Knowledge Representation in Many-Valued Horn Clauses

Barbara Messing
Institut für Angewandte Informatik und Formale Beschreibungsverfahren
D - 76128 Karlsruhe

messing@aifb.uni-karlsruhe.de
Tel.: +49 (0)721/608-4751

Keywords: knowledge representation, many-valued logics, uncertainty

Abstract

Uncertainty, incompleteness and vagueness are typical features of common-sense as well as expert knowledge. Reasoning based on such knowledge demands strategies to combine divergent information. All the more this is true in the context of integrating knowledge bases. This paper addresses knowledge representation, reasoning, and conflict management with many-valued logics. We show by some examples how information can be expressed in many-valued Horn clauses, how fitting truth value spaces can be found -in this context we also consider bilattices- and how conflict solving strategies can be involved. We do not focus on technical details but try to open up prospects of applications.

1 Introduction

One of the main efforts of AI is to model and formalize inexact knowledge: Uncertainty, incompleteness and vagueness are typical features of common-sense as well as expert knowledge. Reasoning based on such knowledge demands strategies to combine divergent information. For example, specific information should be preferred to more general knowledge, hard facts should replace default knowledge, and unsolvable contradictions should be made obvious.

All the more this is true in the context of integrating knowledge bases. This problem arises, if large knowledge bases must be built in teamwork, where each team is responsible for certain parts or aspects [Studer, Landes and Pirlein, 92]: Partial knowledge bases must be finally combined to one. A similar problem appears if knowledge bases are reused. If some parts of a knowledge base are actualized, they have to be integrated into the final system [Neches et al., 91]. There may also be several knowledge bases which represent knowledge of the same domain. A natural language understanding consulting system for instance may obtain knowledge extracted from different text sources [Klose and Luck, 91]. Generated answers should be based on information that is consistent with all the different knowledge bases.

In multiple-agent systems we are also concerned with the problem of conflicting information. Each agent possesses knowledge about the world and other agents, but has restricted facilities, tasks and competencies. There might be a global knowledge base shared by everyone but in general there is no central unit that decides what to do in case of conflicts.

This paper addresses knowledge representation, reasoning, and conflict management with many-valued logics. We show by some examples how information can be expressed in many-valued Horn clauses, how fitting truth value spaces can be found, and how conflict solving strategies can be involved. We do not focus on technical details but try to open up prospects of applications.

Many-valued logics offer a variety of possibilities to express uncertain, incomplete, temporal or fuzzy knowledge and enable us to reason in the presence of inconsistencies. In the last years, many-valued logics have been made more and more accessible to programming techniques. Two approaches can be handled similar to classical logic programming: Annotated logic has been developed by Kifer and Subrahmanian [Kifer and Subrahmanian, 92, Subrahmanian, 92], Subrahmanian, 94], signed logic has been developed independently by Murray [Murray and Rosenthal, 91, Murray and Rosenthal, 93], and Hähnle [Hähnle, 90, Hähnle, 91]. The connection between signed logic and annotated logic has been established in [Lu, Murray and Rosenthal, 93]. The authors have shown how annotated logic programs can be made amenable to a linear restriction rule, that performs both annotated resolution and reduction.

With the increasing computability many-valued logics get more and more attractive for AI-applications. According to the different kinds of inexact knowledge the truth values have to be chosen and combined. Uncertain knowledge like "perhaps she will stop smoking next week" requires a different handling than default knowledge like "birds usually fly" or fuzzy information ("he is a tall man"). The flexibility of many-valued logics lies in the independency of the truth value space. It can be selected according to the application domain.

Annotated logic programs base on lattice structures which are also intuitive for representing knowledge. Signed logic in general makes no restrictions to the structure of the truth value set, but lattice structures make signed logic more tractable for computing.

Bilattices were defined by Ginsberg [Ginsberg, 88], and further investigated by Fitting [Fitting, 92]. The bilattice-approach is a basic contribution to many-valued logics. Bilattices help to distinguish sorts of inexact knowledge and their combining according to the application. We should keep this structures in mind while modelling uncertainty. In this paper we give some examples to show how suggestive bilattices are in the context of knowledge representation and, especially, for the purpose of combining knowledge bases. These structures can be used by the knowledge engineer without having investigated the whole mathematical surrounding. We give here just the very basic definitions. In fact, we do not make use of all bilattice functions, but present them for completeness.

