A Framework for Automated Reasoning in Multiple-Valued Logics *

James J. Lu
Department of Computer Science
Bucknell University
Lewisburg, PA 17838
lu@sol.cs.bucknell.edu
(717) 524-1162

Neil V. Murray
Department of Computer Science
State University of New York
Albany, NY 12222
nvm@cs.albany.edu
(518) 442-3393

Erik Rosenthal
Department of Mathematics
University of New Haven
West Haven, CT 06516
brodsky@charger.newhaven.edu
(203) 932-7463

Draft Date: March 16, 1995

Abstract

The language of signed formulas offers a first order classical logic framework for automated reasoning in multiple-valued logics. It is sufficiently general to include both annotated logics and fuzzy operator logics. Signed resolution unifies the two inference rules of annotated logics, thus enabling the development of an SLD-style proof procedure for annotated logic programs. Signed resolution also captures fuzzy resolution. The logic of signed formulas offers a means of adapting most classical inference techniques to most multiple-valued logics.

Keywords: *multiple-valued logics, signed formulas, annotated logics, fuzzy operator logics, inference*

*This research was supported in part by the National Science Foundation under grants CCR-9225037, CCR-9101208 and CCR-9202013.

1 Introduction

Non-standard logics have received considerable attention in recent years ([3, 4, 7, 8, 10, 11, 14, 15, 16, 17, 24, ?, 29, 30, 38], to cite but a few). One feature common to much of this work is the use of *signs* — subsets of the set of truth values. For example, an adaptation of standard tableau signs for multiple-valued logics (MVL's) was first proposed by Surma [40] in 1974 and by Suchón [39] that same year. In 1987 Carnielli [4] extended Surma's work to a wider class of logics. In 1990 Doherty [7] developed a tableau system for a variant of a three-valued logic developed by Sandewall [36]. The notion of a set of truth values as a sign is implicitly present in both Suchón's and Doherty's work. In 1991 the explicit and formal development of signs was introduced independently in [10, 11] by Hähnle and in [28]. Closely related notions were developed the following year in [2].

This approach often allows the utilization of classical techniques for analysis of a non-standard logic. This is appealing since, at the meta-level, human reasoning is essentially classical. That is, regardless of the domain of truth values associated with a logic, at the meta-level humans interpret statements about the logic to be either $true$ or $false$.

In this paper, we formalize the language of signed formulas and describe how it offers insight into the base multiple-valued logic to which it is related. The logic of the language of signed formulas is classical; it is related to the MVL by restricting attention to the Λ-*consistent* interpretations (see Section 2 for the definition). The insight into the MVL is provided by applying classical inference techniques restricted to Λ-consistent interpretations.

Two widely studied examples are considered in this paper: annotated logics and fuzzy operator logics. In Section 4 the annotated logics of Kifer and Lozinskii [15, 14], Kifer and Subrahmanian [16, 17], and da Costa, Henschen, Lu, and Subrahmanian [5, 24] are shown to be special cases of the language of signed formulas. The two inference rules of annotated logics — annotated resolution and reduction — are unified by *signed resolution*, which makes possible an SLD-style proof procedure for annotated logic programs. In Section 5 fuzzy operator logics are described, and the fuzzy resolution rule developed in [41] and [20] is also shown to be a special case of signed resolution. While only resolution is adapted to these logics, the results imply much more. The logic of signed formulas is a classical logic in which interpretations are restricted to the Λ-consistent ones. While this restriction does mean that classical inference techniques do not automatically apply, we know of no inference rule that is not easily adaptable. Thus, the results of this paper enable the application of a large collection of inference techniques to MVL's.

In the next section, the broad class of MVL's to be considered is defined. A formal basis is provided for the language of signed formulas, and the extension to the first order case is presented. In Section 3.3, the manner in which resolution can be adapted to signed formulas is described, and the completeness of the resulting signed resolution rule is proved.

2 MVL's and Signed Formulas

We assume a language Λ consisting of (finite) logical formulas built in the usual way from a set $\mathcal{A}$ of atoms, a set of connectives, and a set of logical constants. For the sake of completeness, we define a formula in Λ as follows: Atoms are formulas; if Θ is an n-ary connective and if $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n$ are formulas, then so is $\Theta(\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n)$.

Associated with Λ is a set Δ of truth values, and an interpretation for Λ is a function from $\mathcal{A}$ to Δ; i.e., an assignment of truth values to every atom in Λ. A connective Θ of arity n denotes a function $\Theta : \Delta^n \rightarrow \Delta$. Interpretations are extended in the usual way to mappings from the set of formulas to Δ. Alternatively, a formula $\mathcal{F}$ of Λ can be regarded as denoting a mapping from interpretations to Δ.

A *sign* is a subset of Δ, and a *signed formula* is an expression of the form $S:\mathcal{F}$, where S is a sign and $\mathcal{F}$ is a formula in Λ. If $\mathcal{F}$ is an atom in Λ, we call $S:\mathcal{F}$ a *signed atom*.

Signed formulas may be thought of as a formalization of meta-reasoning over multiple-valued logics [32]. A natural interpretation of the signed formula $S:\mathcal{F}$ is the query, "Is the truth value of $\mathcal{F}$ in S?" The answer to such a query is yes or no; that is, either the formula evaluates to S or does not. Observe that both the query and the answer are at the meta-level; observe also that the question cannot even be formulated at the object level. On the other hand, the question, "What is the the truth value of $\mathcal{F}$?" may be interpreted at the object-level since the answer is an element of Δ.

For example, let Δ be the interval [0, 1], where elements of Δ represent the degree of belief of some fixed reasoning agent X. Thus, $1{:}P$ can be interpreted as, "Is X certain of the proposition P?" and $[0,.1]{:}P$ asks, "Is X quite doubtful of P?" These are yes or no question.

To answer arbitrary queries, we map formulas in Λ to formulas in a classical propositional logic Λ_S, *the language of signed formulas*; it is defined as follows: The atoms are signed formulas and the connectives are (classical) conjunction and disjunction. It should be emphasized that a signed formula $S:\mathcal{F}$ is an atom in Λ_S regardless of the size or complexity of $\mathcal{F}$ and thus has no component parts in the language Λ_S. The set of truth values is $\{true, false\}$.

An arbitrary interpretation for Λ_S may make an assignment of $true$ or $false$ to any signed formula (i.e., to any atom) in the usual way. Our goal is to focus attention only on those interpretations that relate to the sign in a signed formula. To accomplish this we restrict attention to Λ-*consistent interpretations*. An interpretation I over Λ assigns to each atom, and therefore to each formula $\mathcal{F}$, a truth value in Δ, and the corresponding Λ-consistent interpretation I_c is defined by $I_c(S{:}\mathcal{F}) =$ *true* if $I(\mathcal{F}) \in S$; $I_c(S{:}\mathcal{F}) =$ *false* if $I(\mathcal{F}) \notin S$. Note that this correspondence between the set of all interpretations over Λ and the set of Λ-consistent interpretations over Λ_S is 1–1. Intuitively, Λ-consistent means an assignment of *true* to all signed formulas whose signs are simultaneously achievable via some interpretation over the original language. Restricting attention to Λ-consistent interpretations yields a new consequence relation: If $\mathcal{F}_1$ and $\mathcal{F}_2$ are formulas in Λ_S, we write $\mathcal{F}_1 \models_\Lambda \mathcal{F}_2$ if whenever I_c is a Λ-consistent interpretation and $I_c(\mathcal{F}_1) =$ *true*, then $I_c(\mathcal{F}_2) = true$. Two formulas $\mathcal{F}_1$ and $\mathcal{F}_2$ in Λ_S are Λ-*equivalent* if $I_c(\mathcal{F}_1) = I_c(\mathcal{F}_2)$ for any Λ-consistent interpretation I_c; we write $\mathcal{F}_1 \equiv_\Lambda \mathcal{F}_2$. The following lemma is immediate.

Lemma 1. Let I_c be a Λ-consistent interpretation, let A be an atom and $\mathcal{F}$ a formula in Λ, and let S_1 and S_2 be signs. Then:

1. $I_c(\emptyset{:}\mathcal{F}) = false$;
2. $I_c(\Delta{:}\mathcal{F}) = true$;
3. $S_1 \subseteq S_2$ if and only if $S_1{:}\mathcal{F} \models_\Lambda S_2{:}\mathcal{F}$ for all formulas $\mathcal{F}$;
4. There is exactly one $\delta \in \Delta$ such that $I_c(\{\delta\}{:}A) = true$. □

3 The First Order Case

It is straightforward to introduce variable, constant, and function symbols into Λ with their usual meaning. Introducing quantifiers is less straightforward. One way is *distribution quantifiers* [**?**], which are defined in Section 3.1. Since our goal is to use Λ_S to analyze first order formulas in

Λ, some means of translating quantifiers in Λ to classical quantifiers is desirable. While there are special cases where such a translation is fairly straightforward — for example, Theorem 2 below — we know of no general technique for doing so. If the formula in Λ is closed, then it denotes a single truth value in Δ, so the corresponding formula in Λ_S must denote either $true$ or $false$. If there are free variables, then queries such as, "Can the formula evaluate to the sign S?" or "Must the formula evaluate to S?" make sense, and the corresponding formula in Λ_S will be existentially quantified in the former case and universally quantified in the latter.

We begin this section with first order issues in Λ_S and conclude it with a discussion of quantifers in Λ.

3.1 First Order Signed Formulas

We use the notation $\mathcal{F}[x_1, x_2, ..., x_n]$ to indicate that $\mathcal{F}$ is a formula in Λ with n free variables. If I is an interpretation for Λ, and if $\mathcal{D}$ is the corresponding domain of discourse, then I maps $\mathcal{F}$ to a function from $\mathcal{D}^n$ to Δ. A *valuation* is an assignment of variables to elements of $\mathcal{D}$. Under interpretation I and valuation V, a formula $\mathcal{F}$ yields a truth value from Δ, which we denote $I^V(\mathcal{F})$. When the valuation V is irrelevant (for example, if $\mathcal{F}$ is ground or if $\mathcal{F}$ is closed), we simply write $I(\mathcal{F})$. A valuation V' is an *x-variant* of valuation V if V and V' assign the same value to all variables except possibly to x.

Given a (first order) formula $\mathcal{F}$ in Λ, if S is a sign, then the expression $S\!:\!\mathcal{F}$ denotes a *signed predicate* in Λ_S. Defining the arity of $S : \mathcal{F}$ requires care. If every variable in $\mathcal{F}$ is bound, then, for any interpretation, $\mathcal{F}$ denotes a single element of Δ, regardless of the valuation. In that case, $S\!:\!\mathcal{F}$ is essentially propositional and has arity zero. More generally, $S\!:\!\mathcal{F}$ "needs" arguments only when there are arguments of $\mathcal{F}$ that contain free variables. Thus, the arity of $S\!:\!\mathcal{F}$ is the number of arguments of $\mathcal{F}$ that contain at least one free variable.

