PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON FIFTH GENERATION COMPUTER SYSTEMS 1988,
edited by ICOT.

RULES AND JUSTIFICATIONS: A UNIFORM APPROACH TO REASON MAINTENANCE AND NON-MONOTONIC INFERENCE

Michael Reinfrank[1], Hartmut Freitag

Advanced Reasoning Methods Group
ZT ZTI INF 22, SIEMENS AG
8000 Muenchen 83, WEST-GERMANY
e-mail: {reinfra, freitag}@ztivax.siemens.com

ABSTRACT

We give a proof-theoretic characterization of reason maintenance systems (RMS) as realizations of finite non-monotonic formal systems (NMFS). NMFS are built on the concept of a valid proof relative to an overall coherent set of beliefs, and the justifications of an RMS are seen as rules used to construct such proofs. We introduce NMFS with variables as a means to describe a certain class of rule-based non-monotonic inference systems, and thus provide a uniform framework both for an RMS and a problem solver that uses it.

Keywords: reason maintenance systems, non-monotonic reasoning

1 INTRODUCTION AND OVERVIEW

The task of a reason maintenance system (RMS) is to maintain coherent belief states for a domain dependent problem solver (PS). The PS communicates to the RMS the results of his inferences in terms of formulae of some, not necessarily, logical representation language, and justifications for derived beliefs. If the PS makes non-monotonic inferences these justifications have the general form "belief in $p_1,...,p_m$ and disbelief in $q_1,...,q_n$ justifies belief in r", where the p's, q's, and r are formulae of the representation language[2].

The RMS then in turn uses this information to determine sets of belief that meet the coherence requirement, that for every member of the set, there is some valid non-circular argument using the available justifications. It is important to note that the RMS considers the PS-formulae as symbolic propositional entities without any further internal structure. Most RMS also support some notion of consistency using contradiction nodes and corresponding contradiction handling routines that try to avoid inconsistent states.

Existing formal descriptions of reason maintenance usually focus either on what it might mean from a logical perspective ([1] is the root node of this class of approaches, see e.g. [2] for one of its most elaborate representatives), or on the algorithmic aspects of reason maintenance, often formulated in terms of constraint labeling procedures, such as e.g. [3] and [4].

Common to all of these approaches is that they generally make only very weak assumptions about

---

[1] The major part of the work reported in this paper was carried out during a stay of the first author at the Laboratory for the Representation of Knowledge in Logic, Department of Computer and Information Science, University of Linkoeping, 58183 Linkoeping, SWEDEN.

[2] Some RMS only use monotonic justifications, but then necessarily need some sort of contradiction handling and premise control facility to support non-monotonic inference.

the problem solver, which - if at all - is described in procedural terms, and thus does not easily lend itself to any systematic analysis. Since RMS are inherently finite and propositional, this leaves in particular the handling of variables, which is often an important part of the whole problem solving process, outside the formal description.

We suggest a uniform approach to describe both an RMS and the PS on top of it. To this end, we first develop a more functionally oriented description of reason maintenance that focuses on the role an RMS plays for a PS, as a complement to the logical and algorithmic approaches mentioned above.

We introduce non-monotonic formal systems (NMFS), a simple formalism which is built upon the core concept of valid proof from a set of base facts, relative to an overall coherent state of beliefs. The justifications of the RMS then are considered as rules being used to construct such proofs, i.e. the process of reason maintenance can be regarded as performing non-monotonic inference in a formal calculus in order to determine what can and should be believed in a certain problem solving state.

We then extend the NMFS-framework to also allow for rules with variables, which are used to represent the specific domain knowledge of a PS. Such a rule essentially plays the role of a parametrized justification, and the connection to the underlying RMS then is to identify the rule with the set of its corresponding ground instantiated justifications, the variables ranging over a suitable universe of constants.

This way we achieve the following overall perspective of the PS/RMS interactions. The PS-rules describe an implicit network of justifications, and firing a (ground instantiated) rule simply means to make it explicit to the underlying RMS, which in turn determines an approriate set of explicit beliefs the PS should be aware of, in order to decide which rules to consider next.

Finally, we discuss some connections of our approach to work going on in the area of logic programming with negation, and to default logic, thus demonstrating that it opens the way for a theoretical analysis of the properties of RMS-based problem solvers.