We have implemented a proof procedure called REGULA for many-valued Horn clauses. We will give a short description of it.

This paper is organized as follows: In section 2 we give an example to motivate the definitions of bilattices and the height in a bilattice. In section 3 we define many-valued Horn clauses and give a second example. Section 4 describes our proof procedure. The last section concludes the work and shows future perspectives.

2 Truth Value Spaces

In many-valued logics, propositions are attached with a truth value taken from a truth value space. A common truth value space is the unit interval [0,1] where 0 stands for "completely wrong" and 1 for "completely true". We give a first example to show how a second "dimension" in a truth value space can make modelling and reasoning more intuitive.

2.1 A first example

Suppose the department of computer science of a university has to decide about the subscription of journals. The number of journals is limited by the budget. The researchers working at the department have divergent interests and they are familiar with some topics while they do not know much about other fields. Now every researcher gets a list of journals and has to make some comments on it. These comments might by characterized as shown in figure 1. According to the degree of the acceptance the comments are evaluated to numerical values between -1 and 1.

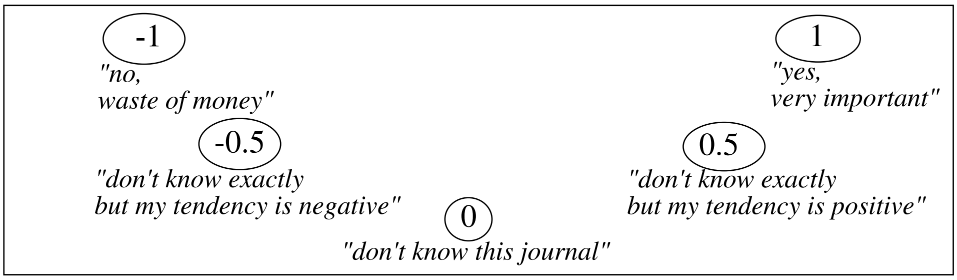

figure 1. Assessments of a journal

Suppose three researchers F_1, F_2, F_3 give the answers concerning the journals J_1,...,J_4 listed in the following table:

	J_1	J_2	J_3	J_4
$v(F_1,J_i)$	0.5	0	1	-1
$v(F_2,J_i)$	0.5	0.5	1	1
$v(F_3,J_i)$	0.5	0.5	-1	0.5
Σ	**1.5**	**1**	**1**	**0.5**

Simply adding the values yields the preference relation $J_1 > J_2 \sim J_3 > J_4$:

- J_1 will be subscribed first although there is nobody who really wants it.
- J_2 ranges higher than J_4, although there is someone who really wants J_4 but nobody who really wants J_2.
- J_2 and J_3 have the same range although there is more information and acceptance about J_3.

There are many situations in which a degree of information plays an important role for decision making. In fact, the values in this scenario have two dimensions: Familiarity with the topic and rejection/acceptance.

The above values can be embedded in a bilattice. The formal definition reads:

Definition 2.2 Bilattice

A ***bilattice*** *is a structure* $\mathcal{B} = (B, \wedge, \vee, \otimes, \oplus, \neg)$ *where B is a set of truth values, such that* $(B, \wedge, \vee)$ *and* $(B, \otimes, \oplus)$ *are complete lattices and* $\neg$ *is a function from B to B (called* ***negation****) with*

i) $\neg(\neg a) = a$ *for any truth value a,*
ii) $\neg$ *is a lattice homomorphism from* $(B, \wedge, \vee)$ *to* $(B, \vee, \wedge)$ *and*
iii) $\neg$ *is a lattice homomorphism from* $(B, \otimes, \oplus)$ *to* $(B, \otimes, \oplus)$.

The partial orderings induced by $\wedge$, $\vee$, $\otimes$, $\oplus$ *we denote as follows:*
$x \leq_k y :\Leftrightarrow x \otimes y = x$ *(and hence* $x \oplus y = y$*)*, $x \leq_t y :\Leftrightarrow x \wedge y = x$ *(and hence* $x \vee y = y$*).*

In a bilattice, the truth values differ in their degree of truth ("true" is "more true" than "false") and in their degree of knowledge (to know that a fact is "false" is more information than to know that it is "default-true"). The smallest non-trivial bilattice is the well-known lattice of four-valued logic FOUR, first introduced by Belnap [Belnap, 77]. $\otimes$ has been called consensus-operator, $\oplus$ gullible-operator [Fitting, 91]. These operators match two values wrt their degree of information.