Variables and function symbols in $\mathcal{F}$ relate to the domain of discourse defined by an interpretation and are independent of the set of truth values associated with Λ. Quantifiers relate variables to Δ, so the meaning of function symbols and unbound variables is unchanged in Λ_S. In particular, given an interpretation and corresponding domain of discourse $\mathcal{D}$, if $x_1, x_2, ..., x_n$ are the variables with at least one unbound occurrence in $\mathcal{F}$, then $S\!:\!\mathcal{F}$ is a function from $\mathcal{D}^n$ to $true$, $false$ defined as follows: If $a_1, a_2, ..., a_n$ are in $\mathcal{D}$, then $S : \mathcal{F}[a_1, a_2, ..., a_n]$ evaluates to $true$ if $\mathcal{F}$ evaluates to S when each a_i is substituted for every free occurrence of x_i in $\mathcal{F}$.

The notion of Λ-consistency can be extended to first order formulas as follows: Let Δ be the domain of truth values for Λ, let I be an arbitrary interpretation with domain of discourse $\mathcal{D}$, and let V be a valuation. Then the corresponding Λ-consistent interpretation (with valuation V) is denoted $I_{\Lambda_S}^V$ and is defined by:

$I_{\Lambda_S}^V(t) = I^V(t)$ for any term t;

$I_{\Lambda_S}^V(S\!:\!\mathcal{F}) = true$ if $I^V(\mathcal{F}) \in S$, $false$ otherwise;

$I_{\Lambda_S}^V(\forall x(S\!:\!\mathcal{F})) = true$ if for all x-variants V' of V, $I^{V'}(\mathcal{F}) \in S$, $false$ otherwise;

$I_{\Lambda_S}^V(\exists x(S\!:\!\mathcal{F})) = true$ if for some x-variant V' of V, $I^{V'}(\mathcal{F}) \in S$, $false$ otherwise.

Note that the correspondence between interpretations over Λ and interpretations over Λ_S is 1–1 as before.

The usual quantifier rules for classical logic apply to formulas in Λ_S, and formulas can be put into prenex normal form in the standard manner. Thus, if $\mathcal{Q}$ is $\exists$ or $\forall$, if $\mathcal{F}$ is any formula in Λ_S, and if $\mathcal{F}[a/b]$ denotes the result of replacing all occurrences of a in $\mathcal{F}$ by b, then we have:

- $\mathcal{Q}y(\mathcal{F}[x/y]) = \mathcal{Q}x\mathcal{F}$, where y does not occur in $\mathcal{F}$;
- $\forall x(\mathcal{F}_1 \bigwedge \mathcal{F}_2) = \forall x\mathcal{F}_1 \bigwedge \forall x\mathcal{F}_2$;
- $\exists x(\mathcal{F}_1 \bigvee \mathcal{F}_2) = \exists x\mathcal{F}_1 \bigvee \exists x\mathcal{F}_2$.

Skolemization can be done as usual. As in standard logic, formulas need not be in prenex normal form, but, for simplicity, the lemma addresses only that case.

Lemma 2. Let $(\mathcal{Q}x_1, \mathcal{Q}x_2, \ldots, \mathcal{Q}x_n(\exists x_0\mathcal{F}))$ be a closed formula in Λ_S, let $y_1, y_2, \ldots, y_m$ be the universally quantified variables in $\{x_1, x_2, \ldots, x_n\}$, and let f be an m-ary function symbol not occurring in $\mathcal{F}$. Suppose that I_{Λ_S} is a Λ-consistent interpretation and that $I_{\Lambda_S}(\mathcal{Q}x_1, \mathcal{Q}x_2, \ldots, \mathcal{Q}x_n(\exists x_0\mathcal{F}))$ $= true$. Then $I'_{\Lambda_S}(\mathcal{Q}x_1, \mathcal{Q}x_2, \ldots, \mathcal{Q}x_n\mathcal{F}[x_0/f(y_1, y_2, \ldots, y_m)]) = true$ for some Λ-consistent interpretation I'_{Λ_S}. □

We define the non-empty Herbrand universe $\mathcal{D}_h$ of a formula $\mathcal{F}$ in the standard way. Let $h_1, h_2, \ldots, h_n$ be a sequence of Herbrand terms (elements of $\mathcal{D}_h$), and let $d_1, d_2, \ldots, d_n$ be a sequence of elements of $\mathcal{D}$. These sequences determine valuations V_h (over $\mathcal{D}_h$) and V (over $\mathcal{D}$), and hence the valuation pair $\langle V_h, V\rangle$ for $x_1, x_2, \ldots, x_n$. We may think of a given valuation pair as a bijection between variable assignments over $\mathcal{D}_h$ and variable assignments over $\mathcal{D}$.

Given formula $\mathcal{F}$ in Λ_S, interpretation I, and valuation pair $\langle V_h, V\rangle$, we construct a corresponding Herbrand interpretation I_h as follows:

- $I_h(t) = t$, for any ground term t;
- $I_h(P(t_1, t_2, \ldots, t_n)) = I(P)(I(t_1), I(t_2), \ldots, I(t_n))$, where $P(t_1, \ldots, t_n)$ is a ground atom;
- $I_h^{V_h}(\mathcal{F}) = I^V(\mathcal{F})$, where $\mathcal{F}$ contains free variables $x_1, x_2, \ldots, x_n$;
- $I_h(\forall x_1, x_2, \ldots, x_n\mathcal{F})$ is $true$ if, for all valuation pairs $\langle V'_h, V'\rangle$, $I_h^{V'_h}(\mathcal{F})$ (and hence $I^{V'}(\mathcal{F})$) is $true$, where $\forall x_1, x_2, \ldots, x_n\mathcal{F}$ is closed.

Suppose $I_{\Lambda_S}(\forall x_1, x_2, \ldots, x_n(\mathcal{F})) = true$; then for all valuations V over $\mathcal{D}$, $I_{\Lambda_S}^V(\mathcal{F}) = true$. The corresponding Herbrand interpretation $I_h^{V_h}$ maps $\mathcal{F}$ to $true$ for any valuation V_h over $\mathcal{D}_h$. Hence $I_h(\forall x_1, x_2, \ldots, x_n(\mathcal{F})) = true$.

Notice that Λ-consistency is not assumed either in the quantifier rules or in the construction of Herbrand interpretations that correspond to arbitrary interpretations; these are independent of Λ-consistency. In particular, the construction of I_h will preserve the presence or absence of this property. This proves:

Lemma 3. Let $\mathcal{F}$ be a quantifier-free formula of Λ_S in which all variables are implicitly universally quantified. Let I be a Λ-consistent interpretation that satisfies $\mathcal{F}$. Then any corresponding Herbrand interpretation I_h satisfies $\mathcal{F}$ and is Λ-consistent. □

Corollary. Formula $\mathcal{F}$ is unsatisfiable with respect to Λ-consistent interpretations if and only if it is unsatisfiable with respect to Λ-consistent Herbrand interpretations. □

Since unsatisfiability with respect to arbitrary Λ-consistent interpretations can be determined by considering only Herbrand interpretations, our variation on Herbrand's Theorem can be established using semantic trees. In [30] and [2], n-ary semantic tree constructions are presented in which the branching factor n corresponds to the cardinality of Δ. Here, Δ may be infinite; however, the finiteness of our formulas enables the adaptation of n-ary semantic tree arguments so as to produce finitary trees. Nevertheless, we feel that the *binary* semantic tree construction below is worth presenting because of its direct relationship to Λ_S.

We define the Herbrand base of $\mathcal{F}$ to be the set $\mathcal{A} = \{A_1, A_2, \ldots\}$ of ground instances of the (unsigned) atoms of $\mathcal{F}$. Let $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$ be the set consisting of all signs appearing in $\mathcal{F}$. The *signed Herbrand base* of $\mathcal{F}$ consists of all ground signed atoms of the form $S_j : A_i$, where $S_j \in \mathcal{S}$ and $A_i \in \mathcal{A}$.

Let $\mathcal{H} = \{S_{k_1} : A_1, S_{k_2} : A_2, \ldots\}$ be an enumeration of the signed Herbrand base of $\mathcal{F}$, where, for $j \geq 1$, $1 \leq k_j \leq n$. A semantic tree for $\mathcal{H}$ is an ordered binary tree that is labeled as follows. The root at level 0 is unlabeled, and each node at level i has two children labeled $S_{k_{i+1}} : A_{i+1}$ and $(S_{k_{i+1}})' : A_{i+1}$, where, for any sign S, $S' = \Delta - S$.

Consider the path from the root to an arbitrary node $S_{k_m}:A_m$ in a semantic tree. To this path there corresponds a truth value assignment, denoted I_{A_m}, defined on the atoms $S_{k_1} : A_1$ through $S_{k_m} : A_m$. The assignment I_{A_m} is a partial interpretation that assigns *true* to each atom on the path; it need not be Λ-consistent (i.e., I_{A_m} may not be extendible to a Λ-consistent interpretation). Determining whether I_{A_m} is Λ-consistent is straightforward: It is Λ-consistent if and only if for each unsigned atom A_k occurring on the path from the root to $S_{k_m} : A_m$, the intersection of the signs that occur in the labels containing A_k is not empty.

Let $\mathcal{F}$ be a formula in Λ_S and $\mathcal{H}$ an enumeration of its signed Herbrand base. Given a semantic tree for $\mathcal{H}$, the node $S : A$ is a *failure node* if the assignment I_A is *not* Λ-consistent or if it falsifies some ground instance of $\mathcal{F}$. A semantic tree is *closed* if each path terminates in a failure node.

Theorem 1. (Herbrand's Theorem) The formula $\mathcal{F}$ in Λ_S is unsatisfiable with respect to Λ-consistent interpretations if and only if there is a finite set of ground instances of $\mathcal{F}$ that is similarly unsatisfiable.

Proof. Suppose $\mathcal{F}$ is unsatisfiable. Then, by Lemma 3, $\mathcal{F}$ has no (Λ-consistent) Herbrand model. Let $\mathcal{H}$ be an enumeration of the Herbrand base of $\mathcal{F}$; construct a semantic tree for $\mathcal{H}$ and extend each path until the first failure node is encountered. This tree must be closed since otherwise there would be a (possibly infinite) path without a failure node, and such a path would determine a Λ-consistent satisfying Herbrand interpretation for $\mathcal{F}$. Then each path will terminate at some finite depth, and, by König's Lemma, the closed tree is finite. Therefore, only a finite number of failure nodes are present, and this determines the required finite set of ground instances.

Suppose $\mathcal{F}$ has a finite set $\mathcal{F}'$ of ground instances that is unsatisfiable with respect to Λ-consistent interpretations. Every interpretation of $\mathcal{F}$ contains an interpretation of $\mathcal{F}'$ that must falsify $\mathcal{F}'$ (since $\mathcal{F}'$ is unsatisfiable), and hence that also falsifies $\mathcal{F}$. □

We note that all the usual results involving Robinson's Unification Theorem are unaffected by our development. Therefore, techniques at the ground level can typically be developed and lifted to the general level without difficulty.

