In: *Computational Intelligence*, **11**, 113-131, 1995.

A Logic of Argumentation for Reasoning under Uncertainty.

Paul Krause[1], **Simon Ambler**[2], **Morten Elvang-Gøransson**[1,3] **and John Fox**[1]

[1] Imperial Cancer Research Fund
London WC2A 3PX, UK

[2] Department of Computer Science and Statistics, Queen Mary and Westfield College
London E1 4NS, UK

[3] Space Division, Computer Resources International
DK-3460 Birkeroed, Denmark

Abstract

We present the syntax and proof theory of a logic of argumentation, LA. We also outline the development of a category theoretic semantics for LA. LA is the core of a proof theoretic model for reasoning under uncertainty. In this logic, propositions are labelled with a representation of the arguments which support their validity. Arguments may then be aggregated to collect more information about the potential validity of the propositions of interest. We make the notion of aggregation primitive to the logic, and then define strength mappings from sets of arguments to one of a number of possible dictionaries. This provides a uniform framework which incorporates a number of numerical and symbolic techniques for assigning subjective confidences to propositions on the basis of their supporting arguments. These aggregation techniques are also described, with examples.

Key words: Uncertain reasoning, epistemic probability, argumentation, non-classical logics, non-monotonic reasoning

1. Introduction

The work presented in this paper should be viewed as a study of epistemic probability (Shafer 1978). That is, of the probability of an event, or confidence in a proposition, as an attribute of opinion based on an appraisal of the arguments concerning that event or proposition. Epistemic probability is, in a sense, a meta-level concept. We propose arguments concerning a claim of interest at the object level, and then reason *about* those arguments to evaluate their relative strength.

At the object level, inference is carried out using a "Logic of Argumentation", LA. The starting position is that arguments have the *form* of logical proof, but they do not have the *force* of logical proof. So we take an existing logic, (&, $\supset$) minimal logic, as the basis for LA. Arguments are proofs in minimal logic. However, as they do not necessarily have the force of certainty we wish to make a notion of *aggregation* primitive to the logic; as we aggregate arguments for a proposition, so we collect more information about that proposition's potential validity.

We will provide a syntax and outline the development of a semantics for LA. We will then discuss a variety of numerical and symbolic methods for deriving figures of confidence in propositions based on the aggregate of their supporting arguments. This provides and illustrates certain minimum criteria which a scheme of argumentation should conform to. That is, (sets of) arguments should form an "evidential closed category" and their strength should factor through a "confidence measure". The concept of an evidential closed category will be introduced and illustrated

with one concrete example (sets of λ-terms). Three examples of how strengths might arise will be described:

(i) simple weights in {0,+,++}, [0,1];
(ii) probability of provability;
(iii) a purely symbolic form of dialectical argumentation.

There is an underlying agenda of this paper. At its core, the work reported here explores the extent to which recent directions being taken in the logic in computer science community may be extended to provide insight into the preoccupations of the artificial intelligence (AI) community. Argument systems are a significant focus of attention in AI. In an analogous way, significant attention is being paid to the study of proofs as entities in their own right by logicians in computer science. The latter leads in a natural way to the abstract modelling of deductive systems and a school of thought that such systems can be usefully modelled using category theory. That is the route we have followed. It provides us with a uniform framework within which we can provide a semantics for argumentation *and* aggregation.

Early versions of the work contained in this paper were originally presented at IPMU '92 (Krause, Ambler and Fox 1993) and at the 9th Conference on Uncertainty in Artificial Intelligence (Elvang-Gøransson, Krause and Fox 1993). The former contained a presentation of the proof rules and outlined the development of the semantics of LA and aggregation. The latter paper contained the definition of the "acceptability" classes discussed here in sub-section 6.3. This paper expands on the material presented in both these papers, with a more detailed discussion of LA and with additional material on the properties of and rationale behind the model of symbolic argumentation discussed in section 6.

2. Overview of the paper

In section 3 we present and discuss the syntax and inference rules of LA. We then outline the development of an "argumentation theorem prover" in section 4. This completes the purely syntactic treatment of LA. However, one certain obligation with the development of a formal logic is to attach a meaning to that logic. That is, the logic must have a formal semantics. We have developed a proof theoretic semantics of LA which is expressed using category theory.

We give LA a semantics by ascribing a meaning to proofs in LA. The propositions themselves are then given a meaning by assigning to them a value corresponding to the *confidence* ascribed to them by their associated arguments. The development of the semantics is outlined, and aggregation and confidence values discussed, in section 5.

We wish to enable reasoning about the arguments in a variety of symbolic and numerical ways. We describe an experiment for the purely symbolic aggregation of arguments in section 6. This draws on a proposal for reasoning with an inconsistent database. Reasoning in the presence of inconsistency is a problem fraught with difficulties, and we are not claiming a complete solution here. However, we feel that it is a sufficiently interesting experiment, capturing a number of intuitions, that it is worth reporting.

Finally, section 7 provides a concluding discussion of some of the strengths and weaknesses of this approach.

3. A logic for identifying arguments

3.1 The choice of logic and argument representation.

As mentioned in the introduction, we decompose argumentation into two components; the construction of arguments themselves, and the reasoning about arguments at the meta-level. We will present the syntax and inference rules of a logic for the generation of arguments, LA, in this section, and outline how this may be turned into a mechanical procedure in the next section.

The proposal is not so much to generate a new logic as to augment an existing logic; we wish to annotate propositions with a representation of their proofs. That is, we wish to use a labelled deductive system (Gabbay 1990), where the labels represent the arguments supporting the subject of the deduction.

What logic should we use to represent the underlying reasoning process? An idealised reasoning agent may construct arguments for a proposition, and may construct arguments for its negation. However, there may be one or more propositions p such that neither p, nor its negation ¬p is a member of that agent's current state of knowledge (assumed to be deductively closed). This reflects a current state of ignorance about p in which it is meaningless to say that p is either true or it is false. For "idealised reasoning agent", substitute "(idealised) mathematician" and we have a situation precisely analogous to the starting point of intuitionistic logic. That is, the logic underlying argumentation is intuitionistic logic (Heyting 1956). In fact, attention will be restricted to the &, $\supset$, $\perp$ (and, implication, contradiction) fragment of minimal logic. Note that minimal logic is weaker than intuitionistic logic in that it does not have $\perp$ entails everything.

We next need to augment the logic by labelling propositions with the arguments which support those propositions. This enables us to reason about the arguments themselves. The first requirement for any formal theory of argumentation is that we be able to identify some criteria for judging when arguments are distinct. This is a weaker condition than the requirement for arguments to be independent, which we may also need, but is a necessary first step.

For example, in order to conclude whether $A \supset B$ is derivable from a knowledge base containing the axiom $A \supset B$, one may just say; "I know $A \supset B$; it is an axiom" (we use $\supset$ here to represent implication). Alternatively, one may be a little more perverse and use the same information in a more complex way. "If I assume *A*, then apply the axiom $A \supset B$ to it, I may conclude *B*. That is, if A is true, then so is B: I conclude $A \supset B$". Both proofs have used essentially the same information, but in a slightly different way. Are they the same, or are they different? The development of an equational theory over proofs has received a great deal of thought from logicians, and there is a well established body of work on typed proof theory which may be applied to this problem (Girard *et al.* 1989; Hindley and Seldin 1986).