The values of example 2.1 can be structured in D_1, (the reader who is familiar with bilattices will realize the bilattice of default reasoning).

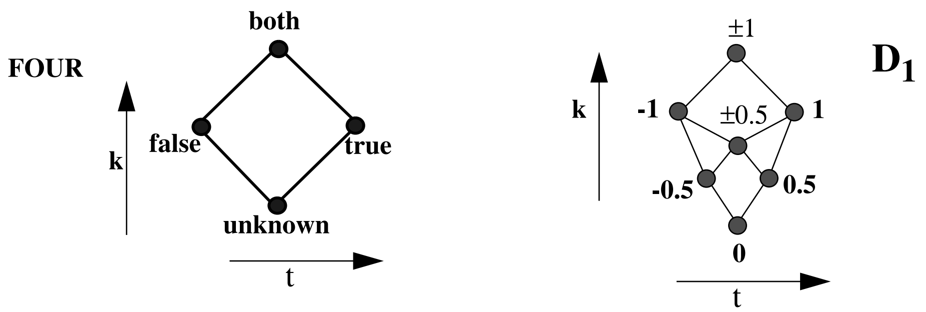

figure 2. Bilattices

The k-height in a bilattice measures the degree of information by a numerical value. We'll apply it in the following example.

Definition 2.3 Height

Let $\mathcal{B} = (B, \wedge, \vee, \otimes, \oplus, \neg)$ *be a bilattice with bottom element* $\perp$ *and top element* $\top$ *wrt* $\leq_k$, $x \in B$.
A k-chain between x_0 *and* x_n *is a sequence* $x_0, x_1, \ldots, x_n$ *with* $x_0 \leq_k x_1 \leq_k \ldots \leq_k x_n$.
$h_k(x) := \mathrm{lub}\,\{n \mid \perp, x_1, \ldots, x_{n-1}, x$ *k-chain between* $\perp$ *and* $x\}$ *is called* ***k-height*** *of x,*
$h_k(B) := h_k(\top)$ *is called the* ***k-height*** *of* $\mathcal{B}$.
$\rho(x) := h_k(x)/h_k(B)$ *is called the* ***relative k-height*** *of x, provided the k-height of* $\mathcal{B}$ *is finite.*

Example: The k-height of D_1 is 4, and $\rho(-1) = 3/4 = 0.75$.

2.4 The first example continued

We define a decision function ϕ that combines the grade of knowledge with the grade of acceptance. The values $v(F_j,J)$ are summed up and then multiplied with the degree of information gathered by $v(F_j,J)$. That is,

$$\phi(J, F_1, F_2, F_3) := \left(\Sigma_{j=1,2,3}\, v(F_j, J)\right) * \rho\left(\bigoplus_{j=1,2,3} v(F_j, J)\right)$$

The preference relation now is $J_3 > J_4 > J_1 > J_2$ (see the following table): J_3 will be subscribed first. There are two researchers that need this journal so this is a reasonable decision. J_4 ranges higher than J_2 which is also intuitive because there is more knowledge and more need of J_4 than of J_2.

There is a variety of other decision functions that can be constructed in a similar way. It depends on the given situation how to do it.

	J_1	J_2	J_3	J_4
$v(F_1,J_i)$	0.5	0	1	-1
$v(F_2,J_i)$	0.5	0.5	1	1
$v(F_3,J_i)$	0.5	0.5	-1	0.5
Σ	1.5	1	1	0.5
$\oplus$	0.5	0.5	±1	±1
ϕ	0.37	0.25	1	0.5

3 Many-valued Horn Clauses

We do not concentrate on technical details but try to make things clear by some examples. We assume a first-order language as usual.

Definition 3.1 Many-valued Horn Clauses, semantics

Many-valued Horn clauses *are rules* $P\langle\mu\rangle \leftarrow P_1\langle\mu_1\rangle,...,P_n\langle\mu_n\rangle$, *where* $P, P_1,...,P_n$ *are first-order literals,* $\mu, \mu_1,...,\mu_n$ *are elements of a complete lattice* Δ *of truth values, or variables, or terms. A set of many-valued Horn clauses is called* ***many-valued program***, *denoted by* $(\mathbf{P}, \Delta)$.
An interpretation I from the ground atoms to Δ ***satisfies*** *a signed ground atom* $P\langle\mu\rangle$ *(P being ground,* μ *an element of the truth value space) iff* $I(P) \geq \mu$. *An interpretation is a* ***model*** *of a clause* $P\langle\mu\rangle \leftarrow P_1\langle\mu_1\rangle,...,P_n\langle\mu_n\rangle$ *iff* $I(P_1) \geq \mu_1,..., I(P_n) \geq \mu_n$ *implies* $I(P) \geq \mu$. *An interpretation is a* ***model of a program*** *iff I is a model of each clause.*

