Defeasible inheritance with doubt index and its axiomatic characterization

Erik Sandewall [a,b,∗]

[a] *Linköping University, Linköping, Sweden*
[b] *Royal Institute of Technology, Stockholm, Sweden*

ARTICLE INFO

ABSTRACT

This article introduces and uses a representation of defeasible inheritance networks where links in the network are viewed as propositions, and where defeasible links are tagged with a quantitative indication of the proportion of exceptions, called the *doubt index*. This doubt index is used for restricting the length of the chains of inference.

The representation also introduces the use of *defeater literals* that disable the chaining of subsumption links. The use of defeater literals replaces the use of negative defeasible inheritance links, expressing "most A are not B". The new representation improves the expressivity significantly.

Inference in inheritance networks is defined by a combination of axioms that constrain the contents of network extensions, a heuristic restriction that also has that effect, and a nonmonotonic operation of minimizing the set of defeater literals while retaining consistency.

We introduce an underlying semantics that defines the meaning of literals in a network, and prove that the axioms are sound with respect to this semantics. We also discuss the conditions for obtaining completeness.

Traditional concepts, assumptions and issues in research on nonmonotonic or defeasible inheritance are reviewed in the perspective of this approach.

© 2010 Elsevier B.V. All rights reserved.

1. Background and overview

Nonmonotonic or defeasible inheritance is one of the classical topics in Knowledge Representation. It concerns structures where there is a number of classes, a subsumption predicate whereby one class can be subsumed by several superior classes, and a defeasible variant of the subsumption predicate where the propositions "A is defeasibly subsumed by B" and "B is defeasibly subsumed by C" allow one to infer "A is defeasibly subsumed by C *unless* there is information to the contrary. Propositions of this kind are commonly called *links*; the classes are referred to as *nodes* in *inheritance networks*.

1.1. Earlier work on nonmonotonic inheritance

The *path-based* approach to this topic defines methods for identifying *paths*, i.e. sequences of subsumption relationships or subsumption-related relationships in the given inheritance network describing the application at hand [56]. Paths may be related as e.g. situators, preemptors, conflictors or defeaters, according to their structure, and this is used to define what paths are *permitted* by a given inheritance network [60]. *Skeptical* approaches define one single extension consisting

---

∗ Address for correspondence: Linköping University, Linköping, Sweden.
  E-mail address: erisa@ida.liu.se.

of permitted paths; *credulous* approaches define a set of *permitted extensions* allowing for different possibilities. A credulous system can either select one of these extensions on extralogical grounds, or use the intersection of the permitted extensions.

Extensions are usually characterized using rules that define or at least constrain the status of the various paths in a given inheritance network. However, it would be useful to have an *axiomatic representation* of these constraints in the sense of a set of logic formulas that are satisfied in (permitted) extensions. This may facilitate the formal analysis of those constraints, and it should also be a first step towards integrating defeasible-inheritance information with other information about the domain at hand.

Given that defeasible inheritance is generally recognized as an example of nonmonotonic logic, it will not be sufficient to merely have a set of axioms; one also needs an appropriate nonmonotonic reasoning policy in the form of, for example, a circumscription policy or a preference relation on extensions. We shall use the term "axiomatic representation" for the combination of a set of axioms and a nonmonotonic reasoning policy.

Sandewall [47] proposed one such axiomatic representation and validated it by applying it to a set of test cases that were widely used in the literature at the time. Simonet [51] observed that this representation obtained unintended results for some additional test cases that had emerged later, and proposed a modification of Sandewall's representation. However, she also observed some other cases that even the modification did not handle as intended.

Schlechta [49] has shown that no path-based approach to skeptical reasoning (in a reasonable sense of that term) can produce the intersection of credulous extensions. Touretzky et al. [60] therefore observed that "we cannot even axiomatize ideal skepticism in our purely path-based formalism".

Unfortunately, the relation between the scenarios to be represented and the proposed representation was never obtained in a systematic way in these works; it was always only shown by way of examples. Several authors, including [47] and [51], have remarked that it would be desirable to replace the use of examples by an underlying semantics for defeasible networks, but the large number of semantics for defeasible inheritance that have been proposed over the years (see Section 10.3) have served other purposes than clarifying the relation between the application at hand and its formal representation.

Recent work on nonmonotonic inheritance generally pay less attention to the traditional representational questions and focus instead on technical aspects of the nonmonotonic logic being used, such as its complexity properties, or the introduction of a priority ordering on the links in the network, which are sometimes represented as default rules [4,36,28]. We propose that it is important to check new theories of defeasible inheritance against a sufficient number of known, difficult cases of node configurations, and this aspect is emphasized in the present article.

1.2. Overview of the article

This article consists of the following sections:

1. The present section;
2. The investigated approach: the characteristic features of how we represent nonmonotonic inheritance, but without going into technical details;
3. Representation of inheritance networks: defines the formal representation of a network as a set of propositions, and defines the concept of extension of an inheritance network;
4. Axioms and other restrictions: specifies the restrictions that we propose to apply to network extensions, i.e., the axiom part of the axiomatic representation;
5. The Proportion Semantics: the underlying semantics which is used for verifying the soundness of the axioms in the axiomatic representation, and for analyzing possible nonmonotonic policies in it;
6. Explanation of the axioms;
7. Inference operation: definition and motivation for the nonmonotonic policy component of the approach;
8. Issues in commonsense inheritance: relates the approach and the results of the present article to some standard examples and issues in the area;
9. Object-level predicates and description logics: discusses an extension of the expressivity of the approach that is used here;
10. Alternative approaches to defeasible inheritance;
11. Conclusion.

Appendix A contains a discussion of the possibility of also proving completeness for the set of axioms and suggests an approach to doing this. Appendix B contains a number of additional examples besides those that are found in the text.

2. The investigated approach

2.1. Characteristic features of the representation

In this article we investigate an approach to the representation of defeasible inheritance that has two characteristic features. First, defeasible subsumption links are *annotated with a doubt index*, i.e., a number indicating the extent to which exceptions must be expected. A *doubt annotated subsumption link* between two classes c and d may be written c sub^m d,

where m is usually a small integer when given by the sources. These doubt indices are used to control the transitivity of defeasible subsumption, so that from c sub^m d and d sub^n e one will defaultwise conclude c sub^{m+n} e. However, the representation requires the use of a threshold K that sets a limit to the chaining, so that the conclusion in the example just shown is only accepted if $m + n \leqslant K$. This provides a cut-off point beyond which further chaining is not admitted.

The use of doubt annotated links distinguishes our approach from earlier approaches to defeasible inheritance, and indeed from earlier approaches to nonmonotonic reasoning in general.

