Representing Meta-Knowledge in Poole-Systems

Gerhard Brewka
Institut für Informatik, Universität Leipzig
Augustusplatz 10-11, 04109 Leipzig, Germany
brewka@informatik.uni-leipzig.de

June 28, 2000

0.1 Introduction

Commonsense reasoning is nonmonotonic, that is, additional information may invalidate former conclusions. Numerous logics have been proposed which model such forms of reasoning (see [11, 6, 1, 17] for overviews), these logics have been applied to various application problems like reasoning about action, diagnosis, legal reasoning and the like, and in the meantime serious systems implementing subsets of the logics are around, e.g. **XSB** [16], **smodels** [12] or **dlv** [10].

In the formalisms developed so far defeasible conclusions are defined on the basis of a distinction between what is certainly true and what is true by default. Some systems use more fine grained distinctions, based on rankings or arbitrary priorities among the default information.

In this paper we want to study ways of making these distinctions more flexible. In particular, preferences among different pieces of information are not always fixed in advance. To the contrary, such preferences often depend on the current context and establishing them is part of the reasoning problem intelligent agents have to face. In many situations we solve conflicts among different pieces of information, say I_1 and I_2, using meta-information. If we know, for instance, that I_1 stems from a more reliable source than I_2 we tend to prefer I_1.

Here is an example that illustrates what we have in mind. Your wife loves Puccini, so (Rule 1) if they play a Puccini opera in your local opera house you should go with her. Your mother invites you to dinner on sundays and, in fact, (Rule 2) she really expects you to come. Now assume it is sunday and Puccini is being played. You are in trouble since you cannot satisfy what is expected from you, that is Rules 1 and 2 are in conflict. You start to use your meta-information about the rules. You know Rule 1 is your wife's rule, Rule 2 your mother's. You also know that your wife is more flexible than your mother, and (Rule 3) normally it is preferable not to violate expectations of an inflexible person. From this you conclude that you should visit your mother. Now it comes to your mind that this sunday is your wife's birthday. Of course, (Rule 4) you do not want to disappoint her on her birthday, that is, there is now a conflict between Rules 3 and 4. You decide that 4 is to be preferred. This makes you change your former conclusion and you decide to go to the opera.

In this paper we will show how nonmonotonic formalisms can be extended to make reasoning of the kind described in the example possible. We need to be able to represent not only the object level information, but also meta-information. It therefore must be possible to speak about pieces of information, that is, about the formulas we use to represent information. We will use names of formulas to refer to them. It is also important to have the possibility to represent priorities among pieces of information in the logical language. We will do this by using a two-place relation symbol $<$ with standard meaning in the language. Finally, we need a method for defining the nonmonotonic conclusions of a given set of premises which takes the preference information into account adequately.

The formalism we use in this paper are Poole-systems [13]. The reason is that these systems are very simple, yet powerful, especially when they are equipped with our techniques.

The outline of the paper is as follows. In Sect. 0.2 we introduce our proposed generalization of Poole systems. It turns out that with our meta-reasoning techniques some default theories do not possess consistent conclusions. In Sect. 0.3 we introduce, therefore, a new definition of consequence based on the least fixed point of a monotone operator. Sect. 0.4 applies our formalism to several problems in commonsense reasoning. Sect. 0.5 discusses related work and concludes.

0.2 Extending Poole systems

Our formalism extends the well-known Poole systems [13]. A Poole system (F, D) is a pair consisting of

1. a consistent set of (first order) formulas F, the facts, and
2. a possibly inconsistent set of formulas D, the defaults.

A set of formulas E is an extension of a Poole-system (F, D) iff $E = Th(F \cup D')$ where D' is a maximal F-consistent subset of D. The (skeptical) consequences of a Poole system are defined as the intersection of all extensions.

As usual we will often use defaults with open variables in D. Such defaults are representations of all ground instances of the formula. Here is a simple example:

$F:$
$bird(tweety) \wedge penguin(tweety)$
$bird(hansi)$
$\forall x.\, penguin(x) \rightarrow \neg flies(x)$

D :
$bird(x) \rightarrow flies(x)$

Since the instance of the default with $x = hansi$ is consistent with F whereas the instance with $x = tweety$ is not, the single extension contains $flies(hansi)$ but not $flies(tweety)$.

Note that due to the consistency requirement for F extensions always exist, are consistent, and for this reason also the consequences of a Poole system are consistent.