2 NON-MONOTONIC FORMAL SYSTEMS

2.1 Basic Definitions

We assume that belief states are represented as finite subsets of a possibly (countably) infinite universe L. In principle, L may consist of the well-formed expressions of any formal representation language if we consider reason maintenance in general, but we will restrict our considerations to a simple logic language to be defined later.

A **non-monotonic formal system** (NMFS) over L then is a set R of rules of the form $<A|B\rightarrow c>$, where A and B are finite subsets of L, and c is a singleton. We call the elements of A and B the monotonic and non-monotonic antecedents of the rule, respectively, and c its conclusion. (When enumerating A and B, we omit the curly set brackets.) Unless specified otherwise, we will always assume R to be finite as well.

The elements of R are considered as rules to construct non-circular arguments for beliefs within a given state $S\subseteq L$, starting from some finite base set P of premise beliefs that need no further justification. A rule $<A|B\rightarrow c>$ can contribute to a non-circular argument for c in S, provided there are such arguments available for all $a\in A$ and for no $b\in B$. Formally, we define an **NM-proof** for q from P valid in S as a finite sequence $(p_1,...,p_n)$, where:

- $\forall i: p_i\in S$

- $p_n = q$

- $\forall i: p_i\in P$ or $\exists <A|B\rightarrow p_i>\in R$

s.t. $A \subseteq \{p_1,...,p_{i-1}\}$ and $B \cap S = \{\}$

Note that the property of being a proof depends not only on P and R but also on S, and that it is non-monotonic w.r.t. S.

A superset $S \supseteq P$ is called an **admissible extension**[1] of P iff it satisfies the following two conditions:

S is **grounded** in P, i.e. for every $q \in S$ there is an NM-proof for q from P valid in S

S is **closed** (w.r.t. R), i.e. for every $<A|B \rightarrow c> \in R$, we have if $A \subseteq S$ and $B \cap S = \{\}$ then $c \in S$

As usual for non-monotonic calculi, there are combinations of P and R s.t. P has no, exactly one or several admissible extensions; we will come back to that issue in section 5.

2.2 Some Results and Examples

The groundedness property directly reflects our intuition on the existence of a non-circular argument. To see that the closure property is also important, consider the NMFS $\{<a|b \rightarrow c>, <a| \rightarrow b>\}$, together with a single premise $P = \{a\}$. The set {a,c} is grounded in P, but not closed, since there is some valid reason to believe b as well, which would render the NM-proof (a,c) for c invalid in {a,b,c}. The set {a,b} is the only admissible extension of P here.

This example also shows that maximal grounded sets such as {a,c} are not necessarily closed. Conversely, minimal closed sets are not necessarily grounded, as the following example shows: $P = \{\}$, $R = \{<a| \rightarrow b>, <b| \rightarrow a>, <|b \rightarrow c>\}$. Obviously, {a,b} is a minimal closed set, but it is not grounded.The result here is even stronger: note that {a,b} is in a sense locally grounded because both a and b are conclusions of rules all of whose monotonic antecedents are in {a,b}, and none of their non-monotonic antecedents are so. Since {a,b} is also miniminal in that respect, we cannot reduce the global property of being grounded to its local equivalent. (McDermott's definition of "well-founded" given in [6], though, would correspond to minimal locally grounded and closed sets.)

In the case of multiple admissible extensions, one might be tempted to try and define a unique sort of theorem-hood ("q follows from P") by either of (1) q is in every admissible extension or (2) q is in at least one admissible extension. The following example shows however, that neither the intersection nor the union of admissible extensions have the desired properties. The NMFS $R = \{<|a \rightarrow b>, <|b \rightarrow a>, <a \rightarrow c>, <b \rightarrow c>\}$ induces two admissible extensions {a,c} and {b,c} of the empty premise set, but {a,b,c} is not grounded and {c} is not closed, as could be expected. The particularly discouraging result is that the intersection is even not grounded. So there is no obvious way to abstract from the existence of a valid proof for beliefs being justified relative to an overall coherent state.

So far, admissible extensions are defined in terms of static conditions upon belief states. The following theorem gives a sort of pseudo-constructive equivalent characterization of admissible extensions. The theorem intuitively says that, if we already knew such an admissible extension we could re-construct it incrementally, and it gives us a way to check candidate sets for their being admissible extensions. Actual construction algorithms of course use different methods, see e.g. [4] and [7].

Given an NMFS R, a premise set P, and a candidate extension S, define a sequence of sets T_i as follows:

$T_0 = P$

---

[1] The terminology is partly borrowed from [5].