3.2 Example

For example, consider the program

(1) drinks_beer(X)<0.5> ← *bavarian(X)*<1>
(2) drinks_beer(X)<–0.5>←*taxidriver(X)*<1>
(3) bavarian(peter)<1>←
(4) bavarian(lucy)<1>←
(5) taxidriver(lucy)<1>←

where the truth values are the elements of D_1 as lattice wrt the k-ordering.
Consider the interpretations I_1, I_2 listed in the following table:

	I_1	I_2		I_1	I_2
bavarian(peter)	1	1	*taxi-driver(lucy)*	1	1
bavarian(lucy)	1	1	*drinks_beer(peter)*	0.5	1
taxi-driver(peter)	±1	0	*drinks_beer(lucy)*	-0.5	±0.5

I_1 is not a model of the program because I_1(*taxi-driver(peter)*)= ±1 $\geq_k$1 but I_1(*drinks_beer(peter)*) = 0.5 and 0.5 $\geq_k$ -0.5 does not hold, so clause (2) is not satisfied.

I_2 is a model of the program. For example, I_2 (*bavarian(peter)*)=1 $\geq_k$ 1 and I_2 (*drinks_beer(peter)*) = 1 $\geq_k$ 0.5, so clause (1) is satisfied for X = *peter.*

Interpretations that map every instance to the top element of the truth value space are always models. On the other hand, if any head of the program clauses has the form L<⊥>, every interpretation is a model.

Many-valued Horn clauses can be treated similar to usual Horn clause programs, especially if the truth value space is finite. In the following we give a more detailed example.

3.3 A Second Example

Suppose a program committee has to decide about the acceptance or rejection of papers. There are three referees for a paper; they have different foreknowledge, so they assess their competence with five values between -1 and 1 (ordered in the usual way and therefore forming a complete lattice). The program committee has some guidelines for the decision about the contributions. Suppose it has to judge about readability then it could use rules like

- A paper is readable, if someone not familiar with the topic finds it easy to read.
- A paper is not readable, if someone familiar with the topic finds it hard to read.
- In other cases, agreements of referees are decisive.

In many-valued Horn clauses, these rules read as follows:

A paper X is readable, if someone not familiar with the topic finds it easy to read.

readable(X)<1> ← *reads(R,X)<1>,comp(R,X)<-1>*

A paper X is not readable, if someone familiar with the topic finds it hard to read.

readable(X)<-1> ← *reads(R,X)<-1>,comp(R,X)<1>*

In other cases, agreements of referees are decisive.

readable(X)<μ> ← *reads(R1,X)<μ>, reads(R2,X)<μ>,* \=*(R1,R2)*

Here \= is (as usual) a binary built-in predicate that is satisfied if the arguments are not equal.

Together with the statements of three referees r, s, t concerning the paper α and competence factors we get the following program:

(1) *comp(r, α)<1>*
(2) *comp(s, α)<-1>*
(3) *comp(t, α)<0.5>*
(4) *reads(r,α)<-1>*
(5) *reads(s,α)<0.5>*
(6) *reads(t,α)<0.5>*
(7) *readable(X)<1>* ← *reads(R,X)<1>,comp(R,X)<-1>*
(8) *readable(X)<-1>* ← *reads(R,X)<-1>,comp(R,X)<1>*
(9) *readable(X)<μ>* ← *reads(R1,X)<μ>, reads(R2,X)<μ>,* \=*(R1,R2)*

Now, what about the readability of α? From (1), (4) and (8) it follows

readable(α)<-1>.

From (5), (6) and (9) it follows

readable(α)<0.5>

A model of this program (if based on [-1, -0.5, 0, 0.5, 1]) would map readable(α) to 0.5 or a greater value. This is not very intuitive. The disagreement between the referees is neglected.