We use typed lambda terms as proof terms. This gives us an equational theory over proof terms which has a formal agreed basis, so that we can say when two proofs are equal (are reducible to the same normal form). In this proof theoretic view of logic, a proof *x* of a formula *A* is equivalent to saying that the 'proof term' *x* is of type *A* (written *x:A*). Then if we have a proof *x:A* and a proof *y:B* for example, pairing *x* with *y* will yield a proof *pair(x,y): A & B* of the conjunction. The functional view of proofs first becomes apparent on considering a proof of an implication. By stating that f: $A \supset B$ we are saying that *f* is a function which takes a proof term of type *A* as a parameter and returns a proof term of type *B*.

Returning to the question of equivalence between two proofs posed above, the second proof is an

example of hypothetical reasoning. By assuming *A* we generate a proof of *B*, and so conclude $A \supset B$. By the above, we would expect the proof term corresponding to the conclusion to be a function and this is where the notation of the lambda calculus comes into play. From the assumption *x:A* a proof *v:B* is derived. The lambda-term $\lambda x.v$ denotes a function which takes a single argument *u* of type *A* and returns a term *[u/x]v* of type *B* (in which all occurrences of *x* in *v* are substituted by *u*). That is, $\lambda x.v$ is the lambda abstraction of the proof of *B*; lambda abstraction corresponds to the discharging of assumptions. In our specific example, if *f*: $A \supset B$ then $\lambda x.apply(f,x)$ is also of type $A \supset B$. But by eta-conversion, (Hindley and Seldin 1986), $\lambda x.apply(f,x)$ reduces to *f*; both proofs reduce to the same normal form and hence are equivalent.

3.2 Syntax of LA.

The current version of LA employs the "and, implication, contradiction" fragment of intuitionistic logic (minimal logic). The symbol '&' will be used to represent *conjunction*, $\supset$ to represent *implication*, and $\perp$ to represent *falsum* or "contradiction". Only the propositional case will be considered. As is usual with minimal or intuitionistic logic, negation is interpreted in terms of implication by defining $\neg\varphi$ to be $\varphi \supset \perp$.

The inference rules of LA manipulate labelled formulae, where:

labelled_formulae ::= *arg*:*formula*

The syntax of formulae is:

formula ::= *prop* | *formula* & *formula* | *formula* $\supset$ *formula*

prop ::= $\perp$ | $\{p_0, p_1, p_2, \ldots\}_1$

As mentioned, formulae are labelled with a representation of the arguments which support them. In the case of axioms, the labels are just atoms (which may follow some naming convention about the source or type of the information they represent). In general the labels are terms in the typed λ-calculus. The types are propositions in the aforementioned logic. The construction *x*:A is to be read as "*x* is a (proof) term of type A", or, "*x* represents a proof of the proposition A". In fact, to be consistent with our line of thinking, *x* is best thought of as an *argument* of type A.

We will outline the notation of the λ-calculus and its interpretation as arguments. Terms must conform to the following syntax:

arg ::= *atom* | *var* | apply(*arg*,*arg*) | λ*var*. *arg* | fst(*arg*) | snd(*arg*) | pair(*arg*,*arg*)

var ::= $\{x_0, x_1, x_2, \ldots\}_1$

atom ::= $\{a_0, a_1, a_2, \ldots\}_1$

Atomic arguments are names of axioms. The others may be read as follows:

- apply(*arg1*, *arg2*). *arg1* will be an argument supporting an inference rule, *arg2* an argument for the antecedent of that rule. By 'applying' the rule to the antecedent we can conclude the consequent of that rule. Hence apply(*arg1*, *arg2*) is an argument for the consequent of the rule supported by *arg1*.
- λ*var*. *arg*. This is an argument supporting an implication A $\supset$ B. *arg* may contain one or more occurrences of the terminal *var*. By substituting an argument of type A for all occurrences of *var* in *arg*, an argument for B will be obtained.
- fst(*arg*). This 'projects' the first formula in a conjunction. If *arg* is an argument for A & B,

then fst(*arg*) is an argument for A.

- snd(*arg*). This 'projects' the second formula in a conjunction. If *arg* is an argument for A & B, then snd(*arg*) is an argument for B.
- pair(*arg1, arg2*). This is an argument for a conjunction A & B, where *arg1* is an argument supporting A and *arg2* an argument supporting B.

3.3 Presentation of LA.

In the preceding two sections, the logic underlying argumentation has been introduced, together with the notation for representing arguments themselves. We will now give a *sequent calculus* presentation of the "logic of argumentation". This form of presentation has been chosen to explicate the equivalence between proofs in LA and their representation as terms in the typed λ-calculus.

A sequent is usually written as an expression $\underline{A} \Rightarrow \underline{B}$; where $\Rightarrow$ represents an entailment relation, $\underline{A}$ and $\underline{B}$ are finite sequences of formulae (Gallier 1987). A naive interpretation of a sequent is that the conjunction of the formulae in $\underline{A}$ implies the disjunction of the formulae in $\underline{B}$. In our presentation, the sequents are of the form:

$$\text{Context} \Rightarrow \text{Term: Formula}$$

This may be read informally as "an argument, *Term*, for *Formula* can be constructed from the arguments for the formulae in *Context*". For example, with the context $\Gamma = \{a1\text{: A}, r1\text{: A} \supset \text{B}\}$, the sequent $\Gamma \Rightarrow apply(r1, a1)\text{: B}$ would be constructible using LA.

$$\text{Axiom} \quad \frac{}{\Gamma, \text{x: A} \Rightarrow \text{x: A}}$$

$$(\& \Rightarrow) \quad \frac{\text{x: A, y:B, } \Gamma \Rightarrow \text{u: C}}{\Gamma, \text{z: A \& B} \Rightarrow [\text{fst(z)/x}][\text{snd(z)/y}]\text{u: C}} \qquad (\Rightarrow \&) \quad \frac{\Gamma \Rightarrow \text{x: A} \qquad \Gamma \Rightarrow \text{y:B}}{\Gamma \Rightarrow \text{pair(x,y): A \& B}}$$

$$(\supset \Rightarrow) \quad \frac{\Gamma, \text{f: A} \supset \text{B} \Rightarrow \text{s: A} \qquad \text{x: B, } \Gamma \Rightarrow \text{u: C}}{\Gamma, \text{f: A} \supset \text{B} \Rightarrow [\text{apply(f, s)/x}]\text{u: C}} \qquad (\Rightarrow \supset) \quad \frac{\text{x: A, } \Gamma \Rightarrow \text{t: B}}{\Gamma \Rightarrow \lambda\text{x.t: A} \supset \text{B}}$$

Figure 1: Rule set defining the construction of arguments with their associated proof terms. (By [u/x]v, we mean the term generated by substituting u for x in v).

These arguments may be generated by a sequence of rule applications using the set of inference rules presented in Figure 1. In these rules Γ is a multiset of declarations *x*: *A* (*x* is a variable of type *A*) and 'Γ, *x*:*A*' denotes the multiset Γ extended by *x*:*A*. The rule labelled '⇒ &' may be read, for example, as; "if *x* is an argument for *A* and *y* is an argument for *B*, derivable from a context Γ, then *pair(x,y)* is an argument for *A & B* from the same context". Note that there are no explicit rules for negation; $\neg A$ is translated into $A \supset \perp$ before applying the rules. In addition, ⊥ should be read as "contradiction", which is intended as a slightly weaker concept than the usual logical reading as *falsum*. A knowledge base which enables support for the contradiction to be derived is not necessarily pathological; consequently we do not admit a rule which propagates support for ⊥ to

all propositions. (It should, however, be noted that as A ⊃ (B ⊃ A) is a tautology of LA, support for ⊥ will be propagated to all negated propositions ¬P (≡ P ⊃ ⊥)).