Since we want to represent preference and other meta-information we extend this approach in the following respects:

1. To be able to refer to formulas we use named defaults, that is, pairs consisting of a formula and a name for the formula. Technically, names are just ground terms that can be used everywhere in the language.

2. We introduce a special symbol $<$ for representing preferences. $d < d'$ intuitively says that in case of a conflict d' should be given up rather than d since the latter is more preferred. We require that $<$ represents a strict partial order.[1]

3. We introduce a new notion of extension which takes the preference information into account adequately.

Note that names for facts will not be needed. Meta-information about formulas is used to handle conflicting defeasible information. Since facts are always accepted there is no need to express preference information about them.

We now present the formal definitions. For simplicity, we only consider finite default theories in this paper. A generalization to the infinite case would have to be based on well-orderings rather than total orders.

Definition 1 *A named formula is a structure of the form $d{:}p$, where p is a first order formula and d a ground term representing the name of the formula.*

We use the functions $name$ and $form$ to extract the name respectively formula of a named formula, that is $name(d{:}p) = d$ and $form(d{:}p) = p$. We will also apply both functions to sets of named formulas with the obvious meaning.

[1] We assume that the properties of $<$, like those of equality, are part of the underlying logic and need not be represented through explicit axioms in our default theories.

Definition 2 *A preference default theory $T = (F, D)$ is a pair consisting of*

1. *a consistent set of formulas F,*

2. *a finite set of named formulas D such that $d_1{:}p \in D$, $d_2{:}q \in D$ and $p \neq q$ implies $d_1 \neq d_2$.*

where the language of F and $form(T)$ contains a reserved symbol $<$ representing a strict partial order.

Item 2 of the definition above guarantees that syntactically different formulas have different names.

The following definitions are the central definitions of this paper. They describe how the preference information in the language is used to control the generation of preferred extensions. We first assume that some external total order, represented through the symbol $\prec$, is given which helps us determine a preferred extension (Def. 3). We then generalize this definition to partial orders (Def. 4). In Def. 5 we finally relate $<$ and $\prec$, that is, the preferences in the language and the external preferences, by imposing a compatibility requirement. Preferred extensions then are those which can be generated by an external ordering which is compatible with the ordering described in the extension.

Definition 3 *Let $T = (F, D)$ be a preference default theory, $\prec$ a total order on D. The extension of T generated by $\prec$, denoted $E_T^{\prec}$, is the set $E_T^{\prec} = Th(\bigcup_{i=0}^{|T|} E_i)$ where*

- *$E_0 = F$, and for $0 < i \leq |D|$*

- *$E_i = E_{i-1} \cup \{form(d_i{:}p)\}$ if this set is consistent, E_{i-1} otherwise. Here $d_i{:}p$ is the i-th element of D according to the total order $\prec$.*

The set $\bigcup_{i=0}^{|T|} E_i$ is called the extension base of $E_T^{\prec}$.

We say E is an extension of T if there is some total order $\prec$ such that $E = E_T^{\prec}$. Obviously, all extensions of the standard Poole system $(F, form(D))$ are extensions of T. We now consider the general case of partial orders.

Definition 4 *Let $T = (F, D)$ be a preference default theory, $\prec$ a strict partial order on D. The set of extensions of T generated by $\prec$ is*

$$Ext_T^{\prec} = \{E_T^{\prec'} \mid \prec' \text{ is a total order extending } \prec\}.$$

We next define two notions of compatibility:

Definition 5 *Let $T = (F, D)$ be a preference default theory, $\prec$ a strict partial ordering of D, S a set of formulas. We say $\prec$ is compatible with S iff*

$$S \cup \{d < d' \mid d{:}p \prec d'{:}q\} \cup \{\neg(d < d') \mid d{:}p \not\prec d'{:}q\}$$

is consistent.
An extension E of T is compatible with S iff there is a strict partial ordering $\prec$ of D compatible with S such that $E \in Ext_T^{\prec}$.

The set of extensions of T compatible with S is denoted Ext_T^S.

Definition 6 *Let T be a preference default theory. A set of formulas E is called a preferred extension of T iff $E \in Ext_T^E$.*

Intuitively, E is a preferred extension if it is the deductive closure of a maximal consistent subset of T which can be generated through a preference ordering compatible with the preference information in E itself.