The above values can be taken as the nodes of the lower edges in the lattice NINE shown in figure 3. NINE can be taken as a finite subset of the bilattice T(I,I) (we changed the notation of the elements just for simplicity):

Definition 3.4 T(I,I)

The bilattice T(I,I) consists of pairs (x,y) where $x, y \in I = [0,1]$ *and*
$(x,y) \leq_k (x',y') :\Leftrightarrow x \leq x' \wedge y \leq y'$,
$(x,y) \leq_t (x',y') :\Leftrightarrow x \leq x' \wedge y' \leq y$,
$\neg(x,y) = (y,x)$.

The elements of T(I,I) are pairs (X,Y) that can be taken as two-dimensional truth values: X refers to what is evidential for a fact, Y to what is against.

Now, returning to the example, a model has to evaluate readable(α) to *-0.5 or a k-greater value so we can conclude from the result that there are divergent opinions about the readability of the paper.

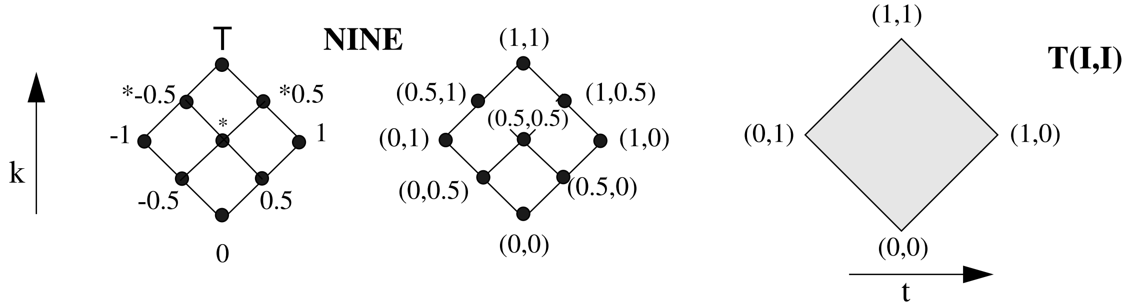

figure 3. T(I,I) and finite subsets

Another remark on T(I,I): It has been proposed to use a pair (X,Y) with X being the *necessity* and Y being the *possibility* of a fact [Baldwin, 87, Zhang, 92]. This interpretation can be transformed to the above given if we take into consideration that **A** is possible as far as **not A** is necessary.

4 R&R-Resolution: A proof procedure for many-valued Horn clauses

R&R-resolution is essentially based on signed resolution. A *signed* clause has the form $\{S_1{:}A_1,...,S_n{:}A_n\}$, where $S_1,...,S_n$ are subsets of an arbitrary set Δ of truth values and $A_1..., A_n$ are first-order atomic formulas. Signed resolution forms the intersection of the signs: For example, if $\Delta = [0,1]$, the signed resolvent of *{[0,0.7]:A, [0,0.5]:B}* and *{[0.5,1]:A, [0.4,1]:C}* is *{[0.5,0.7]:A, [0,0.5]:B, [0.4,1]:C}*. Many-valued Horn clauses are special cases of signed clauses if the signs in many-valued clauses are taken as so-called upsets in an partially ordered truth value space.

Definition 4.1 R&R - Resolution

*Let (**P** ,* Δ*) be a many-valued program. A clause* $\leftarrow Q_1\langle\mu_1\rangle,...,Q_n\langle\mu_n\rangle$ *is called a **query**.*
A clause $Q\langle\langle\sigma\rangle\rangle \leftarrow Q_1\langle\mu_1\rangle,...,Q_n\langle\mu_n\rangle$*, where* σ *is an **arbitrary set** of truth values is called a **proof query**.*

$\uparrow\mu := \{\delta \in \Delta \,;\, \mu \leq \delta\}$ *is called the **upset** of* μ. M^c *denotes the complement of a set M.*

i) Let $P\langle\nu\rangle \leftarrow P_1\langle\nu_1\rangle,...,P_m\langle\nu_m\rangle$ *be a program clause that doesn't share any variables with the query* $\leftarrow Q_1\langle\mu_1\rangle,...,Q_n\langle\mu_n\rangle$ *and P and* Q_1 *are unifiable via mgu* Θ.