The following notes should aid in interpreting the rules:

(⇒ ⊃) The derivation of an implication is described in the example at the end of section 3.1.

(& ⇒) Suppose in a context Γ extended by assuming an argument *x* for *A* and *y* for *B* we have been able to construct an argument *u* for *C*. Then if we extend the same context Γ by assuming an argument *z* for *A* & *B*, we can also construct an argument for *C*. This is obtained by substituting the first projection of *z* (an argument for *A*) for *x* in *u*, and the second projection of *z* (an argument for *B*) for *y* in *u*.

(⊃ ⇒) A specific instance may help to clarify this rule. By one application of the rule "Axiom" we can generate the sequent Γ, *x*: *B* ⇒ *x*: *B*. Substituting this on the right hand side of (⊃ ⇒), the rule states that if we can generate an argument *s* for *A* in a context Γ, *f*:*A* ⊃ *B*, then we can also generate an argument *apply(f,s)* for *B* in the same context; the rule is a generalisation of modus ponens.

4. Outline of the Argumentation Theorem Prover (ATP)

We have presented LA by writing down its syntax and inference rules. An advantage of this style of presentation is that it is relatively straightforward to move from the proof rules to an implementation of a theorem prover for LA. We will give an outline of the argumentation theorem prover (ATP) here.

With a given context, Γ, and a formula *C* to be proven, the general strategy is to run the rules of Figure 1 backwards to generate the proof term which corresponds to an argument from Γ supporting *C*. Unfortunately, application of '⊃ ⇒' will not necessarily result in simpler sub-formulae. For example, in attempting to construct an argument supporting *A* from a context Γ = {*r1*: *A*⊃*B*, *r2*: *B*⊃*A*} (which should fail) a non-terminating recursive loop will be generated through this rule. (To see this, again trivialise the right hand sequent above the line. Then, we wish to construct an argument for A from the context {*r1*: *A*⊃*B*, *r2*: *B*⊃*A*}. The rule shows us how to do this if we can construct an argument *s* for *B* from the same context. Now focus on axiom *r1*. We can use '⊃ ⇒' again to show us how to construct an argument for *B* from the context {*r2*: *B*⊃*A*, *r1*: *A*⊃*B*} if we can construct an argument *s'* for *A* from the same context. But we are now back where we started!)

There are two strategies for eliminating this problem. One could use a programming solution and incorporate some form of loop checking into the theorem prover. Or one can use a modified rule set which is known to allow the same set of theorems, but which avoids the use of the problematic rule '⊃ ⇒' (Dyckhoff 1992). We have versions of the 'ATP' which use each of these approaches.

One needs to do a bit more work than just to apply the rule set in its raw form. The theorem prover should return all *distinct* arguments. So, having used the rule set to generate a proof term as a representation of an argument, it now needs to be reduced to its normal form. Then the distinct normal forms correspond to the distinct arguments which we require. This reduction can be achieved by applying the following rules. The term will be in normal form if it cannot be further reduced by applying any of these rules to the term itself or to any of its sub terms (read "»" as "reduces to"):

fst(pair(*u*,*v*)) » *u* snd(pair(*u*,*v*)) » *v*

pair(fst(*t*), snd(*t*)) » *t*

apply($\lambda x.v,u$) » $[u/x]v$	('beta-conversion')
λx.apply(t,x) » t (x not free in t)	('eta-conversion')

It is a consequence of the *Church-Rosser property* and the *strong normalisation theorem* (Hindley and Seldin 1986) that a reduction using these rules will always terminate with the unique normal form. As with the generation of the term in the first place, one needs to do a little more work to run this part of the program effectively. In particular, we use the de Bruijn notation for lambda calculus to keep track of variables in the terms (de Bruijn 1972).

It should be emphasised that a significant factor in the choice of this labelling scheme is that the λ-term representation of the argument is machine readable. As well as being automatically constructible using the ATP, the terms can be mechanically read and interpreted for a variety of purposes. For example, it is possible to write a program which will "type" a λ-term in a given context. That is, given an argument as a λ-term, one can generate the proposition for which it is an argument. This is useful, for example, in the generation of the textual explanations for sub-components of an argument. It may also have application to forward chaining as opposed to theorem proving in controlling the type of information that is derived from a given context, although we have not explored this.

5. The semantics of LA and the aggregation of arguments

5.1 Proof theoretic semantics and category theory.

Thus far, the paper has focused on syntactic aspects of argumentation. A mechanism has been set up to enable arguments to be constructed which support, or oppose, propositions of interest as information becomes available. As they are identified, support from these arguments then needs to be aggregated to obtain more information about the potential validity of the propositions. We may be able to refine the aggregation procedure further by associating a measure of *confidence* to the arguments. In order to do this, we will need to take an algebraic view of argumentation. To that effect, we will give the semantics of LA a precise mathematical meaning by giving a category theoretic account of arguments. This enables the provision of a unified account of the semantics of argumentation *and* aggregation.

One approach to the provision of a semantics is to use truth tables, whereby any sentence in the logic may be assigned a truth value. Another approach is to use *model theory*, whereby meaning is ascribed in terms of possible *interpretations* of the sentences in the logic. In either case we ascribe a semantics to the logic by saying what the sentences in the logic *denote*. Denotations for a logic are normally either *t* (true) or *f* (false). This is the Tarskian tradition in semantics.

An alternative approach is not to ask "when is a sentence A *true*" (*à la* Tarskian semantics), but to ask "what is a *proof* of A?". We ascribe a meaning to the proofs in the logic, and not to the sentences themselves. In terms of LA, we ascribe meaning to the *arguments* in LA. As arguments are what makes LA different from other logics, it seems sensible to focus attention on them. We have developed a proof theoretic semantics of LA which is expressed using category theory. Category theory is a relatively recent branch of mathematics and familiarity with its concepts is not widespread. So a full presentation of the semantics would unbalance and reduce the accessibility of this paper. Consequently, we only outline this work here. A more complete treatment can be found in (Ambler 1992).

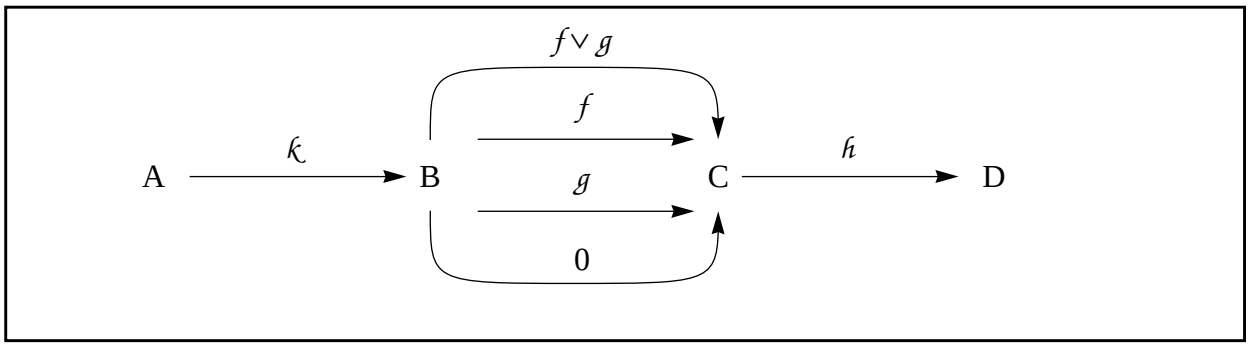

Figure 2: Schematic ordering on arrows.