3.2 Distribution Quantifiers

We begin our discussion of quantifiers in an MVL by recalling what we mean by free (i.e., unquantified) variables in a formula consisting of a single predicate P. As the free variables vary over the domain of discourse $\mathcal{D}$, P takes on different truth values; that is, if there are n free variables, P denotes a function from $\mathcal{D}^n \to \Delta$. Quantifying a variable certainly requires that a single truth value be produced for each $(n-1)$-tuple that the remaining free variables may take on. If we fix the $(n-1)$-tuple and ignore the quantifier for the moment, then, as the quantified variable varies over the domain, P takes on a non-empty set of truth values. Thus, the quantifier must assign that set of truth values a single truth value. As a result, a *distribution quantifier* $\mathcal{Q}$ in a multiple-valued logic Λ is defined as a function from the non-empty subsets of Δ to Δ:

$$\mathcal{Q} : (\mathcal{P}(\Delta) - \emptyset) \rightarrow \Delta,$$

where $\mathcal{P}(\Delta)$ denotes the power set of Δ.

Consider, for example, the expression $(\mathcal{Q}x)P(x)$, where $\mathcal{Q}$ is a quantifier and P is a predicate. As x varies over the domain of discourse, $P(x)$ takes on a (non-empty) set S of truth values. Then the truth value of $(\mathcal{Q}x)P(x)$ is $\mathcal{Q}(S)$. Although some care is required when dealing with a formula with more than one quantified variable, essentially there is no difficulty. For example, given $\mathcal{Q}_1 x \mathcal{Q}_2 y \mathcal{F}(x, y)$, for each x in the domain of discourse, $\mathcal{Q}_2 y \mathcal{F}(x, y)$ has a truth value and thus may be thought of as a predicate with free variable x. It takes on a set of truth values as x varies over the domain, and $\mathcal{Q}_1$ is defined on that set. This analysis obviously extends to any number of quantifiers.

Thought of in this way, the classical universal quantifier $\forall$ maps $\{$*true*$\}$ to *true* and maps both $\{$*true*, *false*$\}$ and $\{$*false*$\}$ to *false*; the classical existential quantifier $\exists$ maps both $\{$*true*, *false*$\}$ and $\{$*true*$\}$ to *true* and maps $\{$*false*$\}$ to *false*. Another way to look at these two quantifiers is to assume that $false < true$; then $\forall$ is the minimum and $\exists$ is the maximum. This leads to a natural generalization to any MVL whose truth domain is a complete lattice: $\forall$ and $\exists$ are, respectively, the infimum and supremum functions. That is, if S is any non-empty subset of Δ, then $\forall(S) = \text{Inf } S$, and $\exists(S) = \text{Sup } S$. For the remainder of this section, we assume that Δ is a complete distributive lattice, and that $\forall$ and $\exists$ are the infimum and supremum defined on $(2^{\Delta} - \{\emptyset\})$.

These two quantifiers can be translated to quantifiers in Λ_S for *regular* signs, which are defined carefully in Section 4.1. All we need for the moment is the *upset* of an element α in a lattice Δ: $\uparrow\alpha = \{\beta \in \Delta : \alpha \leq \beta\}$. In the next lemma, the symbol $\forall$ is used to denote both a quantifier in Λ and a quantifier in Λ_S.

Lemma 4. Let $\forall x \mathcal{F}$ be a formula in Λ in which x is the only variable, and let S be an upset of some element of Δ. Then the truth value of $\forall x \mathcal{F}$ is in S iff $\forall x(S\!:\!\mathcal{F}) = true$.

Proof: Let $\mathcal{R}$ be the range of $\mathcal{F}$ with respect to some interpretation I, and suppose first that $\forall x \mathcal{F} \in S$. Then $\text{Inf } \mathcal{R} \in S$. Since S is an upset, every element of Δ that is greater than $\text{Inf } \mathcal{R}$ is in S; thus $\mathcal{R} \subset S$. In particular, for every x in the domain of discourse, $\mathcal{F}$ evaluates to a truth value in S. Hence, $\forall x(S\!:\!\mathcal{F}) = true$.

Conversely, suppose $\forall x(S\!:\!\mathcal{F}) = true$. Then $\mathcal{R} \subset S$. Since S is an upset, the infimum of every subset of S is in S, so $\text{Inf } \mathcal{R} \in S$. □.

For existential quantifiers we require an additional assumption: Every subset of the lattice has a maximum. (Equivalently, the supremum of every subset of Δ is in the subset.)

Lemma 5. Suppose that every subset of Δ has a maximum. Then if $\exists x \mathcal{F}$ is a formula in Λ in which x is the only variable, and if S is an upset of some element of Δ, the truth value of $\exists x \mathcal{F}$ is in S iff $\exists x S\!:\!\mathcal{F} = true$.

Proof: Let $\mathcal{R}$ be the range of $\mathcal{F}$ with respect to some interpretation I, and suppose first that $\exists x \mathcal{F} \in S$. Then $\delta = \text{Sup } \mathcal{R} \in S$. By assumption $\delta \in \mathcal{R}$. Thus, for some x, $\mathcal{F}[x] \in S$; hence $\exists x S\!:\!\mathcal{F} = true$.

Conversely, suppose $\exists x S\!:\!\mathcal{F} = true$. Then, for some x, the truth value δ of $\mathcal{F}$ is in S. Since S is an upset, anything greater than δ is also in S. In particular, $\text{Sup } \mathcal{R} \in S$. □.

These two lemmas can be extended to multiply quantified formulas. This can be accomplished in a similar fashion to the manner in which the semantics of distribution quantifiers were extended. A formula with one bound variable and one free variable denotes a function from the domain of discourse to Δ. Then the formula can be quantified as if the free variable were the only variable. This reasoning can be applied repeatedly for any number of quantifiers. Note that the condition that supremums be maxima is required if any of the quantifiers is $\exists$. Finally, if a formula has both free and bound variables, then for each interpretation and any valuation, the lemmas hold. This proves

Theorem 2. Let $\mathcal{F}$ be a formula in Λ with n free variables and with m variables bound by $\forall$ or $\exists$, and suppose that S is an upset in Δ. If any of the quantifers are $\exists$, assume further that every subset of Δ has a maximum. Then for each interpretation I and corresponding domain of discourse $\mathcal{D}$, $S\!:\!\mathcal{F}$ in Λ_S denotes a function from $\mathcal{D}^n : \{true, false\}$, and for each valuation A, $I^A(\mathcal{F}) \in S$ iff $I_{\Lambda_S}^A(S\!:\!\mathcal{F}) = true$. $\square$

3.3 Signed Inference

In this section, we describe a method for adapting resolution to produce an inference rule for Λ_S. Many classical inference rules begin with links (complementary pairs of literals). Such rules typically deal only with formulas in which all negations are at the atomic level. Similarly, the inference techniques described below require that signs be at the "atomic level." To that end, we call a formula in Λ_S Λ-*atomic* if it has the property that whenever $S\!:\!A$ is an atom in the formula, then A is an atom in Λ; i.e., if the only atoms in the formula are signed atoms. Often, one assumes that all formulas are Λ-atomic (for example, formulas in annotated logics). There are also many situations in which driving signs inward to the atomic level is straightforward. One such is when Δ is finite — see [32]; another is with fuzzy operator logic — see Section 5.1. In all cases, specific knowledge of the non-logical symbols in Λ is essential. See [32] for details.

As mentioned above, our inference rules are based on the notion of complementary literals, which is captured and generalized in the next lemma. The proof of the lemma follows immediately from part 4 of Lemma 1.

Lemma 6. (The Reduction Lemma) Let $S_1{:}A_1$ and $S_2{:}A_2$ be Λ-atomic atoms in Λ_S where A_1 and A_2 are unifiable with most general unifier (mgu) σ; then

$$S_1{:}A_1 \wedge S_2{:}A_2 \models_\Lambda (S_1 \cap S_2){:}A_1\sigma \quad \text{and} \quad S_1{:}A_1 \vee S_2{:}A_2 \models_\Lambda (S_1 \cup S_2){:}A_1\sigma. \qquad \square$$

The Reduction Lemma encompasses both the classical notion of resolution (i.e., the first part) and merging (i.e., the second part). Observe that in the ground case, the lemma may be strengthened from entailment to equivalence in both parts.

3.4 An Example

DECIDE WHERE TO PUT THIS EXAMPLE

Priest proposed the following *paraconsistent* (i.e., inconsistency tolerant) logic, denoted LP [34]. The domain of truth values Δ is $\{\{0\},\{1\},\{0,1\}\}$. Thus, interpretations assign to each atom one of the sets $\{1\}$, $\{0\}$, $\{1,0\}$. An atom α is said to be $true$ under an interpretation I if $1 \in I(\alpha)$. It is $false$ if $0 \in I(\alpha)$. Negation and conjunction are defined in the usual way: $\neg\alpha$ is $true$ iff α is $false$

and $\neg\alpha$ is $false$ iff α is $true$; $\alpha \wedge \beta$ is $true$ iff both α and β is $true$ and $\alpha \wedge \beta$ is $false$ iff either α or β is $false$. Similarly, disjunction and implication are treated in the same way as classical logic: $\alpha \vee \beta \equiv \neg(\neg\alpha \wedge \neg\beta)$ and $\alpha \rightarrow \beta \equiv \neg\alpha \vee \beta$. Finally, models and logical consequence are defined in the usual way in LP.

Consider the formula $p \wedge \neg p$. If we are interested in determining whether its truth value evaluates to $\{0,1\}$, this corresponds to the signed formula $\{\{0,1\}\} : p \wedge \neg p$. Applying Lemma **??** and simplifying yields the Λ-atomic formula

$$
\begin{aligned}
&(\{\{0,1\}\} : p \wedge \{\{0\}\} : p) \vee \\
&(\{\{1\}\} : p \wedge \{\{0,1\}\} : p) \vee \\
&(\{\{0,1\}\} : p
\end{aligned}
$$

The first two conjuncts reduces to $\{\} : p$ which is not satisfied by any Λ-consistent interpretation. It follows from the third disjunct that, as we expect, p must evaluate to both 0 and 1.

On the other hand, we would expect $\{\{1\}\} : p \wedge \neg p$ be $false$. Again, applying Lemma **??**, we obtain

$$(\{\{1\}\} : p \wedge \{\{0\}\} : p)$$

which does not have any satisfying Λ-consistent interpretation.

3.5 Signed Resolution

Here we generalize the resolution inference rule for classical logic. Combined with the Reduction Lemma, this generalization will lead naturally to an inference rule for Λ_S. We begin with the ground case.

Let $\mathcal{F}$ be a conjunction of disjunctions in a classical logic. We use the term clause for the disjunctions even though we do not assume them to be disjunctions of literals. Let two of the clauses be

$$C = Q_C \vee \left(\bigvee_{j=1}^{m_C} J_j \right) \text{ and } D = Q_D \vee \left(\bigvee_{j=1}^{m_D} K_j \right).$$

Then we may *resolve* clauses C and D and infer

$$(Q_C \wedge Q_D) \vee \left(\bigvee_{j=1}^{m_C} J_j \right) \vee \left(\bigvee_{j=1}^{m_D} K_j \right).$$

Soundness follows from noting that if $C \wedge D$ is $true$, but none of the J_j's and none of the K_j's are, then Q_C and Q_D must both be $true$. Observe that when C and D are clauses of literals and Q_C are Q_D are complimentary, this rule reduces to standard binary resolution.