$T_{i+1} = T_i \cup \{c \mid \exists <A|B \rightarrow c> \in R$
s.t. $A \subseteq T_i$ and $B \cap S = \{\}\}$

Theorem: $S = T_0 \cup T_1 \cup ... \cup T_\infty$ iff
S is an admissible extension of P.

One immediate consequence of the reconstruction theorem for a monotonic NMFS R, i.e. one where no rule has any non-monotonic antecedents, is that every premise set P has one and only one admissible extension, which we denote by Ext(P).

2.3 Rules with Variables

We now proceed to define a simple predicate language in order to be able to represent PS beliefs and rules. We start from a finite vocabulary that consists of a number of relation symbols $REL = \{P_1, P_2, ..., P_k\}$, having some defined arities, and a number of object constants $OBJ = \{a_1, a_2, ..., a_l\}$. Additionally, we use a countable set $VAR = \{x_1, x_2, ...\}$ of variables. In this basic version, we do not consider any quantifiers or connectives. Let

$L = \{P(a_1,...,a_n) \mid P \in REL$, with arity n, and
$a_1,...,a_n \in OBJ\}$

$Lx = \{P(\alpha_1,...,\alpha_n) \mid P \in REL$, with arity n,
$\alpha_1,...\alpha_n \in OBJ \cup VAR\}$

Our universe L of beliefs consists of all ground atoms that result from combining a relation symbol with an approriate number of constants. As before, we consider formal systems over L, but we now will also allow for rule-schemata ranging over Lx, for the purpose of specifying such an NMFS.

A rule schema has the form $[A|B \rightarrow c]$, where $A, B \subseteq Lx$, and $c \in Lx$, and we use the different syntactical notation to distinguish it from a proper rule. We consider such a schema as a description of all of its ground instances that can be achieved by consistently replacing every variable in the schema by an object constant.

Formally, if v is a ground instantiation defined in the usual way for atomic formulae, we define

$v([p_1,...,p_m|q_1,...,q_n \rightarrow r])$
$= <v(p_1),...,v(p_m)|v(q_1),...v(q_n) \rightarrow v(r)>$

and let a schema $[A|B \rightarrow c]$ denote the following set of rules over L:

$\{<A'|B' \rightarrow c'> \mid <A'|B' \rightarrow c'> = v([A|B \rightarrow c])$, for some ground instantiation v$\}$

A finite set of schemata then defines an NMFS in the obvious way. Clearly, this NMFS is also finite, as long as our vocabulary is so (and contains no function symbols, of course).

As an example, consider the well-known paradigmatic "usually, birds fly"-case.

Vocabulary: REL = {BIRD, AB, FLIES, KIWI}
OBJ = {harry, sam}

Schemata: $[BIRD(x)|AB(x) \rightarrow FLIES(x)]$,
$[KIWI(x)| \rightarrow AB(x)]$

Premises: KIWI(harry), BIRD(harry), BIRD(sam)

The two rule schemata describe an NMFS that contains, among others, the following rules:

$<BIRD(harry)|AB(harry) \rightarrow FLIES(harry)>$
$<KIWI(harry)| \rightarrow AB(harry)>$

It induces exactly one admissible extension of the given premise set, namely {BIRD(harry), KIWI(harry), AB(harry), BIRD(sam), FLIES(sam)}.

3 AN NMFS-BASED ARCHITECTURE

We are now ready to sketch a simple architecture for a rule-based inference system that uses an RMS to maintain its states of belief. The PS itself uses schemata to represent its domain specific knowledge. Let R be the NMFS defined this way, and P a finite set of ground instantiated premises.

Furthermore, let R' be the set of rules that have been considered so far by the PS. The RMS then always maintains an admissible extension of the (current) premise set P, w.r.t. to R'. In order to do so, it is clearly sufficient to explicitly consider only the following set N of formulae:

$$N = P \cup \{q \mid \exists <A|B \rightarrow c> \in R' \text{ s.t. } q \in A \cup B \cup \{c\}\}$$

No $q \in L \backslash N$ can belong to an admissible extension of P w.r.t. R'. We call such a triple (N,P,R') a **dependency network**, and representations of such networks are the principal datastructures an RMS works upon. The probably most prominent RMS of today are Jon Doyle's TMS [8] and its descendants such as [4], and Johan de Kleer's ATMS [9]. The usual approach to represent belief states in dependency networks is to attach specific labels to the formulae in N.