5.2 Aggregation of arguments and confidence measures.

A category $\mathcal{C}$ has an associated set of 'objects' and a set of 'arrows'. In particular, a *deductive system* may be modelled as a category where the objects are propositions (A, B, …) and the arrows ($f, g, \ldots$) correspond to proofs (Lambek and Scott 1986). For example, the arrow f: A $\rightarrow$ B corresponds to a proof of A $\vdash$ B. We may compose arrows, so that if f corresponds to a proof of B from A, and g corresponds to a proof of C from B, then the composition gf will correspond to a proof of C from A. That is, the deductive system has the following rule of inference:

$$\frac{f\text{: A} \rightarrow \text{B} \qquad g\text{: B} \rightarrow \text{C}}{gf\text{: A} \rightarrow \text{C}}$$

Further discussion of the conditions under which a deductive system is a category can be found in, for example, (Lambek and Scott 1986). In the case of LA, (sets of) arguments are modelled as the arrows of a semi-lattice enriched category (Kelly 1982). The epithet "semi-lattice enriched" means that the category structure is enriched with some ordering information on the arrows.

In the case of intuitionistic logic, there is a well-trodden route to the development of a categorical semantics. The link we have used in Section 3 between proofs and typed lambda-terms is formally established by the Curry-Howard isomorphism (Girard *et al.* 1989). Then the relationship between typed λ-calculus and Cartesian closed categories, (Lambek and Scott 1986), completes the mapping of formulae to the objects, and (equivalence classes of) proofs to the arrows, of a Cartesian closed category. However, we need to extend this work; we wish to make the idea of aggregation primitive to the logic. In this case, we wish arrows of the "category of arguments" $\mathcal{A}$, what we will call an *evidential closed category*, to correspond to sets of arguments between two propositions. Then we define an aggregation operation in terms of a least upper bound, or *supremum*, which says that the aggregation $f \vee g$ contains at least as much information as either of the two sets of arguments on their own. That is, there is an order imposed on the arrows of $\mathcal{A}$ such that $f \leq f \vee g$ and $g \leq f \vee g$. For any two propositions A and B there is also a least arrow '0' corresponding to the vacuous argument from A to B. And so the arrows from A to B form a join semi-lattice $\langle \mathcal{A}(\text{A,B}), \vee, 0 \rangle$.

In addition, we would expect the following distributive laws to hold for the arrows shown in figure 2:

$$h(f \vee g) = hf \vee hg \qquad\qquad (f \vee g)k = fk \vee gk$$

$$\hbar 0 = 0 \qquad\qquad 0 \wr = 0$$

That is, $\mathcal{A}$ should be a semi-lattice enriched category.

A measure of the *strength* of an argument may then be defined which maps arguments to one of a number of possible 'dictionaries'. We may model a variety of qualitative, semi-qualitative, and numerical uncertainty calculi. For example, if f is the proof term corresponding to an argument from A to B, we may have:

$$s_{A,B}(f) \in \begin{cases} \{0, +, ++\} \\ \aleph^{\infty} \\ [0, 1] \end{cases}$$

In the first case, we have confirmation or proof (++), support (+), or the vacuous argument concerning a proposition (0). This results in a very simple theory preference model. The second case, the natural numbers augmented with infinity, corresponds to a slight refinement of the improper linear model with uniform weighting (Dawes 1979); basically the counting up of arguments. There are several approaches which may be taken to defining a mapping to the unit interval which we can outline here. A rigorous discussion of them can be found in (Ambler 1992).

A first step in the definition of an argument strength mapping is to define a *confidence measure* on arguments. The ordering of the arrows in the category of arguments $\mathcal{A}$ can be thought of as a "content ordering". We would expect that the ordering on the confidence values would respect the content ordering of the arguments themselves. Thus if $\mathcal{A}(A,B)$ denotes the semi-lattice of arguments from A to B, a confidence measure c gives rise to a family $c_{A,B}$: $\mathcal{A}(A,B) \to \mathcal{M}$ of semi-lattice homomorphisms into some suitable dictionary $\mathcal{M}$. Let **SLat** denote the category of (join) semi-lattices and semi-lattice homomorphisms between them. We require that $\langle \mathcal{M}, \bullet, \top \rangle$ be a commutative monoid in **SLat** such that the identity $\top$ is also a top element. Then:

Definition 5.1

A measure of confidence on $\mathcal{A}$ (valued in $\mathcal{M}$) is a family $c_{A,B}$: $\mathcal{A}(A,B) \to \mathcal{M}$ of maps in **SLat** such that:

(i) $c(\zeta) = \top$ for all purely logical ζ.

(ii) $c(f) \bullet c(g) \leq c(fg)$ for all composable f and g.

(iii) $c(f) = c(g)$ whenever g is obtained by a purely logical rearrangement of the argument f (e.g. by currying).

In the first condition, we have introduced the notion of a logical map. In general, an argument for a proposition will be constructed using assumptions (axioms) from a specific context. Any uncertainty associated with these assumptions will be propagated to the argument. However, if an argument contains no assumptions (a closed λ-term) then it will be unaffected by such uncertainty, and we may regard it as a logical proof. This is what we mean by a logical map, and we would intuitively expect that such arguments should be mapped to the top element of $\mathcal{M}$.

The second condition is in effect saying that the composition of the confidences in two arguments should be less than the confidence in the composition of those arguments. The inequality results because the composition of two arguments may lead to the elimination of now redundant axioms through proof normalisation.

The final condition is explained more fully in (Ambler 1992). It essentially covers transformations in the structure of an argument, which we would not expect to diminish the force of that argument

(and also do not involve the introduction or elimination of assumptions). For example, if f is an arrow corresponding to the entailment relation A & X $\vdash$ B, then it should have the same confidence as that of the arrow Λf which corresponds to the entailment relation A $\vdash$ X $\supset$ B (a transformation known as "currying").

Concrete examples of suitable candidates for $\langle \mathcal{M}, \bullet, \mathsf{T} \rangle$ are:

$\langle$[0,1], min, 1$\rangle$

$\langle$[0,1], $*$, 1$\rangle$

"a boolean algebra of propositions".

The first of these corresponds to an evaluation of arguments in terms of their 'weakest link'. For a given argument, we take the minimum of the confidences assigned to the axioms used in constructing the argument to give the overall confidence in that argument. As further arguments concerning a proposition are identified, we aggregate them by taking the maximum of the confidences in the individual arguments.

The second corresponds to an evaluation of arguments by a "shortest path" type of approach. The more axioms that are used in constructing an argument, the more the confidence in that argument will be attenuated by multiplying together the confidences assigned to the axioms. Again, as further arguments concerning a proposition are identified, we aggregate them by taking the maximum of the confidences in the individual arguments.

The final example can be thought of as a preliminary stage in a "probabilistic" appraisal of arguments. We can employ a stronger ordering on the relative confidences in arguments if we explicitly take into account the dependencies between arguments as we aggregate their confidence values. We have the following definition:

Definition 5.1

A *probability valuation* on a distributive lattice $\mathcal{D}$ is a monotone function p: $\mathcal{D} \rightarrow$ [0,1] s.t.

(i) p(0) = 0

(ii) p($\top$) = 1

(iii) p(x $\vee$ y) = p(x) + p(y) - p(x $\wedge$ y)

If we now compose a confidence measure valued in $\mathcal{D}$ with a probability valuation p we will obtain an "argument strength" mapping s valued in [0,1]. If we consider all possible arguments for a proposition A and take the supremum of their strengths, then we obtain a "probability of provability" for A. As noted in (Wilson 1989), this gives a generalisation of Dempster-Shafer belief to non-classical logic.