- *If* $(\uparrow\nu) \cap (\uparrow\mu_1)^c = \emptyset$*, the query* $(\leftarrow Q_2\langle\mu_2\rangle,...,Q_n\langle\mu_n\rangle,P_1\langle\nu_1\rangle,...,P_m\langle\nu_m\rangle)\,\Theta$ *is the R&R-resolvent of the given program clause and the query.*
- *If* $(\uparrow\nu) \cap (\uparrow\mu_1)^c = \sigma \neq \emptyset$*, the proof query* $(Q_1\langle\langle\sigma\rangle\rangle \leftarrow Q_2\langle\mu_2\rangle,...,Q_n\langle\mu_n\rangle, P_1\langle\nu_1\rangle,...,P_m\langle\nu_m\rangle)\,\Theta$ *is the R&R-resolvent of the given program clause and the query.*

ii) Let $P\langle\nu\rangle \leftarrow P_1\langle\nu_1\rangle,...,P_m\langle\nu_m\rangle$ *be a program clause that doesn't share any variables with the proof query* $Q\langle\langle\sigma\rangle\rangle\leftarrow Q_1\langle\mu_1\rangle,...,Q_n\langle\mu_n\rangle$ *and P and Q are unifiable via mgu* Θ.

- *If* $\sigma \cap (\uparrow\nu) =\emptyset$, *the query* $\leftarrow (Q_1\langle\mu_1\rangle,...,Q_n\langle\mu_n\rangle, P_1\langle\nu_1\rangle,...,P_n\langle\nu_m\rangle)\ \Theta$ *is the R&R-resolvent of the given program clause and the proof query.*
- *If* $\sigma \cap(\uparrow\nu) = \tau \neq \emptyset$, *the proof query* $(Q\langle\langle\tau\rangle\rangle \leftarrow Q_1\langle\mu_1\rangle,...,Q_n\langle\mu_n\rangle, P_1\langle\nu_1\rangle,...,P_m\langle\nu_m\rangle)\ \Theta$ *is the R&R-resolvent of the given program clause and the proof query.*

The complement of an upset must not to be confused with a "downset" $\downarrow\mu :=\{\delta \in \Delta\ ;\ \mu \geq \delta\}$. The above definition is a generalization of regularity as given in [Hähnle, 95] for totally ordered sets to lattice orderings.

The completeness of R&R-resolution for distributive truth value spaces that fulfill the descending chain condition follows from the results of [Lu, Murray and Rosenthal, 93] and is shown in [Messing, 95]. The notation R&R-resolution is a shortcut for "resolution and reduction": R&R-resolution performs annotated resolution and reduction. In [Lu, Murray and Rosenthal, 93] it has been shown how both of them can be subsumed to signed resolution.

R&R-resolution allows to build proof trees likely to SLD-resolution. In figure 4 we show the proof tree for the query $\leftarrow$*readable*$(\alpha)\langle *\text{-}0.5\rangle$ of the above example.

There are different possibilities to instantiate μ in (9). In the successful path in the proof tree it is the least element such that the intersection is empty.

4.2 REGULA

We have developed the program REGULA that performs R&R - resolution [Stackelberg, 95]. REGULA contains already some strategies to restrict the search space, which can be enormous especially if variables and functions are involved. REGULA then uses so-called minimal valuations, that is, the variables of a program clause are instantiated with minimal values wrt the query.

For example, in figure 4, when resolving *readable*$(\alpha)\langle|\{-1\}|\rangle\leftarrow$ *reads*$(R,\alpha)\langle -1\rangle$,*comp*$(R,\alpha)\langle 1\rangle$ with clause (9), μ=0.5 is the minimal value to get $\{-1\}\cap\uparrow 0.5 = \emptyset$. This can be generalized to the case that functions occur as sign in a query. On the other hand, if queries contain, say, "both" of FOUR, this value is decomposed to "true" and "false" and then both queries are processed.

5 Related Work, Conclusion, and Outlook

In [Dubois and Prade, 94], defaults of the form $p\rightarrow q$ are encoded in a possibilistic framework as a rule $(p\rightarrow q, \alpha)$ which reads, roughly speaking, as "$p\wedge q$ is with a value α more possible than $p\wedge\neg q$". The rules for evidential reasoning [Baldwin87]: $P \leftarrow P_1,...,P_n\langle\mu\rangle$ and other approaches of uncertainty reasoning have the same form.

The essential difference annotated/signed logics make is that the value is attached to each single literal and not to a rule. This is a more general approach and enables us to express more detailed information.