TMS represents one admissible extension E with IN/OUT labelings:

$$\alpha: N \rightarrow \{IN, OUT\} \quad \alpha(p) = IN \text{ iff } p \in E$$

The ATMS is restricted to monotonic rules only, but offers the additional possibility of considering multiple premise sets at a time. Actually, ATMS stands for assumption-based TMS, but we consider this a matter of perspective: the ATMS accepts the PS assumptions as premises without further justification, and finds out what can be concluded from them.

Recall that for a monotonic NMFS R', every premise set P has exactly one admissible extension Ext(P). Given a network (N,P,R'), the ATMS simultaneously represents the admissible extensions of every subset $Q \subseteq P$. This is done by means of ATMS-labelings β :

$$\beta: N \rightarrow 2^{2^P}$$

$$X \in \beta(p) \text{ iff } (1)\ p \in Ext(X)$$
$$(2)\ \neg \exists Y: Y \subsetneq X \text{ and } p \in Ext(Y)$$

That is ATMS-labelings exploit the fact that admissible extension membership is monotonic w.r.t. the premises for a monotonic NMFS, and only include minimal premise sets.

We now can see that the various labeling algorithms developed for reason maintenance essentially perform inference in a finite propositional NMFS, and play the role of an efficient core inference engine for the overall system. By constructing valid NM-proofs using the available premises and justifications, it determines what can and should be believed *NOW*. In the case of the TMS, it is non-monotonic inference in the proper sense of the word, in the case of the ATMS. it is simultaneous monotonic inference from multiple premise sets.

The PS itself then mainly has to identify which rules apply in a given situation, select one or several of them, and finally fire. Rule firing corresponds to adding the rule to the NMFS R' maintained by the RMS and letting the RMS incrementally recompute an admissible extension of the so-modified system.

In a companion paper [10], we describe an efficient inference procedure for a restricted class of schemata that have the property that every variable that occurs in the conclusion or in a non-monotonic antecedent of a schema also occurs in a monotonic antecedent of the same schema. This restriction is mostly for pragmatic reasons: the interpreter first tries to match the monotonic antecedents of a schema against formulae in the network and relies on the fact that afterwards, the schema is ground instantiated.

The system uses as a simple condition to decide whether to apply a ground instance $<A|B \rightarrow c>$ of a schema to a current belief state S that $A \subseteq S$ and $B \cap S = \{\}$ and $c \notin S$ holds. This way, rule schemata get some sort of operational semantics on how to - tentatively - construct valid NM-proofs.

ATMS-based multiple context reasoners of course would have to rely on a somewhat different condition for when to apply a rule. A reasonable

method is to check whether all of the antecedents of a rule belong to Ext(Q) for at least one $Q \subseteq P$.

4 A NOTE ON CONTRADICTIONS

One important feature of many RMS is their capability to handle some types of inconsistencies. The usual approach is to have special contradiction formulae and to try to avoid extensions that contain such formulae. Conceptually, contradictions are best thought of as complementary to premises in that they must not belong to any consistent extension. Formally speaking, we distinguish another finite set $C \subseteq L$ of contradictions, in addition to a premise set P, and define the consistent admissible extensions of P relative to C (given an NMFS R) as those admissible extensions E of P that are disjoint from C. The way contradictions are handled in the different RMS heavily depends on how PS-assumptions are represented.

As we have argued e.g. in [11], contradiction handling routines - both in the TMS and in the ATMS - perform essentially meta level reasoning relative to the basic NMFS, in that they inspect NM-proofs to identify the source of a contradiction, and they usually also modify the NMFS maintained by an RMS, in order to invalidate such proofs. We think that meta schemata are a promising yet still unexplored candidate for providing clear descriptions for consistency maintenance routines.

5 DISCUSSION AND COMPARISON TO OTHER WORK

The standard architecture for an RMS-based PS is a collection of some sort of pattern-directed procedures, such as e.g. ATMS-Consumers [12], or RUP-Noticers [13]. Such a procedural approach gives the user a considerable power and flexibility in realizing applications, but it does not support any kind of theoretical analysis. On the other hand, one particular advantage of using NMFS and schemata - besides its uniformity with the basic RMS - is the fact that it opens the way to investigating formal properties of the system as a whole. In this chapter, we briefly sketch two lines of such an investigation, and for lack of space, we defer a detailed discussion to a forthcoming long version of the paper.