5.3 An example of a probability valuation.

A simple example will illustrate the form of the dependency analysis which is present in the probability valuation. Suppose we have the following axioms in a database, with associated subjective confidences:

a1: A	(0.80)
a2: B	(0.80)
a3: C	(0.80)
r1: A & B $\supset$ D	(1.00)

r2: A & C ⊃ D (1.00)

Two arguments for D can be constructed from this database:

apply(r1, pair(a1, a2)): D and

apply(r2, pair(a1, a3)): D.

These arguments are first mapped to the sets of assumptions used to construct them:

apply(r1, pair(a1, a2)) → x = {*r1, a1, a2*}

apply(r2, pair(a1, a3)) → y = {*r2, a1, a3*}

In this case, x ∧ y = {*r1, r2, a1, a2, a3*}.

The probability p(x) is obtained by multiplying together the confidences of all the assumptions in x. So we have:

p(x ∨ y) = p({*r1, a1, a2*}) + p({*r2, a1, a3*}) - p({*r1, r2, a1, a2, a3*})
= 0.64 + 0.64 - 0.542
= 0.77 (to 2 decimal places)

Contrast this with the following database in which distinct sets of assumptions can be used to construct the arguments for D:

a1: A (0.80)
a2: B (0.80)
a3: C (0.80)
a4: E (0.80)
r1: A & B ⊃ D (1.00)
r2: E & C ⊃ D (1.00)

Now we have:

p(x ∨ y) = p({*r1, a1, a2*}) + p({*r2, a4, a3*}) - p({*r1, r2, a1, a2, a3, a4*})
= 0.64 + 0.64 - 0.4096
= 0.87 (to 2 decimal places)

The second case gives a significantly higher probability valuation for D. Note that this is, however, a purely syntactic form of dependency analysis.

5.4 Discussion of confidence measures.

Suppose we were doubtful for some reason about the independence of the various arguments, or not willing to entail the computational overhead of the more complex "probability valuation". Then the confidences of the arguments concerning each of the propositions of interest should be aggregated using the simple *max* operator.

However, if we are sure that the arguments are based upon semantically independent information, we would expect that additional arguments could *reinforce* our confidence over and above that of the best one. The probability valuation allows this to be achieved in a coherent way by taking into account common variables in arguments as they are aggregated (although this is purely a syntactic notion of independence).

The following points should be noted:

- These are all to be considered as "pure" arguments. That is, they only argue for their relevant claim. They are completely agnostic about any alternative claim.

- In the motivating domain of application (medicine) we do not *a priori* know what all the relevant options are, which beliefs are alternatives and which beliefs may be held concurrently. For example, patients can, often do, suffer from more than one pathology simultaneously.

It is primarily for these reasons that we wish to weaken the impact of contradiction in the logic, and to avoid the use of a normalisation coefficient in the aggregation of arguments. We wish to see which propositions, or claims, are a logical consequence of a state of knowledge (context) and allow some measure of conviction to be conferred on those claims by (purely subjective) measures of confidence in the arguments which support them.

The bottom line is that we are defining a proof theoretic model of reasoning under uncertainty in which subjective estimates of confidence are assigned to the proofs of LA in a coherent way.

6. Symbolic aggregation of arguments: restoring consistency

6.1 Overview of "symbolic aggregation".

One of the motivations behind labelling propositions with the arguments which support their validity is that we wish to develop purely symbolic methods of reasoning *about* arguments. This is a long standing interest of the authors (Fox *et al.* 1988; Fox *et al.* 1992), that linguistic confidence terms can be defined on purely logical grounds over sets of arguments. In this section we will discuss the assignment of propositions to one of a number of "acceptability" classes, based on a purely symbolic analysis of their supporting and opposing arguments. This introduces notions of defeat and rebuttal of arguments into our framework. The next sub-section contains a discussion of the use of linguistic terms. The acceptability classes are then introduced in sub-section 6.3. This will enable the definition of a purely symbolic measure of confidence which is consistent with the framework established in section 5.

6.2 The use of linguistic terms.

There is a substantial body of literature on assigning a meaning to linguistic or modal qualifiers such as "it is possible that …", "it is plausible that …", "presumably …". The usual approach is to try and map the selected terms to fuzzy modifiers, (Zadeh 1975), or to probabilistic intervals (Dubois *et al.* 1992). However, we believe that there is an alternative non-numerical approach which may be taken to defining such terms. In (Fox 1984) it was suggested that such qualifiers may be interpreted in terms of patterns of argument. In the following we will take the notion of provability as primitive. Then, based on purely logical properties of the arguments which support them, we will show how propositions may be assigned to one of the classes {*certain, confirmed, probable, plausible, supported, open*}. Complementary classes, such as *opposed, doubted*, may be defined in terms of the assignment of negated terms to the earlier classes (Elvang-Gøransson *et al.* 1993).

We should justify the specific terms we have chosen to use. Firstly, let us emphasise that we will regard *every* axiom in the database as a contingent fact. Any of the axioms in the context may be retracted or contradicted. They are current beliefs, or at least working hypotheses, but they should be regarded as knowledge rather than certainty. Consequently, no proposition which depends on any of the axioms in the context should be regarded as certain. However, we do have faith in the logic we are using, and hence regard any tautologies of the logic as certain. That is, a proposition is *certain* if it is derivable without using any of the axioms in the context; it is supported by a *logical argument*, in the notation of LA. For example, we can classify $((a \,\&\, b) \supset c) \supset (a \supset (b \supset c))$ as

certain as it is supported by the logical argument $\lambda z.\lambda y.\lambda x.apply(z, pair(y,x))$; this argument contains no axiom labels.

From *certain* through to *open*, the terms are intended to represent gradually decreasing confidence in the propositions which are members of the respective classes. In the case of a *confirmed* proposition, whilst we do not have the confidence associated with a tautology, we do have at least one "good" argument (in a sense to be made precise later) for it, and no consistent arguments against it. Essentially, a proposition is confirmed if no consistent case can be made against it, or the argument which supports it. It turns out that confirmed propositions correspond to the *innocent bystanders* of (Rescher and Manor 1970). That is, they can be viewed as having no role to play in the derivation of contradiction from the database.

With respect to the word "probable", consider its use in the context of epistemic probability; or perhaps in terms of the intuitive description used by Toulmin in his discussion of argumentation (Toulmin 1956):

> *"When I say 'S is probably P', I commit myself guardedly, tentatively or with reservations to the view that S is P, and (likewise guardedly) lend my authority to that view."*

We will not have grounds to directly contradict a probable proposition, although we may have some grounds for doubting one or more of the steps used in constructing the argument which supports it.

We use the word "plausible" in a fairly weak sense. That is, a proposition "sounds plausible" because we can construct a consistent argument supporting it. However, in the presence of contradictory information, we may equally well be able to construct a consistent argument for its contrary. Weaker still is the notion of *support*. A proposition is supported if we can construct any argument, even an inconsistent argument, for it.

The term "open" is intended to confer a notion of having no information at all, of being completely agnostic, about the certainty of a proposition. Membership of the class *open* includes all well-formed-formulae in the language $\mathcal{L}$ of the logic that are not classifiable by any of the above terms.