Secondly, the present approach corrects a deficiency in many earlier approaches to inheritance networks which can be illustrated by the following example. Consider a distinction between "white birds" and "grey birds". Most doves are grey. In my park there are a lot of birds; most of them are doves. However, about half of them are white and the other half are grey. We therefore have the sub predicate from birds-in-my-park to doves, and from doves to grey birds, and we wish to override the transitivity that is otherwise obtained by default. This is done using an additional 'defeater' predicate nsub in our approach. Most traditional approaches have only one way of suppressing an inferred subsumption, namely by asserting or inferring another subsumption that is incompatible with the one in question, and such approaches cannot express scenarios such as this one.

The defeater predicate is analogous to *exception links* that were used in some of the earliest work on nonmonotonic inheritance, in particular by Fahlman [16] and the early work of Touretzky [59] (and Boutilier [6] uses a related notation), but this representational feature has largely fallen into oblivion.

Concurrently with the present work, Gabbay and Schlechta have proposed an approach to defeasible inheritance using *reactive diagrams* [24] which also uses exception links. However, the use of a doubt index does not have any counterpart in their approach.

In fact, the defeater predicate serves two purposes in the approach that is studied here. Besides its use for characterizing an application, like in the example just given, it is also used as the predicate that is to be minimized for the purpose of obtaining conclusions nonmonotonically, similar to the use of abnormality predicates in circumscription.

2.2. Motivation for the proposed approach

The major reason for the proposed approach is its improved expressivity. The use of an explicit defeater predicate provides expressivity that the traditional approach using positive and negative defeasible links does not have, as illustrated by the grey doves scenario above. Likewise, the use of a doubt index makes it possible to restrict the number of inference steps, which is natural since additional uncertainty is introduced in every such step.

Each link in a network is considered as an elementary proposition, and a network is therefore a set of literals. This makes it possible to specify axioms and other restrictions that define the inference operation in an inheritance network. An additional advantage of the use of doubt indices and defeater literals is that it forms the basis for the definition of a precise underlying semantics for the inheritance networks. This aspect of the approach makes it possible to investigate the formal properties of the inference system, beginning with a verification of the soundness of the set of axioms.

There are also other representational problems in the traditional approach that go away with the proposed new approach. For example, with the traditional approach one obtains circular link structures even in quite simple situations, like disjoint or almost-disjoint classes, or for a case where most A are B but most B are not A, i.e. when A is approximately a small subclass of B. A semantics for such circular structures has been proposed by Wang et al. [61]. Our approach using a defeater predicate does not obtain a circular structure in such cases. An assumption of noncircularity in inheritance networks appears to facilitate proofs and computational procedures, and in our approach it is an insignificant restriction from the application point of view.

Last but not least, the underlying semantics provides a precise basis for identifying the *expected conclusions* in many of the common scenario examples. It is not satisfactory to define expected conclusions in terms of intuition and presumed common sense, and the underlying semantics provides a better basis in this respect.

2.3. User aspects and underlying semantics

The use of a doubt index and a defeater predicate provides additional expressivity for the user of the representation. As always, there is a question of what this requires from the user. On what basis will the user select the doubt indices on the various inheritance links, and will it be practically possible to identify the explicitly stated instances of the defeater predicate?

One possible answer to these questions is to present the nonmonotonic reasoning system with its notation and its inference mechanism as a kind of machine that it is up to the user to use and to master. For example, the user may be advised to start by using the doubt index 1 on each defeasible inheritance link, so that the threshold K simply specifies the maximum permissible number of links in subsumption paths, and then to use other values for the doubt index if there is a particular reason for doing so. One leaves it to the user to select doubt indices and to introduce defeater literals until her representation of her application has been sufficiently debugged in the sense of providing the intended results when tested.

Although this "practical engineering" approach may be the only possible one in practical situations, it is obviously not satisfactory from a scientific point of view, nor in the long run even for practical purposes. For a more systematic approach, the defeasible inheritance network that is provided as the input to the inference process should be viewed as a *domain*

model of the application at hand. This means in particular that inasmuch as default conclusions are those that are drawn in the absence of contrary information, the application model shall be assumed to be *epistemologically complete* with respect to such contrary information, that is, it should be as complete as is required for the purpose at hand. If the application model omits significant information about exceptions, then it is not possible for an inference mechanism to compensate for it. Furthermore, if an epistemological incompleteness is detected, then it should be repaired by adding information that characterizes the exception in question, and not by tinkering with the model in some other way.

This principle requires that the representational conventions that are used for the domain model must be able to express exception information, and that the semantics of such information must be well defined. In the approach that was outlined in Section 2.1, in particular, we need definitions of what are the appropriate values for the doubt indices and what are the appropriate choices of the defeater predicate, based on known properties of the application at hand. We shall use the term *underlying semantics* for such a formal basis.

Notice that an underlying semantics can be useful even if there is no practical way of calculating exact doubt index values in a given situation. It may be that one can define a learning process where these values are adjusted continuously based on experience, and the underlying semantics is relevant for defining the learning process. It may also be that instances of the defeater predicate that have been inferred in some reasoning tasks, can then be accepted to the knowledge base in their own right and for use at later times.

Furthermore, even in those circumstances where the "practical engineering" approach is in fact adopted, one may consider the possibility of using the definitions of appropriate values and choices as heuristic guidelines when the aforementioned nonmonotonic reasoning "engine" is to be equipped with contents.

Finally, as one proceeds to defining rules of inference or other inference machinery for the representation, it is appropriate to have a definition of the meaning of index values and of defeater literals as a basis. One may of course object that this does not matter in the case of the "practical engineering" approach, since in its case the formalism and the machinery are presented to the user and it is up to her to make the best use of it, regardless of whether its design is systematic or ad hoc, but the existence of an underlying semantics ought to be reassuring for designer and user alike.

2.4. Approach for the underlying semantics

Our representation of inheritance networks uniformly uses one type of things, called *classes*. Each class is supposed to have a finite and nonempty *set of members*. These members are *objects*, but objects are only used in the underlying semantics and not in the representation system. In addition the representation system uses the domain of the non-negative real numbers as values of the doubt index. We use the term 'class' alternatingly for the abstract entity and for the symbol representing that entity.

Although the representation system does not have any constructs for objects per se, we shall distinguish between two kinds of classes, namely *singleton classes* that have exactly one member each, and *concept classes* that have an arbitrary number of members, and normally a considerable number of them.

The basic idea in the underlying semantics is to interpret a doubt annotated subsumption link c sub^m d as a statement "the proportion of members of c that are members of d is at least p" where p is a number slightly smaller than 1, or equal to 1, and where there is a one-to-one mapping between the doubt index m and the proportion p. This is the same as saying that the probability that a randomly drawn member of c shall be a member of d, is at least p, using probabilities based on actual frequency.

In common-sense reasoning there may be situations where one states that "members of the class A are typically members of the class B" even though the number of non-B in A is not negligible. Our usage of the sub predicate is not intended to represent all aspects of class subsumption in natural language, however; it is simply defined to have the meaning stated above.

Notice that the interpretation is that the proportion in question is *at least p* and not that it *is equal* to p. This is because there may be more than one chain of links from c to d in the inheritance network at hand which result in different values for p. If the interpretation had been that the proportion in question *is p* then one would obtain an inconsistency in such cases.