5.1 Schema-Level Stratification

It is well-known (for a formal proof see e.g. [14]) that the absence of so-called non-monotonic loops is a sufficient condition for the existence of a unique labeling in dependency networks. In terms of an NMFS R over L, a non-monotonic loop is a sequence of formulae $(p_0,p_1,...,p_n)$ s.t.

- $p_0 = p_n$
- for all $i = 1,2...,n$: $\exists <A_i|B_i \rightarrow p_i> \in R$ s.t. $p_{i-1} \in A_i \cup B_i$
- for at least one i: $p_{i-1} \in B_i$

Given a vocabulary with relation symbols $REL = \{P_1,..., P_k\}$, and a set of rule schemata, we say the set is stratifiable if there is a partial ordering $<$ on REL such that the following holds (P-atom here means an atomic formula in Lx with relation symbol P):

- if $\exists[A|B \rightarrow c]$ s.t. c is a Q-atom and there is a P-atom in A then $P \leqq Q$
- if $\exists[A|B \rightarrow c]$ s.t. c is a Q-atom and there is a P-atom in B then $P < Q$

This definition is chosen in accordance to related work on stratified logic programs with negation, as described e.g. in [15]. Actually, stratifiability analysis for logic programs is often carried out on so-called dependency-graphs, which correspond to RMS dependency networks on the schema level.

It is now easy to see that stratifiability of a set of rule schemata guarantees that there are no non-monotonic loops in the NMFS defined by the

schemata, and hence the NMFS induces unique extensions for every premise set.

Schema level stratifiability is a rather strong condition, and excludes also some NMFS where no non-monotonic loops occur at the instantiated level. Furthermore, there are certain kinds of non-monotonic loops that do not really hurt in that they "only" lead to multiple admissible extensions, though many researchers from the logic programming area might feel uneasy about the resulting sort of three-valued theoremhood. Nevertheless, knowing whether a set of schemata will always lead to a unique and well-defined solution is often an important result.

Also, stratifiability is of considerable practical relevance concerning the algorithmic complexity of admissible extension construction. While there are rather strong polynomial bounds known for loop-free NMFS [4], the unrestricted problem is NP-complete [14].

5.2 Semantical Considerations

Our treatment of rule schemata is similar to Ray Reiter's open defaults [16], which are also considered as a definition of a corresponding set of ground instantiated defaults. A default has the form $(p:Mq_1,...,Mq_n/r)$, where p, the q's, and r are formulae of a full-fledged logic language.

As has been observed independently also in [17], the obvious way to relate an NMFS to default logic is to represent a rule $<p_1,..., p_m|q_1,...,q_n \rightarrow r>$ by the default $(p_1 \wedge ... \wedge p_m:M\neg q_1,...,M\neg q_n/r)$. A default extension of a premise set P is a fixed point of the following operator δ:

$\delta: 2^L \rightarrow 2^L$

$S \rightarrow E_1 \cup E_2 \cup ... \cup E_\infty$

where

$E_1 = P$

$E_{i+1} =$ $Th(E_i) \cup$
$\{r|$ there is a default $(p:Mq_1,...,Mq_n/r)$,
s.t. $p \in E_i$, $\neg q_1 \notin S,..., \neg q_n \notin S\}$

Th(Ei) here stands for the deductive closure of E_i, w.r.t. to some underlying complete inference system for standard logic. This fixed point characterization corresponds closely to our reconstruction sequence for admissible extensions, and Brewka [17] has shown that for every admissible extension of an NMFS, there is a default extension of the corresponding default theory such that the set of ground atoms therein coincides with the admissible extension.

This result, of course, does no longer hold if we allow for compound formulae also in NMFS-rules, since NMFS do not assume an underlying complete set of logical inference rules as default logic does[1]. E.g., the NMFS $\{<p|p \vee q \rightarrow q>\}$ will induce the admissible extension {p,q} of {p}. The point in case is not really that $p \vee q$ is not concluded, but that q is consequently, and there is no default extension which would contain q.

Originally, default logic only had a fixed point semantics, based on the δ-operator above, which is readily available for NMFS without recurring to any logic, by view of the reconstruction theorem. However, default logic has recently been provided with a model theoretic semantics in terms of preferences among sets of models [18].