6.3 Logical Uncertainty?

Very broadly, there are two ways in which we may challenge the claim of an argument. We may attempt to defeat the argument by attempting to rebut the claim of the argument; that is, by directly contradicting the claim. Alternatively, we may attempt to defeat the argument by undercutting it, by challenging some or all of the information used to construct the argument.

There has been a steady interest in developing models for reasoning in the presence of inconsistent data in both the AI (Dubois, Lang and Prade 1992; Fox *et al.* 1992; Pinkas and Loui 1992; Benferhat, *et al.* 1993a; 1993b) and philosophical logic (Nelson 1949; Priest 1987; Priest *et al.* 1988) communities. From a more pragmatic perspective, the sheer impracticality of ensuring consistency in large scale deductive databases has also led the logic programming community to consider consequence relations which are tolerant of varying levels of inconsistency (Wagner 1991). In this section we will describe a system which actually encourages the use of inconsistency to provide a logical model of a form of dialectical argumentation.

Given an inconsistent knowledge-base Δ (such as $\{a1{:}p, a2{:}\neg p, r1{:}p \supset q\}$) we may select consistent sub-bases ($\{a2{:}\neg p\}$ or $\{a1{:}p, r1{:}p \supset q\}$) from which meaningful conclusions may be drawn in

the context of those sub-bases. So, with the database Δ above, δ_1 = *apply(r1, a1)* is an argument *from* Δ *supporting* q, whilst δ_2 = *a2* is an argument from Δ supporting $\neg$p.

Two specific forms of arguments can now be distinguished:

Definition 6.1
An argument δ for p is a *consistent argument* if the assumptions used in δ are consistent.

In the above example, both δ_1 and δ_2 are consistent arguments since the assumptions used in their construction ({a1:p, r1:p $\supset$ q} and {a2:$\neg$p} respectively) form consistent sub-bases of Δ.

Definition 6.2
An argument δ for p is a *logical argument* if δ contains no assumptions.

Definition 6.2 merely says that the proposition p must be a tautology of the logic. From condition (i) of Definition 5.1 it can be seen that a coherent confidence measure must map logical arguments to the top element of the dictionary of terms.

Now, we come back to the two ways in which we may attempt to defeat an argument γ for q. Firstly, δ:p *rebuts* γ:q if and only if p is a direct contradiction of q (p $\dashv\vdash \neg$q). Alternatively, δ:p *undercuts* γ:q if and only if, for some assumption r in γ, p is a direct contradiction of r (i.e. p $\dashv\vdash \neg$r).

Once more using Δ = {a1:p, a2:$\neg$p, r1:p $\supset$ q}, we see that the argument *a2* for $\neg$p rebuts the argument *a1* for p, whilst the argument *a2* for $\neg$p undercuts the argument *apply(r1,a1)* for q (it directly contradicts a1:p).

"Acceptability" classes may then be defined using these two notions of defeat. These classes can be arranged into a hierarchy in which arguments have to pay a successively higher price of membership to be accepted into successive classes. The basic class allows all possible arguments as members. The second class only allows consistent arguments as members. The third class only allows those arguments which are not open to rebuttal as members, whilst the fourth class allows only those arguments which are not open to rebuttal or defeat. The final class consists of the logical arguments.

Let Δ be any set of propositions defined over the language $\mathcal{L}$. Then the following classes reflect increasing degrees of acceptability:

$A_1(\Delta) = \{\delta{:}p \mid \delta \text{ is any argument from } \Delta \text{ for } p\}$
$A_2(\Delta) = \{\delta{:}p \in A_1(\Delta) \mid \delta \text{ is a consistent argument from } \Delta \text{ for } p\}$
$A_3(\Delta) = \{\delta{:}p \in A_2(\Delta) \mid \neg(\exists\delta')\ (\delta'{:}\neg p) \in A_2(\Delta))\}$
$A_4(\Delta) = \{\delta{:}p \in A_3(\Delta) \mid (\forall \text{ assumptions q in } \delta)\ (\neg(\exists\delta')(\delta'{:}\neg q \in A_2(\Delta)))\}$
$A_5(\Delta) = \{\delta{:}p \in A_4(\Delta) \mid \delta \text{ is a logical argument from } \Delta \text{ for } p\}$

It can be seen that in the case of Δ = {a1:p, a2:$\neg$p, r1:p $\supset$ q}, *apply(r1,a1)*:q $\in A_3(\Delta)$ since it cannot be rebutted. However, it is *not* the case that *apply(r1,a1)*:q $\in A_4(\Delta)$ since this argument is open to undercutting defeat from the consistent argument *a2* for $\neg$p.

This hierarchy can now be used to assign linguistic "measures" of uncertainty to formulae. "Confidence classes" associated with a database Δ may be defined as:

$$C_{ce}(\Delta) = \{p \mid (\exists\delta)\ (\delta{:}p \in A_5(\Delta))\}$$

$C_{co}(\Delta) = \{p \mid (\exists \delta)\ (\delta{:}p \in A_4(\Delta))\}$
$C_{pr}(\Delta) = \{p \mid (\exists \delta)\ (\delta{:}p \in A_3(\Delta))\}$
$C_{pl}(\Delta) = \{p \mid (\exists \delta)\ (\delta{:}p \in A_2(\Delta))\}$
$C_{su}(\Delta) = \{p \mid (\exists \delta)\ (\delta{:}p \in A_1(\Delta))\}$
$C_{op}(\Delta) = \mathcal{L}$

The subscripts may be read as: *ce*rtain; *co*nfirmed; *pr*obable; *pl*ausible; *su*pported; *op*en. These are the terms introduced in section 6.2.

An ordering is induced over these classes by the property of the acceptability classes that for $i < j$, $A_i(\Delta) \supseteq A_j(\Delta)$ for any Δ. We have that for any Δ:

Proposition 6.1
$C_{ce}(\Delta) \subseteq C_{co}(\Delta) \subseteq C_{pr}(\Delta) \subseteq C_{pl}(\Delta) \subseteq C_{su}(\Delta) \subseteq C_{op}(\Delta)$.

We might note also that:

Proposition 6.2
For any consistent Δ: $C_{co}(\Delta) = C_{pr}(\Delta) = C_{pl}(\Delta) = C_{su}(\Delta)$.

From these properties we can see that the acceptability hierarchy can "run" in two different modes. In the case of a consistent database, the terms *co, pr, pl* and *su* become equivalent in status, and we can only discriminate between formulae which are open, i.e. have no support, formulae which are supported by contingent data, and tautologies. In the case of an inconsistent database, however, the collapse of *co, pr, pl* and *su* is broken.

This ordering on acceptability classes induces a candidate for a support function. The following function is defined:

Definition 6.3
Let $\Delta \subseteq \mathcal{L}$ be any set of propositions. Let g_Δ: $\mathcal{L} \rightarrow \{ce, co, pr, pl, su, op\}$ be defined by:

$$g_\Delta(p) = \max(\{x \mid p \in C_x(\Delta)\}).$$

Here, "max" is defined over $(\{ce, co, pr, pl, su, op\}, >)$ with *ce* as the maximal element and *op* as the minimal element.

Returning again to our example database, $\Delta = \{a1{:}p, a2{:}\neg p, r1{:}p \supset q\}$. It should be clear that $g_\Delta(p) = g_\Delta(\neg p) = pl$. That is, both p and $\neg$p are "plausible". Consistent arguments can be constructed which support them, but each argument is open to rebuttal by the other. However, $g_\Delta(q) = pr$; q is "probable". No argument can be constructed which rebuts the argument supporting q, as we have seen. Nevertheless, q cannot be confirmed as its supporting argument is open to defeat by a2: $\neg$p.

