all(species2, all(class, exists(path, current_assessment(aflatoxins, path, species2)) & kind_of(aflatoxins, class) & indicates_risk(class, animals, species2) ->can_extrapolate(aflatoxins, animals, species2))),
[all(agent, all(species1, all(species2, all(class, exists(path, current_assessment(agent, path, species2)) &
kind_of(agent, class) & indicates_risk(class, species1, species2) ->can_extrapolate(agent, species1, species2))))],
merged)