In [Leach and Lu, 94] ca-resolution is developed for annotated programs. The authors use constraints to produce regular proofs (and therefore can profit from constraint logic programming techniques). But processing the query from left to right causes incompleteness in ca-resolution. R&R-resolution is in this respect more close to SLD-resolution.

In this contribution we have shown how knowledge can be expressed in many-valued Horn clauses and proposed our proof procedure R&R-resolution. This opens up prospects for various applications.

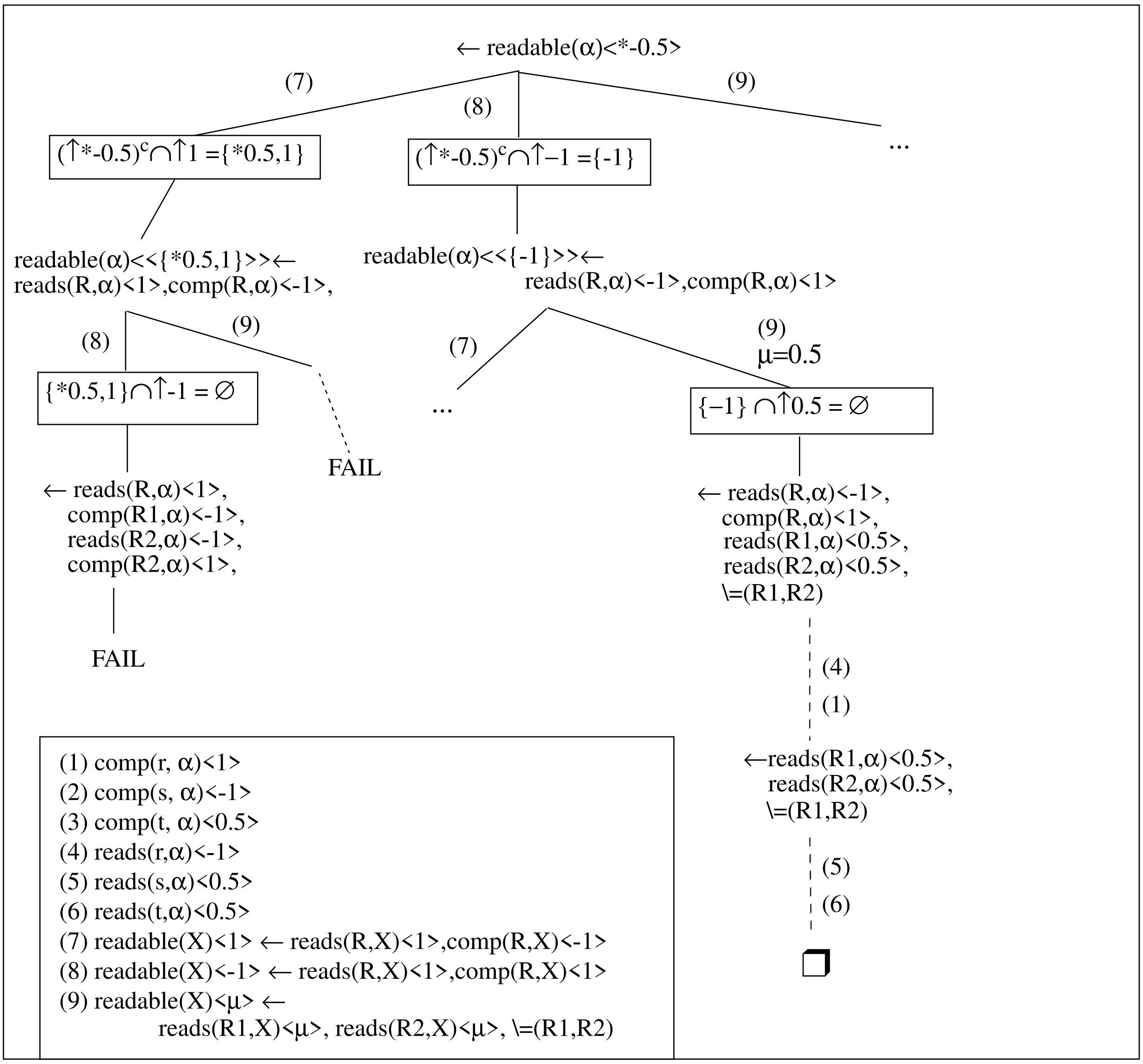

figure 4. A R&R - proof tree

6 References

[Baldwin, 87] Baldwin J.W.: "Evidential Support Logic Programming". Fuzzy Sets and Systems 24, 1-26. North Holland,1987.