If the proportion of c in d is at least p and the proportion of d in e is at least q, then the proportion of c in e is at least $p \times q$ unless the distribution of the c's in d is uneven in the direction of the part of d that is not in e. The index value m in c sub^m d must therefore be interpreted essentially as the logarithm of p, so that the multiplication of proportions is mapped to the addition of their logarithms. More exactly, however, m is interpreted as $-\Phi \times \ln(p)$ where Φ is a constant whose purpose is to obtain values that are often in the range of 1 to 10, or 1 to 100, rather than small fractions of 1. The function ln is the natural logarithm as usual. Notice that if $p = 1 - \varepsilon$ for a small ε then $\ln(p)$ is approximately $-\varepsilon$. The minus sign in the interpretation of m obtains that doubt index values are positive rather than negative. The function φ is defined so that the relation between m and p will be written as $p = \varphi(m)$.

The caveat that the multiplication of proportion values only applies if the distribution of the c's in d permits it is an essential precondition for the formal analysis. It must also be respected when the approach is used for a practical application, to the extent that it is practically possible to respect it. The defeater predicate is used for characterizing situations where the distribution is uneven in such a way that the exception must be stated explicitly.

2.5. Singleton classes

The interpretation of the doubt index described above applies to the case where the nodes in the inheritance network represent sets with an arbitrary and nontrivial number of members (concept classes), and it is not applicable for nodes that represent a single object (singleton classes).

The underlying semantics is defined only for concept classes, but actual uses of inheritance networks must accommodate both singleton and concept classes. In fact, one important type of query for an inheritance network has the form c sub^m d where c is a singleton class and d is a concept class. It is therefore important to provide an interpretation of such queries as well.

We distinguish the following two cases. In a simple case, all links that involve singleton classes *in the given network* are strict links that do not admit any doubt, i.e. links of the form c sub^0 d where c is a singleton class and d is a concept class. The inference process may identify inheritance chains resulting in conclusions of the form c sub^m e where $m > 0$. The interpretation for the user of such a conclusion will then be: "there is some class d in the network that contains the sole member of c such that the probability of a random member of d being a member of e, is at least $\varphi(m)$". From the application point of view this may be abbreviated as "the member of c is a member of e with probability $\varphi(m)$".

In the more general case links of the form c sub^k d with singleton c and a nonzero value of k are admitted in the given inheritance network as well. The underlying semantics does not provide any interpretation for such a link. From an application point of view it is natural to consider it as an expression of a probability, viz., that the probability of c being in d is at least $\varphi(k)$, but the nature of that probability is outside the scope of the underlying semantics. We shall refer to it as an *external probability*. In this case, the interpretation for the user of a conclusion of the form c sub^m e will be: "there is some class d in the network such that the probability is at least $\varphi(k)$ that the sole member of c is a member of d, and the probability of a random member of d being a member of e, is at least q, where $\varphi(k) \times q = \varphi(m)$".

From the application point of view it may well be appropriate to express this more concisely by saying that the probability of c being in e is at least $\varphi(m)$, but from a formal point of view this combines probability information from two different sources with different character.

Links of the form c sub^m d where both c and d are singleton classes must also be interpreted in terms of external probabilities, to the extent that such links are allowed to occur.

Therefore, given sub links with a singleton first argument are not to be used in the defeasible inference process, but shall only be taken as statements of an externally provided probability. Since they have no essential relevance for the topic of the present article, we shall make the simplifying assumption that given inheritance networks only contain strict subsumption links for singleton classes.

3. Representation of inheritance networks

The *representation system* is the entire set of conventions, assumptions and restrictions that are used for expressing domain information and for making inferences from it. The conventions for representing an inheritance network as a set of formulas is the first part of the representation system.

3.1. Predicates and inheritance networks

The representation system is based on named classes each of which has a set of members, which are called objects. Applications may associate other information with classes, besides their sets of members, and it is therefore possible to have two classes that have equal sets of members but which nethertheless are different classes. However, from the formal point of view and in this article, it is only the set of members that is of interest. Therefore, all the predicates on classes are defined in terms of their member sets.

Five predicates on classes are used, namely $\sqsubseteq$, dj, sub, dsub and nsub, with the following intended meanings:

- $a \sqsubseteq b$ means that the set of members of a is a subset of or equal to the set of members of b;
- a dj b means that the classes a and b do not have any common member;
- a sub^m b means that according to given sources, most members of a are members of b, where m is the doubt value, expressed as a real number;
- a $dsub^m$ b means that it has been inferred that most members of a are members of b, with m again being the doubt value;
- nsub(a,b,c) is used to represent a *defeater literal* that suppresses the default of chaining a dsub b and b sub c.

We shall return to the distinction between sub and dsub in Section 4.5.

An expression consisting of one of these predicates with its proper arguments will be called a *literal*. This corresponds to a "link" in many earlier articles. If the classes that occur as arguments in the literal belong to a set *C* of classes, then it is a literal *over C*.

In order to avoid having to define inference steps that are trivial conclusions in set theory for the sets of members for classes, we introduce an assumption to the effect that these trivial conclusions are already present in given inheritance

networks. The following definition is used. Let S be a set of singleton classes and C be a set of concept classes. A *network kernel* over S and C is a set of literals over $S \cup C$ using the $\sqsubseteq$ and dj predicates that is satisfiable and saturated inferentially, so that it contains all possible conclusions with respect to those two predicates. Furthermore the kernel must contain literals of the form c dj d for every pair of different singleton class symbols, which constitutes a unique names assumption.

The satisfiability requirement means that the kernel cannot contain e.g. a literal of the form c dj c. (Recall that each class must have a nonempty set of members.) It also cannot contain a literal of the form c $\sqsubseteq$ d where c and d are different singleton classes.

An *inheritance network* is a fivetuple $\langle S, C, \Gamma, \Delta, \Lambda \rangle$ where S is a set of singleton classes, C is a set of concept classes, Γ is a network kernel over S and C, Δ is a set of literals over C alone using the sub, dsub and nsub predicates, and Λ is a set of literals using the sub, dsub and nsub predicates where the first argument is in S and the other argument(s) are in C.

Notice that only positive literals occur in inheritance networks.

Unlike most earlier approaches, in this system there is no predicate for "members of a are usually not members of b". Such situations have to be expressed in some other way, for example using an additional class c whereby one can write a sub^m c and c dj b.

3.2. Initial inheritance networks and their extensions

The following conventions and definitions will be used in defining how to make inferences from inheritance networks.

The set of those literals in an inheritance network that use a particular predicate R will be called *the contents for R* of the inheritance network.

Consider two inheritance networks $N = \langle S, C, \Gamma, \Delta, \Lambda \rangle$ and $N' = \langle S, C, \Gamma, \Delta', \Lambda' \rangle$ which means that they have equal kernels. Then N' is said to be an *extension of N* iff $\Delta \subseteq \Delta'$ and $\Lambda \subseteq \Lambda'$. It is called a dsub-*extension* iff the two networks differ only in their contents for dsub, and similarly for nsub-*extension*.