Here is a simple example illustrating preference default theories:

F :
$bird(tweety) \wedge penguin(tweety)$
$\forall x.\ d_2(x) < d_1(x)$
D :
$d_1(x) : bird(x) \rightarrow flies(x) \mid x$ is a ground object term
$d_2(x) : penguin(x) \rightarrow \neg flies(x) \mid x$ is a ground object term

To make sure that the different ground instances of defaults can be distinguished by name we have to parameterize the names also. We assume that terms used as names can be distinguished from other terms which we call object terms.[2] In our example, $d_1(tweety)$, for instance, is a proper rule name, $d_1(d_1)$ is not. Since we only consider finite theories we must also assume that the set of object terms is finite. For clarity we mention the admitted instances explicitly in this and the following examples.

In our example we obtain 2 extensions E_1 and E_2. In E_1 the default $d_1(tweety)$ is rejected, whereas E_2 rejects $d_2(tweety)$. Both extensions contain $d_2(tweety) < d_1(tweety)$. It is not difficult to see that only E_1 can

[2]A more elaborate formalization would be based on sorted logic with sorts for names and other types of objects from the beginning. We do not pursue this here since we want to keep things as simple as possible.

be constructed using an ordering of D which is compatible with this information. E_1 is thus the single preferred extension of this preference default theory.

Preference default theories under preferred extension semantics are very flexible and expressive. Unfortunately, they are lacking one of the nice properties of Poole systems: existence of extensions is not guaranteed, because they can express unsatisfiable preference information. The simplest example is as follows (we assume F is empty):

$$d_1 : d_2 < d_1$$
$$d_2 : d_1 < d_2$$

Intuitively, both of the conflicting defaults say "accept the other one". Accepting the first of the two contradictory formulas requires to give preference to the second, and vice versa. No preferred extension exists for this theory. In other words, the set of skeptical conclusions is inconsistent.

One might say this is no problem since there is something wrong with the available preferences. On the other hand, one of the purposes of nonmonotonic systems is to handle conflicting information, so why should this not hold for conflicting preference information? We think extensions of Poole systems should not destroy one of the nice properties of such systems. We, therefore, define an alternative notion of consequence in the next section.

0.3 Guaranteeing consistency

We will now introduce another, somewhat less standard notion of nonmonotonic consequence based on the least fixed point of a monotone operator. Such operators were used in logic programming as an alternative way of characterizing well-founded semantics [2]. In [4] we used a variant of Baral and Subrahmanian's operator to handle preferences in logic programs under well-founded semantics. A similar approach was used by Prakken and Sartor [14] to model preferences in an argumentation theoretic reconstruction of logic programs.

The intuition underlying the approach developed in this section is as follows: instead of checking each extension E with respect to the preference information contained in E we take a more skeptical view of preference information: we disregard only those extensions which are incompatible with the preference information *contained in all extensions which are not themselves disregarded.* To obtain the set of acceptable extensions we first compute all extensions taking no preferences into account. We then eliminate extensions not compatible with the intersection of all extensions. Since the reduced set of extensions, in the general case, may have a bigger intersection with additional preference information we have to iterate this elimination process until no further extension can be disregarded, that is, until a fixed point is reached. This amounts to considering an extension as acceptable if it is contained in the biggest set of extensions $\mathcal{E}$ satisfying the condition: $E \in \mathcal{E}$ implies E is compatible with $\bigcap \mathcal{E}$.

To formalize this idea we define an operator whose least fixed point is the intersection of the extensions which are acceptable in the sense just described. As is well-known, the least fixed point of a monotonic operator can be computed by iterating the operator on the empty set. In our case, the argument of the operator in each step corresponds to the preference information that needs to be taken into account, and the result of the operator corresponds to the intersection of those extensions which are still under consideration.

Definition 7 *Let T be a preference default theory, S a set of formulas. We define an operator C_T as follows:*

$$C_T(S) = \bigcap Ext_T^S$$

Proposition 8 *The operator C_T is monotone.*

Proof: $S \subseteq S'$ implies that an ordering $\prec$ is compatible with S whenever it is compatible with S'. We thus have $Ext_T^{S'} \subseteq Ext_T^S$ and therefore $\bigcap Ext_T^S \subseteq \bigcap Ext_T^{S'}$. $\Box$

Monotone operators, according to the well-known Knaster-Tarski theorem [18], possess a least fixed point. We, therefore, can define the accepted conclusions of a preference default theory as follows:

Definition 9 *Let T be a preference default theory. A formula p is an accepted conclusion of T iff $p \in lfp(C_T)$, where $lfp(C_T)$ is the least fixed point of the operator C_T.*

We call extensions which are compatible with $lfp(C_T)$ accepted extensions.