[Belnap, 77] Belnap N.D., jr.: "A useful four-valued logic". J.M. Dunn and G. epstein (eds.) Modern uses of Multiple-valued logics. Reidel, 8-37, 1977.

[Dubois and Prade, 94] Dubois D. Prade H.: "Expressing Independence in a Possibilistic Framework and its Application to Default Reasoning". A. Cohn (ed.): Proceedings of the 11th European Conference on Artificial Intelligence ECAI 94. John Wiley&Sons, Ltd, 150-154, 1994.

[Fitting, 91] Fitting M.: "Bilattices and the semantics of logic programming". J. Logic Programming 11, 91-116, 1991.

[Ginsberg, 88] Ginsberg M.: " Multivalued logics". Computational Intelligence 4(3), 1988.

[Hähnle, 90] Hähnle R.: " Towards an efficient tableau proof procedure for multiple-valued logic". Proceedings of the workshop on Computer Science Logic, Heidelberg,1990. LNCS, Springer, 553, 248-260, 1990.

[Hähnle, 91] Hähnle R. : " Uniform notation tableau rules for multiple-valued logics". Proc. of the International Symposium on Multiple-Valued Logics, 26-29, 1991.

[Hähnle, 95] Hähnle R.: " Exploiting Data Dependencies in Many-Valued Logics". To appear in Journal of Applied Non-Classical Logics, 1995.

[Klose and Luck, 91] Klose G., Luck K.: " The background knowledge of the LILOG system". O.Herzog,C.-R.Rollinger (Eds.). Text Understanding in LILOG. Lecture Notes in Artificial Intelligence 546, Springer 1991.

[Kifer and Subrahmanian, 92] Kifer M., Subrahmanian V.S.: "Theory of Generalized Annotated Logic Programming and its Applications". Journal of Logic Programming 12, 335-367, 1992.

[Leach and Lu, 94] Leach,S.M., Lu J.J.: " Computing Annotated Logic". Proceedings of the Eleventh International Conference on Logic Programming. MIT Press, Massachusetts, 257-271, 1994.

[Lu, Murray and Rosenthal, 93] Lu J.J. , Murray N.M. , Rosenthal E.: " Signed Formulas and Annotated Logics". Proceedings of the 23th International Symposium on Multiple-valued Logics, 48 - 53. IEEE, 1993.

[Messing, 95] Messing B.: " REGULA: Knowledge Representation and Processing in Many-valued Logics". Research Report No 320 of the Institut für Angewandte Informatik und Formale Beschreibungsverfahren, University of Karlsruhe, 1995.

[Murray and Rosenthal, 91] Murray N.V. , Rosenthal E.: ". Resolution and path dissolution in multiple-valued logics". Proc. of the International Symposium on Methodologies for Intelligent Systems. Springer v. 542, 570-579, 1991.

[Murray and Rosenthal, 93] Murray N.V. , Rosenthal E.: " Signed formulas: A liftable Meta-Logic for Multiple-Valued Logic". Proc. of the International Symposium on Methodologies for Intelligent Systems. Springer v.689, , 275-284, 1993.

[Neches et. al., 91] Neches R., Fikes R., Finnin T.,. Gruber T, Patil R., Senator T., Swartout W.R.: " Enabling technology for knowledge sharing". AI magazine 12(3), 1991.

[Stackelberg, 95] Stackelberg P.: " Modellierung des Risikoprofils für Privatkunden von versicherungen in annotierter Logik" (in German). Diploma Thesis, Universität Karlsruhe,1995.

[Subrahmanian, 92] Subrahmanian V.S. : " Paraconsistent disjunctive deductive databases". Theoretical Computer Science 93, 115-141, 1992.

[Subrahmanian, 94] Subrahmanian V.S. : " Amalgamating Knowledge Bases". ACM Transactions on Database Systems, 19, 291-331, 1994.

[Studer, Landes and Pirlein, 92] Studer R., Landes D., Pirlein T.: "Knowledge engineering and knowledge representation for natural language understanding systems". R. Meersman, R. van de Riet (eds.): Linguistic Instruments in Knowledge Engineering, North Holland Publ. Co.,1992.

[Zhang, 92] Zhang C.: " Cooperating under uncertainty in distributed expert systems". Artificial Intelligence 56 (1992). Elsevier Science Publishers.