Consider now a Λ-atomic formula $\mathcal{F}$ in Λ_S that is in conjunctive normal form (CNF). As such, the clauses of $\mathcal{F}$ are sets of literals. Let $C_j, 1 \leq j \leq r$, be clauses in $\mathcal{F}$ that contain, respectively, Λ-atomic atoms $\{S_j{:}A\}$. Thus we may write $C_j = K_j \vee \{S_j{:}A\}$. Then the resolvent R of the C_j's is defined to be the clause

$$\left(\bigvee_{j=1}^{r} K_j \right) \bigvee \left(\left(\bigcap_{j=1}^{r} S_j \right) : A \right);$$

this definition is the one given above, generalized to n clauses and specialized to Λ-consistent interpretations via the Reduction Lemma. The rightmost disjunct is called the *remnant* of the resolution; observe that it is unsatisfiable if its sign is empty and satisfiable if it is not.

In the first order case, we assume closed formulas in which existential quantifiers have been removed by Skolemization; thus all variables are implicitly universally quantified. Let $C_j, 1 \leq j \leq r$, be clauses in $\mathcal{F}$ (we assume variables have been standardized apart) that contain, respectively, Λ-atomic atoms $\{S_j{:}A_j\}$, where the A_i's are unifiable with most general unifier (mgu) σ. We may write $C_j = K_j \vee \{S_j{:}A_j\}$. Then the resolvent R of the C_j's is defined to be the clause

$$\left(\bigvee_{j=1}^{r} K_j\right)\sigma \quad \bigvee \quad \left(\left(\bigcap_{j=1}^{r} S_j\right) : A\right)\sigma \quad .$$

We must augment this definition with the following obvious simplification rules that stem from the Reduction Lemma. First, if the remnant is unsatisfiable it may simply be deleted from R. Secondly, whenever R contains literals $S_i{:}B$, $1 \leq i \leq k$, we *merge* them into the literal $\bigcup_{i=1}^{k} S_i{:}B$; if $\bigcup_{i=1}^{k} S_i = \Delta$, then R is a tautology and may be deleted from $\mathcal{F}$. Such merging is essentially a generalization to MVL's of ordinary classical ground merging.

A more general merge operation is possible in the first order case; it corresponds to the factoring rule of standard resolution theory. Let $C = C' \bigcup \{S_i : B_i, 1 \leq i \leq k\}$ be a clause in $\mathcal{F}$, where the B_i's are unifiable with mgu σ (i.e., $B = \sigma B_i$). Then we may infer $\left(C' \bigvee \left(\bigcup_{i=1}^{k} S_i\right) : B\right)\sigma$.

The classical notion of subsumption also generalizes to Λ_S: clause C subsumes clause D if there is a substitution θ such that, for every literal $S{:}A \in C\theta$, there is a literal $S'{:}A \in D$ such that $S \subset S'$.

3.6 Completeness

The resolution rule presented above was introduced in [28] for a class of multiple-valued logics. There the truth value domains were assumed to obey rather severe restrictions, and all queries had the form: Can a formula evaluate to 1 (the maximal element of a truth value lattice)? Refutation completeness was proven in the ground case using a modified semantic tree argument in which interpretations over the MVL were represented.

In this paper, we are dealing with a much wider class of first order MVL's and are allowing more general queries about the formulas in those MVL's. The resolution rule and the signed formulas on which it operates, however, are defined in a classical logic. In this setting, completeness is not a statement about what formulas in Λ can be proved inconsistent; rather, completeness means that we can answer certain queries about formulas in Λ by deriving the empty clause in Λ_S. Nevertheless, the semantic tree construction in [28] can be adapted to produce a completeness argument for Λ_S. The crucial observation is that the set of signs in the initial set of clauses is finite. Regardless of the cardinality of Δ, an arbitrary combination of unions and intersections of a finite set of signs (subsets of Δ) can only yield finitely many distinct signs. (In [28], Δ must be finite; here Δ can be uncountable.) This implies that the Herbrand base remains countable, and hence a binary semantic tree argument can be constructed.

In the proof of Theorem 3, however, we use Anderson and Bledsoe's method of induction on the number of excess literals [1]. This technique (as well as the semantic tree argument) works in the ground case, and the usual lifting lemma provides completeness in the general case.

If $\mathcal{C}$ is a Λ-atomic formula, and if there is a derivation of the empty clause from $\mathcal{C}$ using signed resolution, we will write $\mathcal{C} \vdash \Box$. If $\mathcal{C}$ is in CNF, we treat $\mathcal{C}$ as a set of clauses in which each clause is a set of literals (signed atoms). The *number of excess literals* in $\mathcal{C}$ is defined to be the number of literal occurrences in $\mathcal{C}$ minus the number of clauses in $\mathcal{C}$.

Theorem 3. Let $\mathcal{F}$ be a (ground) formula in Λ, let S be any subset of Δ such that no interpretation maps $\mathcal{F}$ into S, and let $\mathcal{L}$ be a Λ-atomic formula such that $\mathcal{L} \equiv_\Lambda \mathcal{F}$. Then $\mathcal{L} \vdash \Box$.

Proof. First, since $\mathcal{L}$ has only the conjunction and disjunction operators of standard logic, $\mathcal{L}$ can be transformed into an equivalent CNF formula $\mathcal{C}$. We will show that $\mathcal{C} \vdash \Box$ by induction on the number n of excess literals.

Suppose first that $n = 0$. If $\Box \in \mathcal{C}$, we are done; otherwise, $\mathcal{C}$ contains only unit clauses. Since $\mathcal{C}$ is Λ-unsatisfiable, for some predicate P, the unit clauses $\{S_i : P\}$, $1 \leq i \leq m$, must be in $\mathcal{C}$, where $\bigcap_{i=1}^m S_i = \emptyset$. (Otherwise, the assignment over Λ that maps P to δ, where $\delta \in \bigcap_{i=1}^m S_i$, induces a Λ-consistent interpretation in which each $S_i : P$ is true.) The signed resolvent of $\{S_i : P\}$, $1 \leq i \leq m$ is $\Box$.

Assume now that whenever an unsatisfiable clause set has at most n excess literals, there is a derivation of the empty clause, and let $\mathcal{C}$ be a Λ-unsatisfiable CNF formula in Λ_S for which the number of excess literals is $n+1$. Since $n+1 > 0$, there is a non-unit clause $H = H' \bigvee S : A$ in $\mathcal{C}$. If $(\mathcal{C} - \{H\})$ is unsatisfiable, then the induction hypothesis applies. If not, any minimal Λ-unsatisfiable subset of $\mathcal{C}$ must include H, and any refutation of $\mathcal{C}$ must contain at least one occurrence of H.

Let $\mathcal{C}_{H'} = (\mathcal{C} - \{H\}) \cup H'$, and let $\mathcal{C}_{S:A} = (\mathcal{C} - \{H\}) \cup \{S : A\}$. Obviously, $\mathcal{C}_{H'}$ and $\mathcal{C}_{S:A}$ are both unsatisfiable. Since the number of excess literals in both is less than $n+1$, the induction hypothesis applies: $\mathcal{C}_{H'} \vdash \Box$ and $\mathcal{C}_{s:A} \vdash \Box$. Denote by $\mathcal{D}_{H'}$ the derivation of $\Box$ from $\mathcal{C}_{H'}$ and by $\mathcal{D}_{S:A}$ the derivation of $\Box$ from $\mathcal{C}_{S:A}$. Let $\mathcal{D}_H$ be the derivation obtained by replacing every occurrence of H' in $\mathcal{D}_{H'}$ by H. Notice that $\mathcal{D}_H$ is a derivation from $\mathcal{C}$. Since $(\mathcal{C} - \{H\}) \not\models \Box$, H' must occur in $D_{H'}$, otherwise D_H would be a refutation of $\mathcal{C}$ not including H. If $\Box \in D_H$, then we are done. If not, then $\mathcal{C} \vdash \{S : A\}$.

Now consider $\mathcal{D}_{S:A}$; we have $((\mathcal{C} - \{H\}) \cup \{s : A\}) \vdash \Box$. Thus $\mathcal{D}_H$ produces the unit clause $\{S : A\}$ required by $\mathcal{D}_{S:A}$, and we may append $\mathcal{D}_H$ and $\mathcal{D}_{S:A}$ to produce $\mathcal{D}_H\mathcal{D}_{S:A}$ which is a derivation of the empty clause from $\mathcal{C}$. $\Box$

Many authors have found a linear restriction desirable; this is especially true for logic programming purposes. A resolution strategy is called *linear* if, at each step, one of the parent clauses is the most recent resolvent. The above proof can be modified to restrict deductions to be linear as follows.

The base case is obvious. In the inductive step, choose H' to be the top clause in $\mathcal{D}_{H'}$ which is linear by the induction hypothesis. The reasoning proceeds as before and results in a deduction $\mathcal{D}_H$ of $\{S : A\}$ that is linear. By choosing $\{S : A\}$ to be the top clause in $\mathcal{D}_{S:A}$ (noting that $\mathcal{D}_{S:A}$ is also linear), we are able to append the two deductions as before, and the resulting deduction is guaranteed to be linear. As a result we have

Corollary. Theorem 3 remains valid with the linear restriction. $\Box$

We may add to the linear restriction the requirement that only resolution steps in which the remnant is unsatisfiable are allowed. This also has little effect on the proof of the theorem (with or without the linear restriction): The base case is again obvious, and the rest goes through as before.

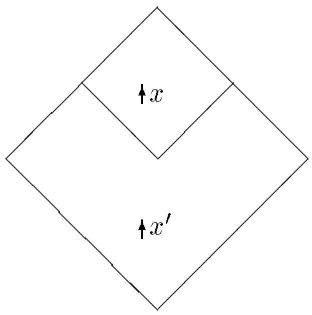

Figure 1: Regular Signs

4 Annotated Logics

In this section, we describe the connection between signed formulas and *annotated logics*, which is a class of multiple-valued logics that has been applied to knowledge representation, expert database systems, quantitative reasoning, and hybrid databases [3, 14, 15, 16, 17, ?, 25, ?, 38, ?]. Specifically, we show that the class of annotated logics corresponds to the class of *regular* signed formulas. As a consequence, signed resolution and its various restrictions, including linear signed resolution, may be employed for reasoning about annotated logics.

4.1 Regular Signed Formulas and Annotated Logics

Assume now that the set of truth values Δ is not simply an unordered set of objects but instead forms a complete lattice under some ordering $\leq$. The greatest and least elements of Δ are denoted $\top$ and $\bot$, respectively, and Sup and Inf denote, respectively, the supremum (least upper bound) and infimum (greatest lower bound) of a subset of Δ.

Let $\langle P; \leq \rangle$ be any partially ordered set, and let $Q \subseteq P$. Then $\uparrow Q = \{y \in P | (\exists x \in Q)\ x \leq y\}$. Note that $\uparrow Q$ is the smallest *upset* containing Q (see [6]). If Q is a singleton set $\{x\}$, then we simply write $\uparrow x$. We say that a subset Q of P is *regular* if for some $x \in P$, $Q = \uparrow x$ or $Q = (\uparrow x)'$ (the set complement of $\uparrow x$). In the former case, we call Q *positive*, and in the latter *negative*. A signed formula is regular if every sign that occurs in it is regular. By Part 1 of Lemma 1, we may assume that no regular signed formulas have any signs of the form $(\uparrow x)'$, where $x = \bot$, since in that case $(\uparrow x)' = \emptyset$.