As in most other work on inheritance networks, the inference operation is defined as an operation that takes an inheritance network as input, called the *given network*, and that produces an extension of that network, containing conclusions that can reasonably be drawn from the given network. This extension will be called the *derived extension*.

We need to make particular assumptions about the given inheritance network in the inference operation and introduce the following definition. An inheritance network $N = \langle S, C, \Gamma, \Delta, \Lambda \rangle$ is said to be *initial* iff the Δ component uses only the predicates sub and nsub, but not dsub, and the Λ component uses only the nsub predicate.

This definition is made because dsub is to be used for conclusions, and not for initially given links. The restriction for Λ expresses that singleton classes are only allowed to participate in strict links such as c $\sqsubseteq$ d, which are then included in the kernel, and not in defeasible links.

This framework imposes a strong restriction on the expressivity of conclusions, compared to what is the case in logic in general. Single literals can be obtained as conclusions; conjunctions of literals can be obtained since the literals in a network are implicitly conjunct. Negation of a proposed literal a $\sqsubseteq$ c can be obtained indirectly if the extension contains a $\sqsubseteq$ b and b dj c for some b, since each class is assumed to have a nonempty set of members. Negations of sub, dsub and dj can be obtained in similar ways. However, a complement set cannot be represented, nor can a disjunction of literals be obtained as a conclusion according to this view of inference, and the same applies of course for a negation of conjuncts. (A generalization in these respects will be briefly described in Section 9.)

4. Axioms and other restrictions

We proceed now to the operation that obtains the derived extension of a given network.

4.1. Notation for extension constraints

The most natural way of defining the derived extension of a given inheritance network may seem to be to introduce a set of inference rules, and to define the derived extension as the set of all literals that can be obtained from the given network by successive use of these inference rules. However, we choose instead to define the derived extension using *extension constraints*, that is, a set of logic formulas that impose restrictions on the permissible contents and structure of extensions, since this provides a natural way of expressing and analyzing a relatively complex inference mechanism. The axioms of the axiomatic representation are introduced as extension constraints.

The notation for expressing extension constraints is separate from, and richer than, the notation for the inheritance networks themselves, and is defined as follows. We assume a set of *variable symbols* that is disjoint from the set of identifiers for classes, and which is partitioned into a set of *class variables* and a set of *numerical variables*. No type distinction is made between variables for singleton classes and those for concept classes. A *numerical term* is defined recursively as a number, a numerical variable, the symbol K, or a composite expression that is formed from numerical terms using the addition function.

An *atomic proposition* is either an expression $x \leqslant y$ where x and y are numerical terms, or an expression that is similar to a literal but with class variables instead of class identifiers, and with numerical terms instead of numbers. A *proposition*

is formed recursively from atomic propositions using the standard propositional connectives, in particular $\wedge$, $\neg$, $\rightarrow$ and $\leftrightarrow$. Quantifiers are not used.

Extension constraints will be expressed as propositions.

The *evaluation* of a proposition can now be defined. Let the following be given: an inheritance network $\langle S, C, \Gamma, \Delta, \Lambda \rangle$; sets of variables for classes and for numbers; mappings from class variables to members of $S \cup C$ and from numerical variables to non-negative real numbers; and finally a specific positive number called K. The value of a variable is always the class identifier or the number assigned to it by the respective variable-value mappings. The value of K is K, and the value of a number is itself. The value of a numerical term of the form $m + n$ is obtained as the sum of the values of the sub-expressions m and n.

The truth-value of an atomic proposition is obtained as follows. If the proposition has the form $x \leqslant y$ then it is T if the value of x is less than or equal to the value of y, and F otherwise. For all predicates having classes as arguments, substitute the arguments containing class variables with their value according to the mapping for class variables. Also, if the predicate in the proposition is `sub` or `dsub` then substitute similarly the numerical term used for the doubt value. If the resulting literal (variable-free atomic proposition) is a member of $\Gamma \cup \Delta \cup \Lambda$ then the truth-value is T, otherwise F.

The truth-value of a composite proposition is obtained using the standard truth-tables.

Finally, a proposition is *satisfied in* an inheritance network and *for* a particular value of K iff its value is T for all possible choices of the mappings for class variables and numerical variables.

Notice that propositions are *only* intended to be used for expressing axioms and other restrictions that are applied to inheritance networks in order to characterize extensions. They are not intended for making composite logical statements about the application that the inheritance network describes. The evaluation rules and the assignment of truth-values to propositions must be understood accordingly.

4.2. Axioms and additional restriction

The following set of eight axioms will be used for obtaining derived extensions.

```
1.    c ⊑ d ↔ c sub⁰ d
2.    c subᵐ d ∧ d ⊑ e → c subᵐ e
3.    c subᵐ d → c dsubᵐ d
4.    ¬(c dsubᵐ d ∧ c dsubⁿ e ∧ d dj e)
5.    ¬nsub(c,c,d)
6.    c dsubᵐ d → m ≤ K
7:1.  c dsubᵐ e ∧ e subⁿ g ∧ ¬nsub(c,e,g) ∧ m+n ≤ K
         → c dsubᵐ⁺ⁿ g
8.    c dsubᵐ d ∧ m ≤ n ∧ n ≤ K → c dsubⁿ d
```

All of these except Axioms 4 and 5 can be read informally as if–then rules that allow one to infer additional literals from given ones. Under this reading it is also possible to construct quasiformal proofs of derived literals. However, in strict terms we shall only use these formulas as restrictions on sets of literals, i.e., on inheritance networks.

Axiom 7:1 is the first member of a sequence of successively stronger axioms that will be called 7:2, 7:3, etc., where we have, in particular,

```
7:2.  c dsubᵏ d ∧ d subᵐ e ∧ e subⁿ g ∧
         ¬nsub(c,d,g) ∧ ¬nsub(d,e,g) ∧ k+m+n ≤ K
            → c dsubᵏ⁺ᵐ⁺ⁿ g
```

Each axiom in that sequence entails its predecessor. We shall return to the later variants of this axiom when considering the completeness of the set of axioms. Axiom 7:1 is easy to understand and it suffices in many situations.

In addition to these *axioms* we shall also use an additional restriction, called *Restriction 9*, which is

```
9.   c dsubᵐ d ∧ nsub(d,e,g) → nsub(c,e,g)
```

It will be just called a restriction, and not an axiom, since its motivation in terms of the semantics is different from that of the axioms.

The constant K is used since the reliability of the conclusions decreases when several defeasible links are combined by way of transitivity. The condition on $m + n$ in Axiom 7:1 is a way of stopping the conclusions when the doubt value has increased to a certain point. The choice of an appropriate value for K depends on the needs of the application and on the metric that is used for the assignment of doubt values. The permitted range of values for K will be specified and explained in Section 5.1. The definitions that follow below are made under the assumption that a value for K has been fixed.