Several illustrative examples will be given in the next section. Here we just show how the theory without preferred extension is handled in this approach. We have $D = \{d_1{:}(d_2 < d_1), d_2{:}(d_1 < d_2)\}$ and $F = \emptyset$. We first compute $C_T(\emptyset)$. Since no preference information is available in the empty set we obtain $Th(\{d_2 < d_1\}) \cap Th(\{d_1 < d_2\})$ which is equivalent to $Th(\{d_2 < d_1 \vee d_1 < d_2\})$. This set is already the least fixed point.

All preferred extensions are accepted, but not vice versa. Consequently, the intersection of accepted extensions, that is the accepted conclusions, are a subset of the preferred conclusions. We have the following proposition:

Proposition 10 *Let T be a preference default theory, p an accepted conclusion of T. Then p is contained in all preferred extensions of T.*

Proof: If T has no preferred extension the proposition is trivially true. So assume T possesses preferred extension(s). A simple induction shows that each preferred extension is among the extensions compatible with the formulas computed in each step of the iteration of C_T. Therefore each preferred extension is also an accepted extension. $\Box$

Proposition 11 *Let T be a preference default theory. The set of accepted conclusions of T is consistent.*

Proof: We can show by induction that the set of formulas obtained after an arbitrary number of applications of C_T is consistent. If S is consistent, then Ext_T^S is nonempty since an S-compatible partial ordering $\prec$ exists and each partial ordering generates at least one extension. Moreover, since extensions are consistent the intersection of a nonempty set of extensions is also consistent. $\Box$

0.4 Applications

In this section we discuss several examples to illustrate the expressiveness of our approach.

0.4.1 Types of information

In preference default theories we can classify information according to its reliability. In the following Tweety example we use the three categories observation, strict-rule and default. Strict rules are considered more reliable than observations since they represent either terminological information or well-established information which is based on numerous observations. Observations, on the other hand, may be wrong since the observer may be mistaken. In any case, they are considered more reliable than mere default information.

Assume D consists of the following formulas:

$d_1 : penguin(tweety)$
$d_2 : \forall x.\ penguin(x) \rightarrow bird(x)$
$d_3 : \forall x.\ penguin(x) \rightarrow \neg flies(x)$
$d_4(x) : bird(x) \rightarrow flies(x) \mid x$ is a ground object term

In addition we have the following set of facts F:

$observation(d_1)$
$strictrule(d_2)$
$strictrule(d_3)$
$\forall x.\ default(d_4(x))$
$\forall n, n'.\ strictrule(n) \wedge observation(n') \rightarrow n < n'$
$\forall n, n'.\ observation(n) \wedge default(n') \rightarrow n < n'$

This default theory has 4 extensions, which are obtained by disregarding d_1, d_2, d_3 or $d_4(tweety)$. Applying C_T to the empty set yields their intersection. Since the intersection of these extensions contains $d_1 < d_4(tweety), d_2 < d_4(tweety)$ and $d_3 < d_4(tweety)$ the next iteration of C_T leaves us with a single acceptable extension which does not contain $d_4(tweety)$. The set of accepted conclusions thus does not contain $flies(tweety)$.

Of course, we can make the distinction between types of information much more fine-grained. We could also assign numerical degrees of reliability and use them to break conflicts in a similar way.

0.4.2 Dynamic preferences

Here is the opera example from the introduction.

The set of defaults is as follows (x, y can be instantiated by ground object terms, d, d' by d_1 or d_2):

$d_1 : puccini \rightarrow go\text{-}opera$
$d_2 : sunday \rightarrow visit\text{-}mom$
$d_3(d, x, d', y) : source(d, x) \wedge source(d', y) \wedge more\text{-}flexible(y, x) \rightarrow d < d'$
$d_4(d, x, d', y) : source(d, x) \wedge source(d', y) \wedge birthday(x) \wedge \neg birthday(y) \rightarrow d < d'$
$d_5(d, x, d', y) : d_4(d, x, d', y) < d_3(d', y, d, x)$

Here are the facts F:

$sunday$
$puccini$

$go\text{-}opera \rightarrow \neg visit\text{-}mom$
$source(d_1, wife)$
$source(d_2, mom)$
$more\text{-}flexible(wife, mom)$

We iterate the fixed point operator on the empty set. The first step yields 2 extensions, E_1 generated by all defaults except d_1, and E_2, generated by the defaults except d_2. Note that so far there is no conflict between any instances of d_3, d_4 and d_5. Since $d_3(d_2, mom, d_1, wife)$ is in all extensions, as are the preconditions of the implication with this name, the preference information $d_2 < d_1$ is contained in all extensions. This means that in the next iteration only those extensions which prefer d_2 over d_1 survive. In our case only E_1 is compatible. E_1 is the single acceptable extension and we conclude $visit\text{-}mom$.

Now assume additional facts are obtained, namely

$birthday(wife)$
$\neg birthday(mom)$

Now there is a conflict not only between d_1 and d_2, but also between the rules $d_3(d_2, mom, d_1, wife)$ and $d_4(d_1, wife, d_2, mom)$. Starting with the empty set we thus obtain 4 extensions. All of them contain $d_5(d_1, wife, d_2, mom)$. This means that in the next iteration the conflict between $d_3(d_2, mom, d_1, wife)$ and $d_4(d_1, wife, d_2, mom)$ is resolved in favour of d_4: only the two extensions containing $d_4(d_1, wife, d_2, mom)$ are still under consideration. Since both of these extensions contain the conclusion of the implication with this name, namely $d_1 < d_2$, the next iteration leaves us with a single extension which this time contains $go\text{-}opera$.

0.4.3 KB fusion, sources of information

The next example discusses knowledge base fusion [9, 3]. We have two knowledge bases KB_1 and KB_2. We represent the information contained in the knowledge bases as defaults. In addition, we keep track of where the default information came from, that is, we use a predicate $source(d, kb)$ to express that default d comes from knowledge base kb. We are now able to represent declaratively our strategy of fusing the knowledge bases. For instance, we may want to express that KB_1 is more reliable with respect to a certain topic whereas KB_2 has reliable information about another topic. We can do this by using a predicate $about(d, t)$ which expresses that d is about topic t. Here is a simple example that illustrates what we have in mind (we already add names to the formulas for later use):

KB_1 :
d_1 : $soccer\text{-}champion(1990, Germany)$
d_2 : $ice\text{-}hockey\text{-}champion(2000, Canada)$
d_3 : $tennis\text{-}champion(1999, Becker)$

KB_2 :
d_4 : $soccer\text{-}champion(1990, England)$
d_5 : $ice\text{-}hockey\text{-}champion(2000, Russia)$
d_6 : $tennis\text{-}champion(1999, Agassi)$

For the fusion of the knowledge bases we represent these formulas as named defaults and add formulas describing their sources, their topics and our fusion strategy as facts:

$\forall d.\ source(d, KB_1) \leftrightarrow (d = d_1 \vee d = d_2 \vee d = d_3)$
$\forall d.\ source(d, KB_2) \leftrightarrow (d = d_4 \vee d = d_5 \vee d = d_6)$
$about(d_1, soccer)$
$about(d_2, ice\text{-}hockey)$
$about(d_3, tennis)$
$about(d_4, soccer)$
$about(d_5, ice\text{-}hockey)$
$about(d_6, tennis)$
$\forall d, d'.\ source(d, KB_1) \wedge source(d', KB_2) \wedge about(d, soccer) \wedge about(d', soccer) \rightarrow d < d'$
$\forall d, d'.\ source(d, KB_2) \wedge source(d', KB_1) \wedge about(d, tennis) \wedge about(d', tennis) \rightarrow d < d'$

Given this preference default theory we (correctly) conclude that Germany was soccer champion in 1990 and Agassi was tennis champion in 1999. We refrain from making conclusions about the 2000 ice hockey champion since we have conflicting information and no preference information is available to break the tie (in fact, both KBs are wrong since the Czechs won the ice hockey world championship 2000).

0.5 Related work and discussion

We proposed an extension of Poole's framework for default reasoning where meta-information can be expressed and used to break ties between conflicting pieces of information. Since preference information is expressed in the logical language we can describe conflict resolution strategies declaratively. It turned out that the inclusion of preference information destroys a nice property of Poole systems, namely existence of extensions and thus consistency of the skeptical conclusions. We therefore introduced a new consequence notion based on the least fixed point of a montonic operator. Intuitively, rather than checking for each extension whether the preference information contained in it was respected during the extension's construction, we use the operator to eliminate stepwise those extensions which are not compatible with the preference information contained in all extensions that are still under consideration.