The diagram in Figure 1 is a good way of thinking of regular signs. The larger diamond represents a lattice, and the point x is an element in the lattice. The inner diamond is the set $\uparrow x$, while the complement is the set $(\uparrow x)'$.

Let $\mathcal{A}$ be the atom set, and let the complete lattice Δ be the domain of truth values, as before. Then the propositional *annotated logic* $\mathbf{P}\mathcal{T}$ is defined as follows. If $A \in \mathcal{A}$ and $\mu \in \Delta$, then $A{:}\mu$ is an *annotated atom*, and an *annotated literal* is either an annotated atom or the negation of an annotated atom. The formulas of $\mathbf{P}\mathcal{T}$ are those that are constructed from annotated literals (positive or negative) using the connectives $\wedge$ and $\vee$. As usual, an interpretation is a mapping from $\mathcal{A}$ to Δ; we use $\mathcal{I}$ to denote the set of all interpretations for $\mathbf{P}\mathcal{T}$. An interpretation I is said to satisfy

1. the annotated atom $A{:}\mu$ iff $I(A) \geq \mu$;
2. the annotated literal $\sim(A{:}\mu)$ iff I does not satisfy $A{:}\mu$;
3. the disjunction $\mathcal{F}_1 \vee \mathcal{F}_2$ if it satisfies either $\mathcal{F}_1$ or $\mathcal{F}_2$;

4. the conjunction $\mathcal{F}_1 \wedge \mathcal{F}_2$ if it satisfies both $\mathcal{F}_1$ and $\mathcal{F}_2$.

The relationship between regular signed formulas and the annotated logic $\mathbf{P}\mathcal{T}$ is made precise in Theorem 4. First, define the mapping Sgn from annotated literals to regular signed atoms as follows: If $A{:}\mu$ is a positive annotated literal, then $Sgn(A{:}\mu) = {\uparrow}\mu{:}A$, and if $\sim(A{:}\mu)$ is a negative annotated literal, then $Sgn(\sim(A{:}\mu)) = ({\uparrow}\mu)'{:}A$. The mapping may be extended to any formula in the obvious way: If $\mathcal{F}$ and $\mathcal{G}$ are annotated formulas, then $Sgn\mathcal{F} \wedge \mathcal{G}) = Sgn(\mathcal{F}) \wedge Sgn(\mathcal{G})$, and $Sgn(\mathcal{F} \vee \mathcal{G}) = Sgn(\mathcal{F}) \vee Sgn(\mathcal{G})$. Observe that Sgn is a bijection between annotated formulas and regular signed formulas.

Theorem 4. Suppose $\mathcal{F}$ is an annotated formula and I is an interpretation. Then I satisfies $\mathcal{F}$ iff I_c (the corresponding interpretation over Λ_S) satisfies $Sgn(\mathcal{F})$.

Proof. If $\mathcal{F}$ is a positive annotated literal of the form $A : \mu$, and if I satisfies $\mathcal{F}$, then $I(A) \geq \mu$. It follows that $I(A) \in \uparrow \mu$. Hence I_c satisfies $\uparrow \mu : A = Sgn(\mathcal{F})$. If $\mathcal{F}$ is a negative annotated literal of the form $\sim (A : \mu)$, and if I satisfies $\mathcal{F}$, then $I(A) \not\geq \mu$. It follows that $I(A) \in (\uparrow x)'$. Hence I_c satisfies $(\uparrow x)' : A = Sgn(\mathcal{F})$. The case where $\mathcal{F}$ is a conjunction or a disjunction follows by induction. $\square$

The above result is straightforward and illustrates a natural semantic connection between annotated logics and signed formulas. Hähnle [11] has considered a class of multiple-valued logics, called *regular logics*, that appear to be closely related to regular signed formulas. [1]

4.2 Signed Resolution for Annotated Logics

A sound and complete resolution proof procedure was defined for clausal annotated logics in [14], and an extension was introduced in [24]. The procedure contains two inference rules that we will refer to as *annotated resolution* and *reduction.*[2] In this section we show that, under the mapping sgn, these two procedures correspond to disjoint instances of signed resolution.

Two annotated literals L_1 and L_2 are said to be complementary if they have the respective forms $A{:}\mu$ and $\sim(A{:}\rho)$, where $\mu \geq \rho$.

Definition. (Annotated Resolution) Given the annotated clauses $(L_1 \vee D_1)$ and $(L_2 \vee D_2)$, where L_1 and L_2 are complementary, then the *annotated resolvent* of the two clauses on the annotated literals L_1 and L_2 is $D_1 \vee D_2$.

Two clauses can be so resolved only if the annotation of the positive annotated literal that is resolved upon is greater than or equal to the annotation of the negative literal resolved upon. In that case the two clauses are said to be *resolvable* on the annotated literals L_1 and L_2.

Definition. (Reduction) Suppose $(A{:}\mu_1 \vee E_1)$ and $(A{:}\mu_2 \vee E_2)$, are two annotated clauses in which μ_1 and μ_2 are incomparable. Then the annotated clause $(A{:}\mathrm{Sup}\ \{\mu_1, \mu_2\}) \vee E_1 \vee E_2$ is called a *reductant* of the two clauses, and we say that the two clauses are *reducible* on the annotated literals $A{:}\mu_1$ and $A{:}\mu_2$.

Lemma 7. Suppose L_1 and L_2 are the complementary annotated literals $A : \mu$ and $\sim (A : \rho)$, respectively. If $S_1 : A = Sgn(L_1)$ and $S_2 : A = Sgn(L_2)$, then $S_1 \cap S_2 = \emptyset$.

[1]We are grateful to Reiner Hähnle for pointing out the similarity. The precise connection between the two is currently under investigation.

[2]Kifer and Lozinskii refer to their first inference rule simply as resolution. However, since we are working with several resolution rules in this paper, we will use appropriate prefixes to avoid ambiguity.

Proof. Since $Sgn(L_1) =\uparrow \mu : A$ and $Sgn(L_2) = (\uparrow \rho)' : A$, and since $\mu \geq \rho$, we clearly have $\uparrow \mu \cap (\uparrow \rho)' = \emptyset$. $\square$

Corollary. Suppose C_1 and C_2 are annotated clauses that are annotated-resolvable on the annotated literals L_1 and L_2. Then $Sgn(C_1)$ and $Sgn(C_2)$ are sign-resolvable with the unsatisfiable remnant on the signed atoms $Sgn(L_1)$ and $Sgn(L_2)$. $\square$

Lemma 8. Suppose L_1 and L_2 are the annotated literals $A : \mu_1$ and $A : \mu_2$, respectively. If $S_1 : A = Sgn(L_1)$ and $S_2 : A = Sgn(L_2)$, then $S_1 \cap S_2 =\uparrow \sqcup\{\mu_1, \mu_2\}$.

Proof. Since $S_1 =\uparrow \mu_1$ and $S_2 =\uparrow \mu_2$, we have $\delta \in S_1 \cap S_2$ iff $\delta \geq \mu_1$ and $\delta \geq \mu_2$, iff $\delta \geq \sqcup\{\mu_1, \mu_2\}$, iff $\delta \in\uparrow \sqcup\{\mu_1, \mu_2\}$. $\square$

Corollary. Suppose C_1 and C_2 are annotated clauses that are reducible on the annotated literals L_1 and L_2. Then $Sgn(C_1)$ and $Sgn(C_2)$ are sign-resolvable with the remnant satisfiable on the signed atoms $Sgn(L_1)$ and $Sgn(L_2)$. $\square$

The last two lemmas demonstrate that annotated resolution and reduction correspond to, respectively, the case when signed resolution produces an unsatisfiable remnant and the case when signed resolution produces a satisfiable remnant. We may therefore regard signed resolution as unifying annotated resolution and reduction.

We extend the function Sgn to apply to sequences of clauses in the obvious way: Given a sequence of annotated clauses $D = C_1, \ldots, C_n$, $Sgn(D)$ denotes the sequence of signed clauses $Sgn(C_1), \ldots, Sgn(C_n)$.

The next theorem is now immediate.

Theorem 5. Suppose that $\mathcal{F}$ is a set of annotated clauses and that $\mathcal{D}$ is a deduction of $\mathcal{F}$ using annotated resolution and reduction. Then $Sgn(\mathcal{D})$ is a signed deduction of $Sgn(\mathcal{F})$. In particular, if $\mathcal{F}$ is an unsatisfiable set of first order annotated clauses, then there is a signed refutation of $Sgn(\mathcal{F})$. $\square$

4.3 Regular Signed Resolution

The results of the previous section indicate that signed resolution provides a simple and general method for implementing annotated logics. However, as is often the case in theorem proving, there is a trade-off between generality and efficiency. Since annotated resolution and reduction are instances of signed resolution, and since there exist signed resolutions for which no corresponding annotation resolutions or reductions exist, the search space induced by signed resolution will generally be larger than the search space induced by annotated resolution and reduction. In this section, to overcome this inefficiency, we further characterize the class of signed deductions restricted to regular signed clauses.

Theorem 6. Suppose $S_1, \ldots, S_n$ are regular signs whose intersection is empty, and suppose that no subset of $\{S_1, \ldots, S_n\}$ has an empty intersection. Then for some j, $1 \leq j \leq n$, $S_j = (\uparrow x_j)'$, and for $i \neq j$, $S_i = \uparrow x_i$, where $x_1, \ldots, x_n \in \Delta$. $\square$

Proof. Reordering if necessary, let $S_1, \ldots, S_k$ be the positive signs, and let $S_{k+1}, \ldots, S_n$ be the negative signs. If $k = n$, i.e., if each S_i has the form $\uparrow x_i$, then $\cap_{i=1}^n S_i =\uparrow (\sqcup_{i=1}^n x_i)$, which is non-empty. Similarly, if $k = 0$, i.e., if each S_i has the form $(\uparrow x)'$, then $\cap_{i=1}^n S_i$ would also be non-empty. Thus $1 < k < n$.

We must show that $k = n - 1$, so suppose $k < n - 1$, and let $S = \cap_{i=1}^k S_i$. Note that S may be written equivalently as $\uparrow \sqcup_{i=1}^k x_i$. Since $\cap_{i=k+1}^n (\uparrow x_i)' \cap S = \emptyset$, we must have, for at least one $j > k$, $x_j \not\geq \sqcup_{i=1}^k x_i$. But then $(\uparrow x_j)' \cap S = \emptyset$, contradicting the minimality condition. □

We say that a literal $L_1 \in C_1$ *supports* a literal $L \in C$ if $L = L_1$ and $C = C_1$ or if C is the result of C_1 resolving with another clause on the literal L_1, and L is the remnant. In addition, if $L_1 \in C_1$ supports $L_2 \in C_2$, and if L_2 supports $L_3 \in C_3$, then L_1 also supports L_3. Thus supports is a reflexive transitive relation. Equivalently we say that L_3 is supported by L_1. In a deduction $\mathcal{D}$, the clauses C_1 and C_2 *indirectly resolve* upon the literals $L_1 \in C_1$ and $L_2 \in C_2$ if there is a resolution of two clauses $(K_1 \vee D_1)$ and $(K_2 \vee D_2)$ in $\mathcal{D}$ on K_1 and K_2, and if K_i is supported by L_i, $i = 1, 2$. Note that when C_1 and C_2 (directly) resolve, they also indirectly resolve. The significance of the last theorem is that one may impose the restriction that no two negative literals be indirectly resolved. Experiments indicate that this will typically (but not necessarily) reduce the number of deductions required for finding a proof.