Notice that Axiom 5 does not force negated literals of the form $\neg$`nsub(c,c,d)` to be included in inheritance networks; it merely prevents literals of the form `nsub(c,c,d)` from being included there. Inheritance networks consist of positive literals only.

Notice also that the combined occurrence of `nsub(c,d,e)` and c `dsub`^{m} e is not necessarily a contradiction and it may be meaningfully used. Consider for example the following literals:

    c dsub¹ d
    d dsub¹ e
    nsub(c,d,e)
    c dsub⁴ e

Here, the occurrence of the `nsub` literal precludes the conclusion c `dsub`^{2} e that would otherwise have been possible using Axiom 7:1. There is still a link from c to e, but with a higher doubt index.

4.3. Correct and valid inheritance networks

Definition. An inheritance network is said to be *correct* iff Axioms 1 to 6 are satisfied in it.

Definition. An inheritance network is said to be *valid* iff Axioms 1 to 8 are satisfied in it.

Proposition 1. *Every correct inheritance network has at least one valid extension.*

Proof. The extension obtained by adding literals `nsub(a,b,c)` for all combinations of concept classes a, b, c that occur in the given network and where a is different from b satisfies Axioms 1 to 8. $\square$

Definition. An inheritance network is *consistent* iff it has a valid `dsub`-extension.

Definitions and results in the sequel will apply to correct inheritance networks, so the role of Axioms 1 to 6 is to enforce well-formedness conditions on the inheritance networks being considered. For example, a correct inheritance network cannot contain inconsistencies such as

    c sub^m d
    c sub^n e
    d dj e

with $m \leqslant K$ and $n \leqslant K$.

The correctness requirement precludes making those statements with m and n greater than K. This condition is imposed since otherwise Axiom 4 could not be maintained.

The effect of Axiom 8 is that in every valid inheritance network, the possible values for m in c `sub`^{m} d are either the empty set, or a closed interval from some k to K. In practice it is the left endpoint k of that interval that one is interested in, and that is also how it should be represented in an implementation, but for the formal analysis it is technically simpler to allow the entire interval.

4.4. Minimal extensions of inheritance networks

Definition. A valid `dsub`-extension of a consistent inheritance network is said to be *minimal* iff there does not exist any other valid `dsub`-extension whose contents for `dsub` are strictly smaller in the sense of being a strict subset.

Proposition 2. *Every consistent inheritance network has a unique minimal `dsub`-extension.*

Proof. The proof is straightforward from the definitions. $\square$

It is easily seen that the minimal `dsub`-extension contains all conclusions that are enforced by the axioms, and no others. There is a separate question of `nsub`-minimality where one is interested in extensions whose `nsub` components are minimal for obtaining consistency, but this is a topic for a later section.

We must also be able to relate extensions with different contents for `nsub`. Consider two valid extensions N_1 and N_2 of a given, correct inheritance network. The *meet* of N_1 and N_2 is also an extension of the given inheritance network whose contents for `nsub` is the *union* of those contents for N_1 and N_2, and whose contents for the other non-kernel predicates (i.e., `sub` and `dsub`) is the *intersection* of those contents for N_1 and N_2.

Notice in particular the role played by Axiom 8 when this definition is applied: the intersection of the contents for c dsub^m d for given c and d will obtain the weakest one of the contributions from the two given extensions.

We shall use

Proposition 3. *The meet of two valid extensions of a correct inheritance network is valid.*

Proof. The proof follows easily by inspection of the eight axioms. □

4.5. Distinction between the predicates sub *and* dsub

The distinction between the predicates sub and dsub may at first seem redundant. Could we just identify those two, and dispense with Axiom 3?

The reason for the distinction between sub and dsub is technical. Let us first describe it from the point of view of a sequence of inference steps. Suppose the following information is given from the sources (doubt values omitted)

```
a sub b
b sub c
c sub d
d sub e
```

Both Axioms 1 to 8 and the simplified set of axioms will obtain that a dsub e is a conclusion, given that sub and dsub are synonyms in the simplified case. However, if Axioms 1 to 8 are used then the following is the only way of obtaining the intermediate steps towards the conclusion:

```
a dsub b
a dsub c
a dsub d
a dsub e
```

whereas with the simplified set of axioms there are several different such sequences, which may be thought of as proofs. The use of Axioms 1 to 8 obtains the same effect as the ascending construction of inheritance paths in path-based approaches.

Therefore, if one considers adding a defeater literal in order to block this conclusion, then one defeater is sufficient if Axioms 1 to 8 are used, for example nsub(b,c,d). In the simplified alternative one must add several defeaters, in order to stop all the possible ways of obtaining the conclusion.

The use of the distinction between the sub and dsub predicates has two advantages, therefore. It is one way (of several possible ways) of avoiding the combinatorial explosion when chaining along long paths, in those cases where such chaining is needed. Furthermore, and more importantly, it contributes to keeping down the number and the size of the defeater sets that must be assumed in order to obtain the intended default conclusions.

5. The Proportion Semantics

We now proceed to the definition of a formal framework for analyzing and motivating the axioms that were introduced in Section 4.2. We shall define an *underlying semantics* which defines whether a literal in an inheritance network *holds* in an underlying structure. This underlying semantics will be used as the basis for studying the properties of the axiomatic representation.

5.1. Underlying structures

We have already defined one semantical level where *propositions* are evaluated in an inheritance network consisting of literals. The underlying semantics is a lower level where *literals* in an inheritance network are evaluated in a more detailed structure and in a nontrivial way. In order not to confound the two evaluation levels, we shall use the terminology that a literal *holds in* the underlying structure, whereas a proposition is *satisfied* in an inheritance network according to earlier definitions. The term "the truth of" will be used in both cases.

An *underlying structure* is a fivetuple consisting of a nonempty finite object domain O, a nonempty finite set C of class names, a mapping M that assigns a nonempty subset of O to each class name, a *doubt scale factor* Φ which shall be a large positive number, and a *threshold K* that is a positive number $< -\Phi * \ln(0.5)$. For example, if $\Phi = 1000$ then K must technically be $< 693.14\ldots$, although in practice it should be much smaller. The constant K is, as before, the quantity that is used as the value of the constant K in propositions. Normally the domain O is a large set.

If U is an underlying structure then we write K_U for the threshold component of U.

If A and B are subsets of O and |A| is the cardinality of the set A then we define `prop(A,B)` as $|A \cap B|/|B|$. For example, if B has 10 members, and 8 of the 10 members of B are in A, then `prop(A,B)` = 0.8. The number of members in A is not relevant for this. Therefore `prop(A,B)` can be thought of as the conditional probability P(A|B).