6.4 Critique.

We will look at some of the properties of this system of argumentation in this section.

The point of this system was to see how far we could get using simple notions of defeat, without using any other form of preference between arguments. In isolation, this does not enable us to

handle a notion of preference on the basis of specificity using the "dialectical argumentation" model. We can see why this is so by looking at the standard example database {*a1:penguin, r1:penguin* ⊃ ¬*fly, r2:penguin* ⊃ *bird, r3:bird* ⊃ *fly*}. In our scheme, we have *penguin* as confirmed. However, both *fly* and ¬*fly* are merely plausible; they directly rebut one another. Since we make no distinction between any of the steps used in constructing the arguments for *fly* and for ¬*fly*, we are unable to discriminate amongst them on purely syntactic grounds.

There are a number of alternatives here. We may make some sort of distinction between two classes of "primitive" arguments, as exemplified by {*r1:penguin* ⊃ ¬*fly, r2:penguin* ⊃ *bird*} and {*r3:bird* ⊃ *fly*}. This is the approach taken in default logic, for example, where the latter class corresponds to default, or defeasible, arguments. Alternatively, we might try to introduce some syntactic definition of specificity as exemplified in Poole (1985), although this is not without its problems (e.g. Prakken 1992). Note that this last also requires a non-logical delimitation of necessary facts, contingent facts and hypotheses.

One might also provide a further refinement of the designations of argument types with an associated strength ordering. Kimbrough and Hua (1991), for example, label arcs in proof trees ("steps in an argument") with terms corresponding to "presumably", "materially" (for material implication), "causally" and "logically". These labels are "swept up" as an argument is constructed in a rather similar way to the way in which proof terms are constructed in LA. Kimbrough and Hua then propose a number of decision rules for resolving dilemmas based on an assessment of the relative supports conferred by the labels.

An approach which does fit naturally within our framework is to introduce an epistemic entrenchment ordering on propositions (Gärdenfors 1988). Consistency is then restored by breaking the offending arguments at their weakest link. That is, the relative strength of arguments, obtained using a confidence measure of section 5.2, is used to guide the process of belief revision.

These approaches are all the subject of ongoing work aimed at enriching our model. However, we would emphasise that the purpose of this exercise was to see just how far we could go with the absolute minimum of information about the propositions of interest.

Further work is required to refine the support function $g_\Delta$into a confidence measure that is fully consistent with the framework described in section 5. However, to provide further insight into the properties of our system of dialectic argumentation we will look to see to what extent it satisfies the axioms of possibility theory (Dubois and Prade 1988a). Dubois and Prade (1988a) consider these to be the weakest axioms that a general confidence evaluation function should satisfy to ensure a minimum of coherence.

Since any class of propositions, $C_x(\Delta)$, enjoys the closure property, the following proposition immediately follows:

Proposition 6.3

Suppose p ⊃ q is a tautology. Then if q is derived from p and p ⊃ q using modus ponens, then $g_\Delta(p) \leq g_\Delta(q)$.

In addition we have the following properties:

Proposition 6.4

$g_\Delta(\top) = ce$ and $g_\Delta(\bot) = \begin{cases} op \text{ iff } \Delta \text{ is consistent} \\ su \text{ iff } \Delta \text{ is inconsistent} \end{cases}$

These properties suggest the following variant of the axioms of possibility theory.

(Ax1) $g_\Delta(\top) = ce$

(Ax2) $g_\Delta(p) \leq g_\Delta(q)$, if $\vdash p \supset q$

(Ax3) $g_\Delta(\bot) = \begin{cases} su \text{ iff } \Delta \text{ is inconsistent} \\ op \text{ iff } \Delta \text{ is consistent} \end{cases}$

Proposition 6.5

Assuming that we augment our logic with syntax and rules for disjunction:

- g_Δ satisfies Ax1-3 for any Δ.
- $\max(g_\Delta(p), g_\Delta(q)) \leq g_\Delta(p \vee q)$
- $g_\Delta(p \wedge q) \leq \min(g_\Delta(p), g_\Delta(q))$

The implication of Ax2 must be tautological in order to ensure that it is 'context' independent. Without this extra requirement, the assignment of propositions to acceptability classes would not satisfy Ax2. This is illustrated by the following example:

Example: Let $\Delta = \{p \wedge r, (p \supset q) \wedge \neg r\}$. Here p, $p \supset q$ are both probable (*pr*), but q is only supported (*su*). Thus $g_\Delta(p) > g_\Delta(q)$. This problem arises because we are trying to compare supports which are derived using mutually inconsistent (sub-)contexts. The solution in this case is to combine information into a single context before it is applied. In the example, where we want to apply the pieces of information $p \supset q$ and p derived from the sub-contexts $\{(p \supset q) \wedge \neg r\}$ and $\{p \wedge r\}$ respectively, this is done by forming their conjunction $p \wedge (p \supset q)$. Since $(p \wedge (p \supset q)) \supset q$ is a tautology, Ax2 applies to this case and we have $g_\Delta(p \wedge (p \supset q)) \leq g_\Delta(q)$ (both are equal to *su*).

The example illustrates that the degree of uncertainty assigned to a proposition is strongly dependent on context - reflecting the fact that logics of uncertainty are not truth functional (Dubois and Prade 1988b). The following proposition explains how information can be combined and applied across the context-boundaries.

Proposition 6.6

$g_\Delta(p \wedge (p \supset q)) \leq g_\Delta(q)$ for any Δ.

It is easy to see that in the case where $g_\Delta(p \supset q) = ce$, i.e. the implication is tautological, then this is equivalent to Ax2.

We would emphasise again that our aim has been to see how far we could get using a simple notion of argumentation based on sub-theories of an inconsistent database. Consequently, our notion of undercutting defeat is rather simplistic compared with that which is explored in the work of Pollock (1992), Nute (1988) and Loui (1987) for example. However, as an experiment in the classification of propositions purely on the basis of their supporting arguments, we think it has produced a very interesting and encouraging result.

7. Final summary and discussion

As we mentioned in the introduction, this paper should be viewed as a study in epistemic probability. The definition of the syntax and semantics of a logic of argumentation, LA, has been central to this work. This provides us with a uniform framework which supports a variety of numerical

and symbolic techniques for assessing the confidence in a proposition based on the nature of the arguments which support it.

In developing LA, the property of classical logic and intuitionistic logic in which support for the contradiction is propagated to all propositions was rejected. This was for several reasons. Primarily we wish to be free to choose how, or whether, contradiction should be resolved; the significance of a level of contradiction may differ in different problem solving contexts. Reasoning about and with contradiction is an active research topic (Fox *et al.* 1992; Gabbay and Hunter 1991). The work reported in section 6 provides one strategy for classifying propositions on the basis of the level of inconsistency in their supporting arguments. This is a topic of ongoing work which is beginning to establish a fruitful connection with much recent work in philosophical logic (see also Loui 1991; Pollock 1991; Rescher 1977).

However, there is a limitation to the extent to which we can currently develop this approach. The work reported in this paper is founded on the established relationship between proofs in *minimal* logic and deductive systems modelled as a certain class of categories. This gives us a relatively weak form of argumentation, although a form we are finding useful in the development of applications. In defence, we would emphasise that the application of category theory is still in its infancy and its use is controversial. Future work will be able to build on the foundation presented here.