A signed deduction is *regular* if every sign that appears in the deduction is regular.

Theorem 7. Suppose $\mathcal{D}$ is a regular deduction of a set S of clauses. Then $\mathcal{D}$ does not contain an indirect resolution on two literals with negative signs. □

Unrestricted signed resolution allows resolution between any two clauses of the form $((\uparrow x)' \colon A) \vee C$ and $((\uparrow y)' \colon A) \vee D$. In light of Theorem 7, irrelevant deduction steps that would be avoided by annotated resolution may be performed by signed resolution.

The next theorem tells us that the class of all regular signed deductions is precisely the class of all deductions using annotated resolution and reduction. As a result, restricting signed resolution to regular clauses captures annotated resolution and reduction without increasing the search space.

Theorem 8. Let $\mathcal{D}$ be a sequence of annotated clauses. Then $\mathcal{D}$ is an annotated deduction iff $Sgn(\mathcal{D})$ is a regular signed deduction.

Proof. If $\mathcal{D}$ is an annotated deduction, then $Sgn(\mathcal{D})$ is a signed deduction of $Sgn(\mathcal{F})$ by Theorem 5. Conversely, for each signed resolution, observe that the intersection of two negative regular signs is regular iff one is contained in the other, which we assume are eliminated by subsumption checks. Intersection of a positive with a negative regular sign, unless empty, cannot be regular since it neither contains $\top$ nor $\perp$. Thus the only possible signed resolutions that admit non-empty, regular signs must use two positive regular signs, which obviously corresponds to a reduction step. □

Corollary. (Completeness of Regular Signed Resolution) Suppose $\mathcal{F}$ is an unsatisfiable set of regular signed clauses. Then there is a regular signed deduction of the empty clause from $\mathcal{F}$. □

4.4 Linear Signed Resolution and Horn Sets

Logic Programming is based on the observation that the class of definite horn clauses have a naturally associated procedural interpretation based on linear resolution. In annotated logic, linear signed resolution provides the setting with which a set of definite horn annotated clauses may be viewed as a computer program. However, some complications arise.

Using the linear restriction no longer guarantees the "regularity" of a proof. Consequently, such a proof can not be a p-resolution proof since, according to Corollary 1, only regular signed proofs correspond to p-resolution proofs. This is illustrated in the next example.

Example 1. fill in

** anything else to say?? the implementation method has already been described in the ICLP paper **

5 Signed Formulas and Fuzzy Operator Logic

Fuzzy logics, introduced by Zadeh, have been studied extensively, and several important applications have been developed. On the whole, however, relatively little has been done to explore practical inference methods for such logics. Several authors ([19, 26, 20, 41], for example) have adapted resolution to fuzzy logics. Here we follow the treatment in [41, 20] and investigate their relationship between signed formulas and fuzzy operator logic.

5.1 Fuzzy Operators

In [41] a calculus called *Fuzzy Operator Logic* (FOpL) is developed. It is a multiple-valued logic whose set of truth values is the unit interval [0, 1]. Formulas of FOpL are built from the *fuzzy operator*, a generalized negation, and the connectives $\wedge$ and $\vee$. Formally, a *fuzzy formula* is defined as follows. Suppose A is an atom, $\mathcal{F}$ and $\mathcal{G}$ are fuzzy formulas, and $\phi \in [0,1]$[3]. Then

1. ϕA is a fuzzy formula, also called a fuzzy literal;
2. $\phi(\mathcal{F} \wedge \mathcal{G})$ is a fuzzy formula;
3. $\phi(\mathcal{F} \vee \mathcal{G})$ is a fuzzy formula.

A simple example of a fuzzy formula is

$$\mathcal{F} = A \wedge .3(.9B \vee .2C).$$

Several observations are in order. First, fuzzy operators are represented by real numbers in the unit interval. (That there are uncountably many fuzzy operators should not cause alarm. In practice, considering only rational fuzzy operators is not likely to be a problem. Indeed, with a computer implementation, we are restricted to a finite set of terminating decimals of at most n digits for some not very large n.) In particular, real numbers in the unit interval denote both truth values and fuzzy operators. Secondly, every formula and subformula is prefixed by a fuzzy operator; any subformula that does not have an explicit fuzzy operator prefix is understood to have 1 as its fuzzy operator.

In an MVL in which Δ is a complete lattice, the most natural generalizations of the connectives $\wedge$ and $\vee$ are as the infimum and supremum operators. Should the truth domain be linear — for example, should it be $[0,1]$ — then the infimum and supremum operators on finite sets represent the minimum and maximum. In other words, the infimum and supremum of a finite set in a linear domain are simply the minimum and the maximum, respectively. In [41] these connectives are not

[3] In [41] the authors use λ for the fuzzy operator and Λ for the threshold of acceptability (introduced in Section 5.2). We use ϕ and Γ to avoid confusion with notation used earlier in this and in previous papers.

defined explicitly, but the definition of *confidence* (described in Section 5.2) amounts to defining $\wedge$ and $\vee$ as the infimum and supremum.

The semantics of fuzzy operator are given via a kind of "fuzzy" product [41, 20].[4]

Definition. If $\phi, \delta \in [0, 1]$, then

$$\phi \otimes \delta = (2\phi - 1) \cdot \delta - \phi + 1.$$

Observe that $\otimes$ is commutative and associative. Also observe[5] that

$$\phi \otimes \delta = \phi \cdot \delta + (1 - \phi) \cdot (1 - \delta).$$

This last observation gives us the intuition behind the fuzzy product $\otimes$: Were ϕ the probability that A_1 is $true$ and were δ the probability that A_2 is $true$, then $\phi \otimes \delta$ would be the probability that A_1 and A_2 are both $true$ or both $false$. (This probabilistic analogy is for intuition only; fuzzy logic is not based on probability, as has been pointed out frequently in the literature.)

To give a semantics to a fuzzy formula $\phi\mathcal{F}$, ϕ must be defined as a function from [0, 1] to [0, 1]. It will be convenient to do so using functional notation, so define Φ_ϕ by $\Phi_\phi(\delta) = \phi \otimes \delta$. In [41], the truth value of a fuzzy formula $\phi\mathcal{F}$ is given recursively via the confidence function C_I, given below. Intuitively, C_I corresponds to interpretation over Λ as developed in Section 2 in that it maps fuzzy formulas to Δ, in this case [0,1].

Definition. (Confidence) **** ADD QUANTIFIERS ***** Let A be an atom, let $\mathcal{F}$ and $\mathcal{G}$ be fuzzy formulas, and let ϕ be a fuzzy operator. Then

1. $\mathcal{C}_I(A) = I(A)$, i.e., the truth value assigned to A by I[6];
2. $\mathcal{C}_I(\phi\mathcal{F}) = \phi \otimes \mathcal{C}_I(\mathcal{F}) = \Phi_\phi(\mathcal{C}_I(\mathcal{F}))$;
3. $\mathcal{C}_I(\mathcal{F} \wedge \mathcal{G}) = min\{\mathcal{C}_I(\mathcal{F}), \mathcal{C}_I(\mathcal{G})\}$;
4. $\mathcal{C}_I(\mathcal{F} \vee \mathcal{G}) = max\{\mathcal{C}_I(\mathcal{F}), \mathcal{C}_I(\mathcal{G})\}$.

With respect to the development of MVL in Section 2, we will extend interpretations in fuzzy operator logics to assign truth values to arbitrary fuzzy formulas by

$$I(\phi\mathcal{F}) = \mathcal{C}_I(\phi\mathcal{F}).$$

A useful intuition for understanding the fuzzy operator is as the likelyhood of the accuracy of the truth value of the formula. Hence intuitively, $1\mathcal{F}$ asserts the certainty of the assigned truth value of $\mathcal{F}$, $.8\mathcal{F}$ asserts that the assigned truth value is likely to be correct, and $.2\mathcal{F}$ asserts that the assigned truth value of $\mathcal{F}$ is likely to be wrong. It is important to distinguish between the fuzzy operator as the likelihood of the accuracy of the truth value of a formula and the actual truth value of the formula, given by C_I. For example, if $\mathcal{F} = 0$ ($false$), then the truth value of $0\mathcal{F}$ is 1; the fuzzy operator 0 indicates that the truth value of $\mathcal{F}$ is certain to be wrong.

There are three fuzzy operators that play a special role: 0, .5, and 1. Let $\delta' = \phi \otimes \delta$. If $\phi = 1$, then $\delta' = \delta$. In effect this says that if we're certain of the truth value of $\mathcal{F}$, then $\mathcal{F}$ has the truth value of $\mathcal{F}$. If $\phi = 0$, then $\delta' = 1 - \delta$. The operator 0 asserts certainty that the assigned truth value is wrong and converts the truth value to its complement. This is essentially standard negation.

[4] Their choice of the operator $\otimes$ was well thought out: It produces exactly the right unary operators. They do **not** use the term *fuzzy product*.

[5] We would like to thank Don Fridshal for pointing out this relationship.

[6] Interpretations in [41] are limited to assigning 1 or 0 ($true$ or $false$) to atoms; this restriction is unnecessary.

Lastly, if $\phi = .5$, then $\delta' = .5$. This is not surprising since $\phi = .5$ indicates a complete uncertainty regarding the truth value of $\mathcal{F}$.