Now consider an underlying structure $\langle O, C, \text{M}, \varPhi, K \rangle$, an inheritance network $\langle S, C, \varGamma, \Delta, \varLambda \rangle$, and a literal over C. This literal is said to *hold in* the underlying structure according to the following conditions:

- `c ⊑ d` holds iff $\text{M(c)} \subseteq \text{M(d)}$
- `c dj d` holds iff $\text{M(c)} \cap \text{M(d)}$ is the empty set
- `c sub`^{m}` d` holds iff

$$K \geqslant m \geqslant -\varPhi * \ln(\text{prop(M(d),M(c))})$$

- `c dsub`^{m}` d` holds iff `c sub`^{m}` d` holds
- `nsub(c,d,e)` does not hold iff

$$\text{prop(M(e), M(c)} \cap \text{M(d))} \geqslant \text{prop(M(e), M(d))}$$

For example, if $\text{M(c)} \subseteq \text{M(d)}$ then `prop(M(d),M(c))` = 1 and `c sub`^{0}` d` holds since $\ln(1) = 0$. If a few members of a not-so-small c are not in d then `prop(M(d),M(c))` decreases to a number $1 - \epsilon$ for a small positive ϵ. The corresponding doubt factor is then approximatively $\varPhi * \epsilon$ since $\ln(1 - \epsilon)$ is approximately $-\epsilon$ for small ϵ.

Notice that the singleton part of an inheritance network is not considered in this definition, for the reasons that were discussed in Section 2.5.

A *model* for an inheritance network is an underlying structure where all the network's literals hold.

The *reference network* for a given underlying structure with C as its set of class names is the inheritance network consisting of C and the set of all literals over C that hold in that structure. The set of singleton classes shall be empty. It follows that any underlying structure is a model for its reference network.

It follows immediately:

Proposition 4. *Let* $N = \langle S, C, \varGamma, \Delta, \varLambda \rangle$ *be an inheritance network having a model M, and let* $N^* = \langle \emptyset, C, \varGamma^*, \Delta^*, \emptyset \rangle$ *be the reference network of M. Then* $\varGamma \subseteq \varGamma^*$ *and* $\Delta \subseteq \Delta^*$.

These definitions disregard the singleton part of the inheritance network. The generalization to including singleton classes is straightforward, but omitted here since it does not affect the results in this article. The notation is prepared for future additional work.

5.2. Soundness of the axioms

In order to analyze the soundness of the restrictions, we extend the use of underlying structures from literals to propositions, using the reference network. A proposition P is said to be *satisfied in an underlying structure U* iff it is satisfied for K_U in the reference network for that structure.

Proposition 5. *Axioms* 1 *through* 8 *are satisfied in all underlying structures.*

Proof. For a given underlying structure, consider its reference network, and verify each of the axioms by considering its value there. The proof is trivial or next to trivial for all axioms except for Axiom 7. With respect to Axiom 4, the conclusion follows according to the condition on the value of K. Falsifying this restriction would require that d and e, which are disjoint sets, both are assigned more than half of the members of c in the underlying structure.

The validation for Axiom 7 depends on the definition for `nsub` of *holds* and can be characterized as an *assumption of at least equal proportion*. Used when `c sub`^{m}` d` and `d sub`^{n}` e`, the absence of `nsub(c,d,e)` is the assumption that the members of $\text{M(c)} \cap \text{M(d)}$ are divided between M(e) and its complement with at least the same proportion in M(e) as the members of M(d) are.

We first show the proof with respect to Axiom 7:2. Recall that this axiom is as follows:

```
7:2.  c dsub^k d ∧ d sub^m e ∧ e sub^n g ∧
           ¬nsub(c,d,g) ∧ ¬nsub(d,e,g) ∧ k+m+n ⩽ K
              → c dsub^{k+m+n} g
```

Consider an assignment of values to the variables in this formula where the antecedents of Axiom 7:2 are satisfied so that

```
c dsub^k d
d sub^m e
```

$$\text{e sub}^n \text{ g}$$
$$k + m + n \leqslant K$$

and so that `nsub(c,d,g)` and `nsub(d,e,g)` are not literals in the inheritance network at hand. We then have

$$K \geqslant k \geqslant -\Phi * \ln(\text{prop(M(d),M(c))})$$
$$K \geqslant m \geqslant -\Phi * \ln(\text{prop(M(e),M(d))})$$
$$K \geqslant n \geqslant -\Phi * \ln(\text{prop(M(g),M(e))})$$

Writing `C` for `M(c)` and similarly for `D`, `E` and `G` we obtain at once, and since $k > 0$,

$$K \geqslant m + n \geqslant -\Phi * \ln(\text{prop(E,D)} * \text{prop(G,E)})$$

However, since `nsub(d,e,g)` is not in the given reference network we also have `prop(G, D ∩ E) ≥ prop(G,E)`, and we obtain

```
prop(E,D) * prop(G,E) ≤ prop(E,D) * prop(G, D ∩ E) =
|D ∩ E|/|D| * |D ∩ E ∩ G|/|D ∩ E| = |D ∩ E ∩ G|/|D| ≤
|D ∩ G|/|D| = prop(G,D)
```

so that

$$K \geqslant m + n \geqslant -\Phi * \ln(\text{prop(M(g),M(d))})$$

An analogous argument is used for combining this conclusion with

$$K \geqslant k \geqslant -\Phi * \ln(\text{prop(M(d),M(c))})$$

which we observed above, obtaining

$$K \geqslant k + m + n \geqslant -\Phi * \ln(\text{prop(M(g),M(c))})$$

In the case of Axiom 7:2 it is therefore required to repeat the same argument two times. The proof for axiom variants 7:*n* for other values of *n* is entirely analogous. This concludes the proof. □

Proposition 6 (*Soundness of* `dsub`-*extension*). *If N is a consistent inheritance network then all literals in the minimal* `dsub`-*extension of N hold in all the models of N.*

Proof. All literals in *N* itself hold by definition in each of the models. Additional literals for `dsub` must have been enforced by Axiom 7 or Axiom 8. The result follows using Proposition 5. □

Notice that consistency has been defined in terms of satisfying the axioms, and not in terms of having a model, and that we do not have a result guaranteeing the existence of a model for an arbitrary consistent inheritance network. This is not a major issue, however, since inheritance networks are to be used for representing the actual situation in an application, and then the application itself is a model.

Appendix A discusses the existence of models for consistent networks in relation to the issue of completeness. In this context it also discusses properties and usefulness of circular subsumption structures.

6. Effects of Restriction 9 and Axiom 7

We have defined Axiom 7:1 and a stronger variant of it, called Axiom 7:2. Axiom 7:1 can be obtained by selecting `c = d` in Axiom 7:2 and using Axioms 1 and 5. The need for the stronger variant can be understood through the following example which first shows the reasons for using Restriction 9. Consider the following example:

```
C ⊑ RE
RE sub^k E
E sub^m GA
nsub(RE,E,GA)
```

Without Restriction 9 one can conclude `C dsub`^{k} `E` and `C dsub`$^{k+m}$ `GA`. This is not a desirable conclusion, since the `nsub` literal expresses that the other literals do not allow us to conclude that a randomly chosen member *c* of `RE` is

a member of GA with probability $\varphi(k+m)$. Therefore, and in the absence of additional information about C, we ought not to be able to conclude C dsub^{k+m} GA. Restriction 9 has the desirable effect of forcing nsub(C,E,GA) to be added to the accepted extension, which prevents the conclusion of C dsub^{k+m} GA. This is in line with the standard view in the literature on defeasible inheritance.