The main obstacle to inheriting this semantics to NMFS is their inherently finitary and hence logically incomplete character. In a separate paper [19], we have rerepresented Etherington's results using preferences among partial non-truth-functional valuations [20]. Such partial valuations provide a model-theoretical tool to semantically

---

[1]David Etherington has pointed out to me that - in some unpublished work - Prof. Reiter has studied also other versions of default logic, built on arbitrary basic logics.

deal with incomplete belief states, and we are currently investigating ways on how to define an intuitively appealing semantics for NMFS along these lines.

6 CONCLUSIONS

We presented a simple formal framework for RMS that takes into account their inherently finitary and propositional character, and that gives the core concept of valid proof relative to a coherent belief state the predominant role that it deserves. Built on this basic theory, we introduced an approach to truth maintenance with variables, thus providing a uniform framework both for RMS and RMS-based inference systems. One major advantage of our theory is that it offers a sound basis for a systematic analysis of such systems, and in particular supports formal semantical considerations.

ACKNOWLEDGEMENTS

We are indebted to Allen Brown for extensively commenting on an earlier draft of this paper. David Etherington contributed some helpful insights to the relation between NMFS and default logic. No need to say that any remaining faults are exclusively ours. The first author would also like to thank Erik Sandewall and his lab for providing an excellent research environment.

REFERENCES

[1] Drew McDermott and Jon Doyle: Non-Monotonic Logic 1, ARTIFICIAL INTELLIGENCE 13(1,2), pp. 41-72, 1980

[2] Allen Brown: Logics of Justified Belief. Proc. 8th ECAI, pp. 507-512, 1988

[3] David McAllester: An Outlook on Truth Maintenance. Memo 551, MIT AI-Lab,1980

[4] James Goodwin: A Theory and System for Non-Monotonic Reasoning. PhD-Thesis, University of Linkoeping, 1987.

[5] Jon Doyle: Some Theories of Reasoned Assumptions. Research report CMU-CS-83-125, Carnegie-Mellon University, 1983

[6] Drew McDermott: Contexts and Data Dependencies, A Synthesis. IEEE Trans. on PAMI 5(3), pp. 237-246, 1983.

[7] Allen Brown, Dale Gaucas, and Dan Benanav: An Algebraic Foundation for Truth Maintenance. Proc. 10th IJCAI, pp. 973-980, 1987.

[8] Jon Doyle: A Truth Maintenance System. ARTIFICIAL INTELLIGENCE 12, pp. 231-272, 1979.

[9] Johan de Kleer: An Assumption-Based TMS. ARTIFICIAL INTELLIGENCE 28, pp. 127-162, 1986

[10] Hartmut Freitag, Michael Reinfrank: A Non-Monotonic Deduction System Based on (A)TMS. Proc. 8th ECAI, pp. 601-606, 1988

[11] Michael Reinfrank: Lecture Notes on Reason Maintenance Systems, Part I: Fundamentals. RKLLAB-Memo 87-04, University of Linkoeping, 1987.

[12] Johan de Kleer: Problem Solving with the ATMS. ARTIFICIAL INTELLIGENCE 28, pp. 197-224, 1986.

[13] David McAllester: Reasoning Utility Package: User's Manual. Memo 667, MIT AI-Lab, 1982

[14] Dav Benanav, Allen Brown, and Dale Gaucas: Reason Maintenance in a Lattice-Theoretic Framework. Unpublished manuscript, 1987

[15] Teoder Przymusinski: On the Declarative Semantics of Stratified Deductive Databases and Logic Programs. In: Jack Minker (edt.), Foundations of Deductive Databases and Logic Programming, pp. 193-216, Morgan Kaufmann, 1988

[16] Raymond Reiter: A Logic for Default Reasoning. ARTIFICIAL INTELLIGENCE 13(1,2), pp. 81-132, 1980.

[17] Gerd Brewka: Nonmonotonic Logics, Nonmonotonic Inference Systems, and Truth Maintenance - Some Clarifying Remarks. GMD-Research Report, Bonn, 1988.

[18] David Etherington: A Semantics for Default Logic. Proc. IJCAI-87, pp 495-498

[19] Michael Reinfrank: Defaults as Preferences among Partial Worlds - Preliminary Report. Paper presented at the European Workshop on Logical Methods in Artificial Intelligence, Roscoff, 1988.

[20] Erik Sandewall: Semantic States and Non-Truth-Functional Logic. Invited paper at the Third International Symposium on Methodologies for Intelligent Systems (ISMIS 88), Turin, 1988.