We can better understand the semantics of fuzzy operator logic by distinguishing between a fuzzy operator in [0, .5) and a fuzzy operator in (.5, 1]. Let $\phi = .5 + \phi'$. Thus, $\phi' < 0$ in the former case, and $\phi' > 0$ in the latter. If δ is any truth value ($\delta \in [0, 1]$), then manipulating $(\phi' + .5) \otimes \delta$ produces

$$\phi \otimes \delta = .5 - \phi' + 2\phi'\delta.$$

Observe first that $\phi \otimes \delta = .5$ iff $\phi' = 0$ or $\delta = .5$; i.e., iff $\phi = .5$ or $\delta = .5$. Also, as a function of δ, the fuzzy operator ϕ is monotonically increasing or decreasing as $\phi' > 0$ or $\phi' < 0$, i.e., as $\phi > .5$ or $\phi < .5$. In particular, if $\phi > .5$, then

$$\phi \otimes min\{\delta_1, \delta_2\} = min\{\phi \otimes \delta_1, \phi \otimes \delta_2\}$$

and

$$\phi \otimes max\{\delta_1, \delta_2\} = max\{\phi \otimes \delta_1, \phi \otimes \delta_2\},$$

but if $\phi < .5$, then

$$\phi \otimes min\{\delta_1, \delta_2\} = max\{\phi \otimes \delta_1, \phi \otimes \delta_2\}$$

and

$$\phi \otimes max\{\delta_1, \delta_2\} = min\{\phi \otimes \delta_1, \phi \otimes \delta_2\}.$$

This combined with the associativity of $\otimes$ proves the following lemma.[7]

Lemma 9. Let $\mathcal{F}$ and $\mathcal{G}$ be fuzzy formulas, and let ϕ be a fuzzy operator. If $\phi > .5$, then

$$\phi(\mathcal{F} \wedge \mathcal{G}) = \phi\mathcal{F} \wedge \phi\mathcal{G} \quad \text{and} \quad \phi(\mathcal{F} \vee \mathcal{G}) = \phi\mathcal{F} \vee \phi\mathcal{G}.$$

If $\phi < .5$, then

$$\phi(\mathcal{F} \wedge \mathcal{G}) = \phi\mathcal{F} \vee \phi\mathcal{G} \quad \text{and} \quad \phi(\mathcal{F} \vee \mathcal{G}) = \phi\mathcal{F} \wedge \phi\mathcal{G}.$$

In particular, every fuzzy formula is equivalent to one in which 1 is the only fuzzy operator applied to non-atomic arguments. □

In view of the lemma, a fuzzy operator ϕ is called *positive* (resp. *negative*) if $\phi > .5$ (resp. $\phi < .5$). Similarly, a sign is *positive* (resp. *negative*) if the sign is a subset of (.5, 1] (resp. [0, .5)). Note that signs may be neither positive nor negative.

5.2 Fuzzy Resolution

The fuzzy resolution inference considered in [41] works with fuzzy formulas in conjunctive normal form – conjunctions of disjunctions of literals. Lemma 9 provides a means of converting any fuzzy formula to an equivalent conjunctive normal form.

The fuzzy resolution rule is pretty much the standard resolution rule once we have a notion of satisfiability and know what a complementary pair of literals is. In essence, for an interpretation to satisfy a formula, we want the confidence of the formula under the interpretation to be close to 1. More precisely, we assume a *threshold of acceptability* $\Gamma \in [0.5, 1]$. Then the interpretation I is said to Γ-*satisfy* the formula $\mathcal{F}$ if $\mathcal{C}_I(\mathcal{F}) \geq \Gamma$.

[7] The same result was proved in [41, pp. 74–75]. However, what we have presented hopefully provides a more intuitively clearer development of the role and importance of fuzzy operator.

Definition. (Fuzzy Consequence) Suppose F and G are two fuzzy formulas. G is said to be Γ-fuzzy consequence of F if for every interpretation such that $C_I(F) \geq \Gamma$, $C_I(G) \geq \Gamma$.

The significance of the threshold can be made clear by looking at some simple examples. Let $\Gamma = .7$ and consider each of the following three formulas: $.8A$, $.2A$, and $.6A$. Suppose A is 1; i.e., $I(A) = 1$ for some interpretation I. Then $\mathcal{C}_I(.8A) = .8 \geq \Gamma$, so the first formula is satisfied. The latter two evaluate to .2 and to .6, and so neither is satisfied. Now suppose A is 0. The first formula evaluates to $.8 \otimes 0 = .2$, and the second evaluates to $.2 \otimes 0 = .8$, so the second formula is Γ-satisfied. In effect, since the fuzzy operator .2 is less than $1 - \Gamma$, $.2A$ is a negative literal and is Γ-satisfied by assigning $false$ to the atom A. The value of the third formula is now $.6 \otimes 0 = .4$. Thus, in either case, the third formula is Γ-unsatisfiable. The authors of [41] in fact define a clause to be Γ-*empty* if every fuzzy operator of every literal in the clause lies between $1 - \Gamma$ and Γ; it is straightforward to prove that every Γ-empty clause is Γ-unsatisfiable.

The fuzzy resolution rule relies upon complementary pairs of literals. However, complementarity is a relative notion with respect to the threshold Γ. Two literals $\phi_1 A$ and $\phi_2 A$ are said to be to be Γ-*complementary* if $\phi_1 \leq 1 - \Gamma$ and $\phi_2 \geq \Gamma$. Fuzzy resolution can now be defined.

Definition. (Fuzzy Resolution) Let C_1 and C_2 be two clauses, and let L_1 and L_2 be Γ-complementary literals in C_1 and C_2, respectively. Let B_1 and B_2 be the two clauses such that $C_i = B_i \vee L_i, i = 1, 2$. Then the Γ-*resolvent* of C_1 and C_2 on the literals L_1 and L_2 is the clause $B_1 \vee B_2$.

In [41] the authors prove the appropriate results, including completeness, for fuzzy resolution.

5.3 Signed Inference with Fuzzy Operator Logics

Let Λ be the fuzzy operator MVL as developed, and let $\mathcal{F}$ be a formula in Λ. By Lemma 9, we may assume that all fuzzy operators in $\mathcal{F}$ are at the atomic level. (More precisely, we are assuming that 1 is the only non-atomic fuzzy operator.) We wish to analyze signed formulas $S:\mathcal{F}$, where S is any sign, i.e., where $S \subset [0, 1]$. Consider first $\{.5\}:\mathcal{F}$. For any Λ-consistent interpretation I_{Λ_S} (with corresponding interpretation I over Λ), $I_{\Lambda_S}(\{.5\} : \mathcal{F}) = true$ iff $I(\mathcal{F}) = .5$. By the observation before Lemma 9, the latter can occur only if some atom in $\mathcal{F}$ is assigned .5 by I or if some fuzzy operator in $\mathcal{F}$ is .5. Both of these conditions are trivial (at least uninteresting), so we shall assume that $.5 \notin S$. It turns out that the signs of interest in this section is also regular, where the definition of regularity is extended slightly to include signs are downsets or upsets or complements of downsets or of upsets. Since we are only considering signs that do not contain .5, the regular signs are positive (i.e., subsets of (.5, 1]) or negative (i.e., subsets of (.5, 1]).

To apply any of the inference techniques that have been developed for signed formulas, in particular signed resolution, requires driving all signs inward to the atomic level; that is, the inference rules for Λ_S operate on Λ-atomic formulas. Thus, we must determine how to drive positive and negative regular signs to the atomic level. There are three cases to consider: $\mathcal{F} = \mathcal{G} \vee \mathcal{H}$, $\mathcal{F} = \mathcal{G} \wedge \mathcal{H}$, and $\mathcal{F} = \phi A$, where A is an atom. (We need not consider the case $\mathcal{F} = \phi\mathcal{G}$ for an arbitrary $\mathcal{G}$ since we are assuming that the fuzzy operators are all at the atomic level.) The first two cases are handled by the next lemma.

Lemma 10. If S is positive and regular then

$$S:(\mathcal{G} \vee \mathcal{H}) \equiv_\Lambda S:\mathcal{G} \vee S:\mathcal{H} \quad \text{and} \quad \mathrm{S}:(\mathcal{G} \wedge \mathcal{H}) \equiv_\Lambda \mathrm{S}:\mathcal{G} \wedge \mathrm{S}:\mathcal{H},$$

and if S is negative and regular, then

$$S:(\mathcal{G} \vee \mathcal{H}) \equiv_\Lambda S:\mathcal{G} \wedge S:\mathcal{H} \quad \text{and} \quad \mathrm{S}:(\mathcal{G} \wedge \mathcal{H}) \equiv_\Lambda \mathrm{S}:\mathcal{G} \vee \mathrm{S}:\mathcal{H}.$$

Proof. First let S be positive, say $S = [\gamma, 1]$, and let I be an interpretation and I_{Λ_S} the corresponding Λ-consistent interpretation. Then $I(S:(\mathcal{G} \vee \mathcal{H})) = true$ iff $\gamma' = \mathrm{Sup}\{I(\mathcal{G}), I(\mathcal{H})\} \in S$ iff $\gamma' \geq \gamma$ iff $I(\mathcal{G}) \geq \gamma$ or $I(\mathcal{H}) \geq \gamma$ iff $I(S:\mathcal{G}) = true$ or $I(S:\mathcal{H}) = true$. The other cases are similar and left to the reader. □

To drive the sign S past the fuzzy operator ϕ in $S:(\phi A)$, again let $\phi = .5 + \phi'$; as before, we have

$$\Phi_\phi(\delta) = \phi \otimes \delta = .5 - \phi' + 2\phi'\delta.$$

If $\phi > .5$ (i.e., $\phi' > 0$), then we know that Φ_ϕ is monotonically increasing in δ. Let S be the regular positive sign $[\gamma, 1]$, and let $\Gamma = \Phi_\phi(\gamma)$. Since $\Phi_\phi(.5) = .5$, $\Gamma > .5$, and so $\Phi_\phi(S) = [\Gamma, \phi]$ is a positive sign. In general, unless $\phi = 1$, $\Phi_\phi(S)$ will not be regular. However,

$$\Phi_\phi^{-1}([\Gamma, 1]) = \Phi_\phi^{-1}([\Gamma, \phi]) = [\gamma, 1],$$

which of course is regular and positive, so we have

Lemma 11. Let S be a regular sign and let ϕ be a positive fuzzy operator. Then $\Phi_\phi^{-1}(S)$ is regular and positive or negative as S is positive or negative. If ϕ is negative, then $\Phi_\phi^{-1}(S)$ is regular and positive or negative as S is negative or positive. □

Lemma 12. Let A be an atom in Λ, let S be a (positive or negative) regular sign. Then

$$S:\phi A \equiv_\Lambda \Phi_\phi^{-1}(S):A\ .$$

Proof. If I is any interpretation over Λ with corresponding Λ-consistent interpretation I_{Λ_S}, let $\delta = I(A)$. Then $I_{\Lambda_S}(S:(\phi A)) = true$ iff $\phi \otimes \delta \in S$ iff for some $\gamma \in S$, $\gamma = \Phi_\phi(\delta)$ iff for some $\gamma \in S$, $\delta = \Phi_\phi^{-1}(\gamma)$. □

Observe that $\Phi_\phi^{-1}(S)$ will be empty unless there is at least one γ with $\gamma < \phi$ or $\phi < 1 - \gamma$ (in the case that S is positive; the reverse must hold if S is negative). In the case when it is empty, then of course $\Phi_\phi^{-1}(S):A \equiv_\Lambda false$.

The next theorem is now straightforward.

Theorem 9. Let $\mathcal{F}$ be a fuzzy formula, let S be a positive or negative regular sign, and let $\mathcal{F}'$ be the signed formula that is produced by repeated applications of Lemmas 9 and 10 to $S:\mathcal{F}$ until all fuzzy operators other than 1 and all signs are at the literal level. Then $S:\mathcal{F} \equiv_\Lambda \mathcal{G}$ where $\mathcal{G}$ is the formula obtained from $\mathcal{F}'$ by replacing each literal $S:\phi A$ in $\mathcal{F}'$ with $\Phi_\phi^{-1}(S):A$. □

For example, let $\Gamma = .7$, so that $S = [.7, 1]$, and consider the following signed atoms: $S:.8A$, $S:.25A$, and $S:.6A$. The corresponding Λ-atomic atoms are $[.33, 1]:A$, $[0, .1]:A$, and $\emptyset:A$. Notice that $.8A$ and $.25A$ are Γ-complementary, and that the signs of the corresponding signed atoms have an empty intersection. Moreover, the fuzzy literal $.6A$ is Γ-unsatisfiable, and the sign of the corresponding signed atom is empty. It is trivial to verify that these observations generalize to the next lemma.