The above example is a well-known schema in the literature on defeasible inheritance, and is known as "Clyde, the Royal Elephant", and it is easy to find other, similar examples. However, one must also consider what is the effect of Restriction 9 if there is other information that will tend to override, directly or by way of inference, the conclusion that is obtained from Restriction 9.

In the direct case there is actually no problem. If C sub^n GA is added to the given inheritance network, then there is anyway no contradiction due to Restriction 9. This is because the occurrence of a literal with nsub does not contradict the corresponding literal with dsub, it merely prevents it from being inferred in a particular way.

A slightly more complex example is obtained by adding the following literals to the original ones.

    C ⊑ CE
    CE sub^j GA

The conclusion C sub^j GA is obtained in this example as well. The literal for nsub does not block it since its middle argument provides the required selectivity.

The situation is different if the direct link from CE to GA is replaced by a link from CE to E, as follows:

    C ⊑ CE
    CE sub^j E

The conclusion CE dsub^{j+m} GA follows in this case; it is not affected by the given and inferred nsub literals. The interesting question is whether C dsub^{j+m} GA should also be inferred. In the framework of the Proportion Semantics it is natural to discuss this in terms of probabilities. The example provides two parallel paths from C to E, namely, through CE and through RE. If the latter two classes almost coincide then one should not be able to infer C dsub^{j+m} GA. However, in this case one should also not be able to infer CE dsub^{j+m} GA, so the given network ought to include the literal

    nsub(CE,E,GA)

in order to be a correct representation of the application at hand, in accordance with the requirements on the domain model that were specified in Section 2.3. If it does not do so, and it does not allow that literal to be inferred using Restriction 9, then it should come as no surprise that unwarranted conclusions are obtained.

On the other hand, if the contents of CE are unrelated to the contents of RE, at least with respect to inclusion in GA, and more specifically if CE dsub^{j+m} GA is a warranted conclusion, then C dsub^{j+m} GA ought to be so as well.

This is the point where Axiom 7:2 is needed. Axiom 7:1 is sufficiently strong in very many of the situations where chaining of links is required, but the present case is an example where it is not sufficient: Restriction 9 forces nsub(C,E,GA) to be included in the extension, and this blocks the inference of C dsub GA both through RE and through CE using Axiom 7:1. However, Axiom 7:2 is able to bypass the restriction of nsub(C,E,GA) and to allow inferring the link from C to GA via CE.

We shall use the term *off-path preclusion* for a situation in an inheritance network where there is a sub path through some classes c, d, e and g and also a literal nsub(d',e,g) where d' is different from d. The nsub literal may preclude the use of the sub path, and the node d' is off-path. (Please notice that although somewhat related, this is not the distinction between on-path and off-path preemption systems.)

Unfortunately, Axiom 7:2 is not sufficiently strong for all situations; it fails in examples involving *double off-path preclusion*, as in the following example.

    C ⊑ RE
    RE sub^k E
    E sub^m GA
    nsub(RE,E,GA)
    C ⊑ CE
    CE sub^j E
    GA sub^n LA
    CE sub^p CA
    CA sub^q GA
    nsub(CA,GA,LA)

The first six literals are the same as above, and the following five literals adds another, similar off-path preclusion. There is an upward sub chain from C via CE, E and GA to LA (which could be interpreted as "land animal", for example), and the two successive nodes E and GA in that chain occur as the middle argument of nsub, which blocks the chaining of dsub at

that point. We have seen how Axiom 7:2 allows the inference system to pass by one such block, but it is not sufficient for passing two successive blocks in a subsumption chain.

At the same time, using the same argument as above, it is reasonable to expect the system to obtain $\text{C sub}^{j+m+n} \text{ LA}$ as a conclusion, since there is no indication of CE being related to RE, or of CA being related to E. It is easily seen that Axiom 7:3, constructed by analogy with Axiom 7:2 but with a chain of four sub literals in its antecedent instead of three, is able to obtain this conclusion.

By extrapolation, it would be appropriate to use the variant Axiom 7:n where n is the largest number of consecutive off-path preclusions that may occur in the inheritance networks being considered. If these networks are known to be cycle-free for sub and dsub literals then the maximum length of a chain of such literals will be sufficient, but it seems likely that a significantly smaller value of n will be sufficient in practice. Some empirical information on the frequency of multi-off-path preclusion situations in actual, large knowledge bases would therefore be of interest.

7. Inference operation

7.1. Inference of the defeater predicate

The analysis of Axioms 1 to 8 in Section 5 depended on an assumption that all applicable literals for the nsub predicate are already present in the given inheritance network. Section 6 discussed the use of Restriction 9 which allows some literals for nsub to be inferred from others, thereby relieving the user from the obligation to write out all occurrences of nsub explicitly. Restriction 9 has a heuristic and cautionary character: it is heuristic in the sense of not being formally sound according to the underlying semantics, and it is cautionary in the sense that it tends to suppress default conclusions when there is a good reason to suspect that the preconditions for drawing the conclusion are not present.

Traditional nonmonotonic reasoning uses another condition for inferring instances of a particular predicate, namely, in order to avoid inconsistencies. If a given set of propositions is inconsistent, then the inconsistency may be removed by adding literals for an *abnormality predicate*, assuming of course that the entire system is set up in such a way that adding those literals will suppress certain previously obtained conclusions. It is customary to add a minimal set of such literals under the requirement to add sufficiently many so that the inconsistency goes away. In our case the nsub predicate can serve as such an abnormality predicate, and addition of literals for nsub may turn an inconsistent inheritance network into a consistent one.

However, although removing inconsistency is essential from the point of view of classical logic, it does not in itself represent a need in the application. We propose that it is more constructive to think of the addition of abnormality literals as a *critical review process*, where the initially given information (the given inheritance network, in our case) is checked for indications of anomalies, and abnormality literals are added in a cautionary fashion in order to avoid making conclusions on dubious grounds.

Consider for example the case of *directly conflicting subsumers* which is more often known as the "Nixon Diamond". Using our notation it is:

```
N sub Q
N sub R
Q sub P
R sub NP
P dj NP
```

The annotation of doubt values is not of interest when considering this example and it has therefore been omitted. It is assumed that doubt values have been selected in such a way that the restriction using the constant K is not violated. The same will apply for many of the examples in the sequel.

When this example is used in the literature it is often assumed that the node N represents a singleton or a single individual, but here we consider the case where all the classes in the example are concept classes, including N. The singleton case will be discussed below.