The main message of this paper is that we promote arguments, or proofs, to be first class objects themselves. That in reasoning under uncertainty we need to reason about the arguments, not just the relevant propositions. The work described in this paper provides a firm foundation for that goal. But the challenge now is to develop a range of sound techniques for meta-level reasoning, to explore fully the possibilities which this approach opens up.

Acknowledgements

The authors would like to thank the late Mike Clarke of Queen Mary and Westfield College for many helpful discussions on this work. This paper is dedicated to his memory. Thanks must also go to Philippe Besnard for very helpful advice and comments. The first two authors are supported under the DTI/SERC project 1822: a Formal Basis for Decision Support Systems. This work has received an added stimulus with our involvement in and support from the Esprit Basic Research Programme 3085, DRUMS. We would also like to thank the reviewers for a great deal of work on their part which has helped us to tighten up the argument presented in this paper.

References

Ambler S. 1992. A Categorical Approach to the Semantics of Argumentation. Technical Report, Department of Computer Science, Queen Mary and Westfield College, London.

Benferhat S. Dubois D. and Prade H. 1993a. Argumentative inference in uncertain and inconsistent knowledge bases. IRIT Research Report, Université Paul Sabatier, France.

Benferhat S., Cayrol C., Dubois D., Lang J. and Prade H. 1993b. Inconsistency management and prioritized syntax-based entailment. *Proc. IJCAI '93,* Chambery, France.

de Bruijn N.G. 1972. Lambda Calculus Notation with Nameless Dummies, a Tool for Automatic Formula Manipulation, with Application to the Church-Rosser Theorem. Indagationes Mathematicae, 34, pp. 381-392.

Dawes R. M. 1979. The robust beauty of improper linear models in decision making. American Psychologist, 34, pp. 571-582.

Dubois D. and Prade H. 1988a. Possibility Theory: an Approach to Computerised Processing of Uncertainty, Plenum Press.

Dubois D. and Prade H. 1988b. An Introduction to Possibilistic and Fuzzy Logics. In: Smets P., Mamdani E.H., Dubois D. and Prade H., eds, Non-Standard Logics for Automated Reasoning, Academic Press.

Dubois D., Lang J. and Prade H. 1992. Inconsistency in possibilistic knowledge bases: To live with it or not live with it. In: Zadeh L. and Kacprzyk (eds.), Fuzzy Logic for the Management of Uncertainty, John Wiley, Chichester.

Dubois D., Prade H., Godo L. and Lopez de Màntaras R. 1992. A symbolic approach to reasoning with linguistic quantifiers. Proc. 8th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufman.

Dyckhoff R. 1992. Contraction-free Sequent Calculi for Intuitionistic Logic. To appear in Journal of Symbolic Logic.

Elvang-Gøransson M., Krause P.J. and Fox J. 1993. Dialectic reasoning with inconsistent information. In: Heckerman, D. and Mamdani, A. (eds.), Uncertainty in Artificial Intelligence. Proceedings of the Ninth Conference, Morgan Kaufmann, San Mateo, Ca.

Fox J. 1984. Language, Logic and Uncertainty. Technical Report, Imperial Cancer Research Fund, London.

Fox J., Glowinski A.J., O'Neil M.J. and Clark D.A. 1988. Decision Making as a logical process. *Proceedings of Expert Systems '88*, Cambridge University Press.

Fox J., Krause P.J. and Ambler S.J. 1992. Arguments, contradictions and practical reasoning. Proceedings of ECAI '92, John Wiley and Sons, pp. 623-627.

Gabbay D. 1990. Labelled Deductive Systems. CIS Technical Report 90-22, University of Munich.

Gabbay D. and Hunter A. 1991. Making Inconsistency Respectable: a logical framework for inconsistent reasoning. Proceedings of the International Workshop on Fundamentals of Artificial Intelligence Research, Bratislava.

Gallier J.H. 1987. Logic for Computer Science. John Wiley, Chichester.

Gärdenfors P. 1988. Knowledge in Flux. Modelling the Dynamics of Epistemic States. MIT Press, Cambridge, Ma.

Girard J., Taylor P. and Lafont Y. 1989. Proofs and Types. Cambridge University Press, Cambridge.

Heyting A. 1956. Intuitionism. An Introduction. North-Holland Publishing Company, Amsterdam.

Hindley J.R. and Seldin J.P. 1986. Introduction to Combinators and λ-Calculus. Cambridge University Press, Cambridge.

Kelly G.M. 1982. Basic Concepts of Enriched Category Theory. London Mathematical Society Lecture Note Series 64. Cambridge University Press.

Kimbrough S.O. and Hua H. On Nonmonotonic Reasoning with the Method of Sweeping Presumptions. Minds and Machines, 1, pp. 393-416.

Krause P.J. Ambler S.J. and Fox J. 1993. The Development of a "Logic of Argumentation". In: Bouchon-Meunier, B., Valverde, L. and Yager, R. (eds.), Advanced Methods in Artificial Intelligence, Springer-Verlag.

Lambek J. and Scott P.J. 1986. Introduction to higher order categorical logic. Cambridge University Press.

Loui R.P. 1987. Defeat among arguments: a system of defeasible inference. Computational Intelligence, 3, pp. 100-106.

Loui R.P. 1991. Argument and Belief: Where We Stand in the Keynsian Tradition. Minds and Machines, 1, 357-365.

Nelson D. 1949. Constructible falsity. Journal of Symbolic Logic, 14, pp. 16-26.

Nute D. 1988. Defeasible Reasoning and Decision Support Systems. Decision Support Systems, 4, pp. 97-110.

Pinkas G. and Loui R.P. 1992. Reasoning from Inconsistency: A Taxonomy of Principles for Resolving Conflict. Proc. of the 3rd Inter. Conf. on Principles of Knowledge Representation and Reasoning (KR '92), Cambridge, Mass, pp. 709-719.

Pollock J.L. 1991. Self-Defeating Arguments. Minds and Machines, 1, pp. 367-392.

Pollock J.L. 1992. How to reason defeasibly. Artificial Intelligence, 57, pp. 1-42.

Poole D. L. 1985. On the comparison of theories: preferring the most specific explanation. *Proc. IJCAI '85*, Los Angeles, USA, pp. 144-147.

Prakken H. 1993. An argumentation framework in default logic. Annals of Mathematics and Artificial Intelligence, (to appear).

Priest G. 1987. In Contradiction: A Study of the Transconsistent. Martinus Nijhoff Publishers.

Priest G., Routley R. and Norman J. (eds.) 1988. Paraconsistent Logics. Philosophia Verlag.

Rescher N. 1977. Dialectics: A Controversy Oriented Approach to the Theory of Knowledge. State University of New York Press.

Rescher N. and Manor R. 1970. On Inference from Inconsistent Premises. Theory and Decision, 1, pp. 179-217.

Shafer G. 1978. Non-additive Probabilities in the Work of Bernoulli and Lambert. Archive for History of Exact Sciences, 19, pp. 309-370.

Toulmin S. 1956. The uses of argument. Cambridge University Press.

Wagner G. 1991. Ex contradictione nihil sequitur. Proceedings of IJCAI 91, Morgan-Kaufman.

Wilson, P.N. 1989. Justification, computational efficiency and generalisation of the Dempster Shafer Theory. Research Report no. 15, Dept. of Computing and Math. Sciences, Oxford Polytechnic (also to appear in Artificial Intelligence).

Zadeh L.A. 1975. The Concept of a Linguistic Variable and its Application to Approximate Reasoning - III. Information Science, 9, pp. 43-80.