Lemma 13. Given a threshold of acceptability Γ, let $S = [\Gamma, 1]$, let ϕA, $\phi_1 A$, and $\phi_2 A$ be fuzzy literals with $\phi_1 < \phi_2$, and let $S':A$, $S_1':A$, and $S_2':A$ be the signed atoms that correspond, respectively, to $S:\phi A$, $S:\phi_1 A$, and $S:\phi_2 A$. Then $1 - \Gamma < \phi < \Gamma$ iff $S' = \emptyset$. Moreover, $\phi_1 A$, and $\phi_2 A$ are Γ-complementary (i.e., $\phi_1 < \Gamma$ and $\phi_2 > \Gamma$) iff S_1' and S_2' are non-empty and $S_1' \cap S_2' = \emptyset$. □

The next theorem follows immediately from the last lemma.

Theorem 10. A deduction by fuzzy resolution is a special case of a deduction by signed resolution. □

Examples.

1. Let $C_1 = .2A \vee .9B, C_2 = .98A \vee .87A$, and let $\Gamma = .75$. Then

$$[\Gamma,1]\!:\!C_1 \equiv_\Lambda ([0,.083]\!:\!A) \vee ([.8125,1]\!:\!B) \ \text{ and } \ ([\Gamma,1]\!:\!\mathrm{C}_2) \equiv_\Lambda ([.76,1]\!:\!\mathrm{A}) \vee ([.838,1]\!:\!\mathrm{A}).$$

   Merging the two literals in the second clause and resolving yields $[.8125,1]\!:\!B$.

2. Let $C_1 = .11A \vee .43A \vee .19A$, $C_2 = .63B \vee .66A$, and let $\Gamma = .65$. Then

$$[\Gamma,1]\!:\!C_1 \equiv_\Lambda ([0,.31]\!:\!A) \vee (\emptyset\!:\!A) \vee ([0,.258]\!:\!A) \ \text{ and } \ [\Gamma,1]\!:\!\mathrm{C}_2 \equiv_\Lambda (\emptyset\!:\!\mathrm{B}) \vee ([.969,1]\!:\!\mathrm{A}).$$

   Note that the second literal of the first clause and the first literal of the second clause are signed with the empty set. Thus they may be removed without changing the models of the clauses, and we may resolve the other literals to yield the empty clause.

3. Let $C_1 = .1A \vee .2A$, $C_2 = .8A$, and $\Gamma = .7$. Then

$$[\Gamma,1]\!:\!C_1 \equiv_\Lambda ([0,.25]\!:\!A) \vee ([0,.167]\!:\!A) \ \text{ and } \ [\Gamma,1]\!:\!\mathrm{C}_2 \equiv_\Lambda [.83,1]\!:\!\mathrm{A}.$$

   Applying signed resolution produces the empty clause.

6 Discussion and Conclusion

There is an intuitive, but not very clearly understood connection between signed formulas and several other important research endeavors. The move from Λ to Λ_S is closely tied to the notion admitting propositions as objects. Such a formalism, commonly known as *reification*, plays an important role in AI – especially knowledge representation and common sense reasoning. The typical objects of investigation have been on reifying classical logic, modal logic, and temporal logic. In this context, signed formulas may be viewed as a formalization of reification over multiple-valued logics, where the connection between interpretations of the object-level logic and the reified logics are less obvious than two-valued logics. This is brought forth in the relationship that we established between Λ-interpretations and Λ-consistent interpretations over Λ_S.

*** signed formulas may also be embedded in annotated logics, such a translation enables paraconsistent reasoning with respect to both epistemic and ontological inconsistency.

*** meta-programming

References

[1] Anderson, R. and Bledsoe, W. W., A linear format for resolution with merging and a new technique for establishing completeness, *J.ACM*, 17(3) (1970), 525-534.

[2] Baaz, M., and Fermüller, C. G., Resolution for many-valued logics, *Proc. of Logic Programming and Automated Reasoning*, Lecture Notes in Artificial Intelligence 624, 107-118, Springer (1992).

[3] Blair, H.A. and Subrahmanian, V.S., Paraconsistent logic programming, *Theoretical Computer Science*, 68 (1989), 135-154.

[4] Carnielli, W.A., Systematization of finite many-valued logics through the method of tableaux. *Journal of Symbolic Logic*, 52,2 (1987), 473-493.

[5] da Costa, N.C.A., Henschen, L.J., Lu, J.J., and Subrahmanian, V.S., Automatic theorem proving in paraconsistent logics: theory and implementation, *Proc. of the 10th International Conference on Automated Deduction*, Lecture Notes in Computer Science (CADE 90), Springer, (1990).

[6] Davey, B.A., and Priestley, H.A., *Introduction to Lattices and Order*, Cambridge University Press (1990).

[7] Doherty, P., Preliminary report: NM3 — A three-valued non-monotonic formalism. *Proc. of the Fifth International Symposium on Methodologies for Intelligent Systems*, (1990). In: *Methodologies for Intelligent Systems*, 5 (Ras, Z., Zemankova, M., and Emrich, M. eds.) North-Holland, (1990), 498-505.

[8] Fitting, M., First-order modal tableaux, *J. Automated Reasoning*, 4, (1988) 191-213.

[9] Frühwirth, T., Herold, A,, Küchenhoff, V., Le Provost, T., Lim, P., Monfroy, E., Wallace, M., Constraint logic programming — an informal introduction, ECRC-93-5.

[10] Hähnle, R., Towards an efficient tableau proof procedure for multiple-valued logics, *Proc. of the Workshop on Computer Science Logic*, (1990), Springer, v. 533, 248-260.

[11] Hähnle, R., Uniform notation tableau rules for multiple-valued Logics, *Proc. of the International Symposium on Multiple-Valued Logic*, (1991), 26-29.

[12] Hähnle, R., *Automated Proof Search in Multiple-Valued Logics*, Oxford University Press (1993).

[13] Jaffar, J., Lassez, J-L, Constraint logic programming, in the Fourth IEEE Symposium on Logic Programming (1987).

[14] Kifer, M. and Lozinskii, E., RI: A logic for reasoning with inconsistency, *IEEE Symposium on Logic in Computer Science* (1989), 253-262.

[15] Kifer, M. and Lozinskii, E., A logic for reasoning with inconsistency, *Journal of Automated Reasoning* 9 (1992), 179-215.

[16] Kifer, M., and Subrahmanian, V. S., On the expressive power of annotated logics, *Proc. of the North American Conference on Logic Programming*, MIT Press (1989), 1069-1089.

[17] Kifer, M. and Subrahmanian, V.S., Theory of generalized annotated logic programming and its applications, the *Journal of Logic Programming* 12 (1992), 335-367.

[18] Leach, S., Lu, J.J., *Computing annotated logic programs: theory and implementation*, TR93-6 Bucknell University. Submitted.

[19] Lee, R., Fuzzy logic and the resolution principle. *J. ACM* 19 (1972), 109-119.

[20] Liu, X.H., Tsai, J.P., Weigert, T., Λ-resolution and the Interpretation of Λ-implication in Fuzzy Operator Logic, in: *Information Science*, 56, 1991, 259-278.

[21] Lloyd, J., *Foundations of Logic Programming*, Springer (1988).

[22] Loveland, D. W., A linear format for resolution, *Proc. IRIA Symp. on Automatic Demonstration*, Lecture Notes in Mathematics, Springer (1970), 147-162.

[23] Loveland, D. W., A unifying view of some linear Herbrand procedures, *J. ACM*, 19 (1972), 366-384.

[24] Lu, J.J., Henschen, L.J., Subrahmanian V.S., and da Costa, N.C.A., Reasoning in paraconsistent logics, *Automated Reasoning: Essays in Honor of Woody Bledsoe* (R. Boyer ed.), Kluwer Academic (1991), 181-210.

[25] Lu, J.J., Murray, N.V., Rosenthal, E., Signed Formulas and Annotated Logics, in: *Proc. of the 23rd IEEE International Symposium on Multiple-Valued Logic*, 1993, 48-53.

[26] Mukaidono, M., Fuzzy inference of resolution style, in *Fuzzy Set and Possibility Theory*, R. Yager (Ed.), Pergamon, NY (1982), 224-231.

[27] Murray, N.V. and Rosenthal, E., Inference with path resolution and semantic graphs. *J. ACM* 34,2 (1987), 225-254.

[28] Murray, N. V. and Rosenthal, E., Resolution and path dissolution in multiple-valued logics, *Proc. of the International Symposium on Methodologies for Intelligent Systems*, Springer v.542 (1991), 570-570.

[29] Murray, N.V. and Rosenthal, E., Improving tableaux proofs in multiple-valued logic, *Proc. of the 11 International Symposium on Multiple-Valued Logic* (1991), 230-237.

[30] Murray, N. V. and Rosenthal, E., Signed formulas: a classical approach to multiple-valued logics, TR91-12 (1991), SUNY at Albany. Presented at the 1992 summer meeting of the ASL.

[31] Murray, N. V. and Rosenthal, E., Dissolution: making paths vanish, *J. ACM*, 40, 2 (1993), 504-535.

[32] Murray, N. V. and Rosenthal, E., Adapting classical inference techniques to multiple-valued logics using signed formulas, *Fundamenta Informaticae Journal*, (to appear).

[33] Oppacher, F., and Suen, E., HARP: A tableau-based theorem prover, *Journal of Automated Reasoning* 4 (1988), 69-100.

[34] Priest, G., Reasoning about truth, *Journal of Artificial Intelligence*, 39 (1989), 231-244.

[35] Reed, D.W., Loveland, D.W., Near-Horn prolog and the ancestry family of procedures, presented at the symposium *Logic in Databases, Knowledge Representation and Reasoning* at the University of Maryland Institute for Advanced Computer Studies, Nov. 1993.

[36] Sandewall, E., The semantics of non-monotonic entailment defined using partial interpretations, in *Non-Monotonic Reasoning, 2nd International Workshop*, M. Ginsburg, M. Reinfrank, and E. Sandewall, ed., Springer (1988).

[37] Schwind, C., A tableau based theorem prover for a decidable subset of default logic. *Proc. of the 10th International Conference on Automated Deduction* (1990), Springer, v. 449, 528-542.

[38] Subrahmanian, V.S., Paraconsistent disjunctive databases, *Theoretical Computer Science*, 93 (1992), 115-141.

[39] Suchón, W., La méthode de Smullyan de construire le calcul n-valent de Lukasiewicz avec implication et négation. Reports on Mathematical Logic, Universities of Cracow and Katowice (1974) 2, 37-42.

[40] Surma, S. J., An algorithm for axiomatizing every finite logic. In *Computer Science and Multiple-Valued Logics*, David C. Rine, Ed., North-Holland, Amsterdam, 1984, 143-149.

[41] Weigert, T. J., Tsai, J.P., Liu, X.H., Fuzzy Operator Logic and Fuzzy Resolution, *J. Automated Reasoning* 10 (1993), 59-78.