Numerous papers have proposed ways of adding preferences to different nonmonotonic formalisms, for a discussion of a variety of such proposals see [7]. Almost all of them do not represent meta-information in the logical language and thus lack the flexibility of our approach. The only exception - apart from earlier work of this author - is [14]. Least fixed points have been used to define consequences of prioritized logic programs in [4] and [14]. The latter authors acknowledge that their approach is based on a suggestion by this author.

In a companion paper [5] we apply a variant of the framework of this paper to belief revision [8]. We represent epistemic states as preference default theories under the least fixed point semantics. New information is simply added to the default theory. The nonmonotonic inference relation takes care of everything. Strategies of how to revise beliefs can be represented declaratively.

Acknowledgements

The presented work was funded by DFG, Forschergruppe Kommunikatives Verstehen. I also thank R. Booth, S. Lange, H. Sturm and F. Wolter for helpful comments.

Bibliography

[1] Antoniou, G., *Nonmonotonic Reasoning*, MIT Press, 1997

[2] Baral, C., Subrahmanian, V.S., Duality between Alternative Semantics of Logic Programs and Nonmonotonic Formalisms, Intl. Workshop on Logic Programming and Nonmonotonic Reasoning, 1991

[3] Benferhat, S., Dubois, D., Kaci, S., Prade, H., Encoding Classical Fusion in Ordered Knowledge Bases Framework, Proc. 8th Intl. Workshop on Nonmonotonic Reasoning, Uncertainty Frameworks track, Breckenridge, 2000 Framework for Default Reasoning, Proc. IJCAI-89, 1989

[4] Brewka, G., Well-Founded Semantics for Extended Logic Programs with Dynamic Preferences. *Journal of Artificial Intelligence Research*, 4:19–36, 1996.

[5] Brewka, G., Declarative Representation of Revision Strategies, Proc. European Conference on AI, ECAI, Berlin, 2000

[6] Brewka, G., Dix, J., Konolige, K., *Nonmonotonic Reasoning - An Overview*, CSLI Publications, Stanford, 1997

[7] Brewka, G., Eiter, T., Prioritizing Default Logic, *Intellectics and Computational Logic: Festschrift 60th Anniversary of W. Bibel*, Kluwer Academic Publishers, Applied Logic Series 19, 2000

[8] Gärdenfors, P., *Knowledge in Flux*, MIT Press, Cambridge, MA, 1988

[9] Cholvy, L., Reasoning About Merging Information, in *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, Vol. 3, 233–263, 1998

[10] Eiter, T., Leone, N., Mateis, C., Pfeifer, G., Scarcello, F., The KR System dlv: Progress Report, Comparisons and Benchmarks, Proc. 6th Intl. Conference on Principles of Knowledge Representation and Reasoning, Morgan Kaufmann, 1998

[11] Marek, W., Truszczyński, M., *Nonmonotonic Logics – Context-Dependent Reasoning.* Springer, 1993.

[12] Niemelä, I., Simons, P., Efficient Implementation of the Stable Model and Well-Founded Semantics for Normal Logic Programs, Proc. 4th Intl. Conference on Logic Programming and Nonmonotonic Reasoning, Dagstuhl, Germany, Springer Verlag, 1997

[13] Poole, D., A Logical Framework for Default Reasoning, *Artificial Intelligence* 36, 1988

[14] Prakken, H., Sartor, G., Argument-Based extended Logic Programming with Defeasible Priorities, *Journal of Applied Nonclassical Logics* 7:25–75, 1997

[15] Reiter, R., A Logic for Default Reasoning, *Artificial Intelligence* 13:81–132, 1980

[16] Sagonas, K., Swift, T. and David S. Warren. An Abstract Machine for Computing the Well-Founded Semantics. In Proceedings of the Joint International Conference and Symposium on Logic Programming Bonn, Germany, pages 274–288,MIT Press, 1996

[17] Schaub, T., *The Automation of Reasoning with Incomplete Information: From Semantic Foundations to Efficient Computation*, Springer Verlag, 1998.

[18] Tarski, A., A lattice-theoretical fixpoint theorem and its applications, *Pacific Journal of Mathematics* 5:285-309, 1955