In order for Axioms 3 and (any variant of) Axiom 7 to be satisfied, unless there are additional literals for nsub, a valid extension must contain both of

```
N dsub P
N dsub NP
```

which violates Axiom 4. This can only be avoided by adding one or both of the following

```
nsub(N,Q,P)
nsub(N,R,NP)
```

Both of these are possible because of the structural symmetry. Neither Restriction 9 nor any of the axioms will force either of these literals to be added, so some other mechanism is needed in order to avoid an inconsistency in this example. The

traditional approach is to allow two admitted extensions, one containing only `nsub(N,Q,P)`, the other containing only `nsub(N,R,NP)`, with the understanding that one or the other of these must be the case, but we do not know which.

The problem with this is the following. Since it has been stated that typical members of `N` are members of both `Q` and `R`, clearly there is something nonstandard about the situation. In this case, why should one exclude the possibility that the members of `N` are evenly distributed between `P` and `NP`? Maybe it is *neither* the case that most `N` are in `P`, nor that most `N` are in `NP`. Taking the intersection of the two permitted extensions will therefore obtain the right conclusion set (no conclusion concerning `N`) but for an unconvincing reason. From the "critical review" point of view it would be better to have a mechanism that adds *both* `nsub(N,Q,P)` and `nsub(N,R,NP)` in view of the recognized anomaly. This is similar to the position taken by traditional skeptical approaches to defeasible inheritance.

Moreover, a well-known and minor modification of this example results in a scenario where one actually obtains an unintended conclusion, and not merely an inappropriate reason for an intended conclusion. Consider

```
N sub Q
N sub R
Q sub P
R sub NP
P dj NP
NP sub IM
P sub IM
```

Here, IM may be read as "ideologically motivated". Two extensions are obtained, like before, and since `N dsub IM` is included in both of them, it is obtained as a conclusion of the given network. This is an example of a *floating conclusion* [35,53]. Since the given network does not state that `P` and `NP` are complementary classes, it is arguably an unintended conclusion.

This topic may also be discussed in terms of the Proportion Semantics. Consider an object domain O that is partitioned into two disjoint sets `P` and `NP` (we assume the same encoding of the scenario as above, and identify class names and their sets of members), assume two sets `Q` and `R` most but not all of whose members belong to `P` and `NP`, respectively, and consider all possible ways of choosing a set `N` most of whose members belong to `Q ∩ R`. Question: in what percentage of the cases will the set `N` belong wholly or almost wholly to `P`, the same for `NP`, and in what percentage of the cases will neither apply? It seems safe to assume that the 'neither' case will dominate strongly, which means that it would be appropriate to infer both `nsub(N,Q,P)` and `nsub(N,R,NP)`. The Defeater Inheritance operation does not do that, so in this respect it is not in line with the informal interpretation of the Proportion Semantics.

We shall return to the problem of floating conclusions in a later section.

One may also object against specific instances of this example, such as the 'Nixon Diamond' instance, on the grounds that the dichotomy between, for example, "pacifist" and "non-pacifist" is too simplistic, and that a person may take intermediate positions concerning the use of military force. However, this objection depends on the particular instance of the schema, whereas the previously mentioned objection applies for all scenarios involving classes with a substantial number of members.

7.2. Difficulties with the critical review stance

The critical review stance is attractive in principle but not so easy in realize consistently. One problem is that because of the nature of nonmonotonic reasoning, the addition of an abnormality literal that is made on a cautionary note may turn out to have the opposite effect of enabling a conclusion that would otherwise not have been made. Consider for example the following inheritance network.

```
c sub d
d sub e
e sub g
nsx(d,e,g)
g sub q
c sub m
m sub p
p dj q
```

The first four literals constitute an on-path preclusion situation; the first one plus the last four constitute a case of directly conflicting subsumers. The use of Restriction 9 disables the side of the "Nixon Diamond" that leads from `c` to `q`, thereby enabling the conclusion `c dsub p` in the derived extension. This is a cause of concern since the use of Restriction 9 serves a cautionary purpose.

Another example of the same kind is the Cascaded Ambiguities scenario proposed by Touretzky et al. in [57], and which can be written as follows in our representation:

```
N sub Q
Q sub P
N sub R
R sub NP
P dj NP
P sub AM
R sub FF
FF sub NAM
AM dj NAM
```

This scenario contains two structures with conflicting subsumer, one inside the other. The Defeater Inference operation does not produce either `N sub AM` nor `N sub NAM`, which is reasonable, but if both `nsub(N,Q,P)` and `nsub(N,R,NP)` are imposed then one branch of the outer diamond structure is disabled, and `N sub NAM` is obtained as a conclusion.

In view of these difficulties we shall proceed to study the properties of an inference operation that minimizes `nsub` as an abnormality predicate. This is done since it is in accordance with standard approaches in nonmonotonic reasoning in general, and in spite of the reservations about whether it is the best way to go in the long run. The use of the axioms is combined with the use Restriction 9 as a heuristic and cautionary restriction. We shall consider to what extent the effects of this inference operation are consistent with the Proportion Semantics and the critical review position.

7.3. The Defeater Inference operation

The *Defeater Inference operation* is an operation on a correct initial inheritance network that is defined as follows.

1. Consider all the `nsub`-extensions of the given inheritance network. Replace each of them by its minimal `dsub`-extension.
2. Remove those where Restriction 9 is not satisfied.
3. Restrict that set to those members that are minimal with respect to their contents for `nsub`, i.e., those for which no other member of the set contains a strict subset of literals for `nsub`. These will be called the *accepted extensions*.
4. Obtain the meet of all the accepted extensions. It will be called the *derived extension* under the Defeater Inference operation.

Proposition 7. *If N is a correct initial inheritance network then the derived extension of N under the Defeater Inference operation is valid.*

Proof. This follows immediately from Proposition 3. □

8. Issues in commonsense inheritance

The research literature on defeasible multiple inheritance contains different opinions about the proper handling of certain unusual situations, but there is a wide agreement about what are to be considered as commonsense conclusions for some standard types of situations or *schemas*. In this section we shall discuss several of these schemas from the following points of view:

- What conclusions are expected for the schema according to the existing literature.
- What are the contents of the derived extension of the schema according to the Defeater Inference operation.
- What ought to be the contents of the derived extension according to the Proportion Semantics and the critical review position.

8.1. Decoupling

The possibility of decoupling occurs if a class has a defeasible subsumption (directly or indirectly) to the lowest class in an instance of the schema of directly conflicting subsumers. For example, suppose the 'Nixon Diamond' example is augmented with one more class, becoming

```
N sub Q
Q sub P
N sub R
R sub NP
P dj NP
H sub N
```

Several published approaches require that there should still be two permitted extensions, namely one where both `H` and `N` are subsumed by `P`, and one where they are subsumed by `NP`. Some however allow four extensions, including those where `N` and `H` are assigned different subsumers. This situation is characterized as *decoupling* and is usually considered to be contrary to commonsense.

It is easily verified that the Defeater Inheritance operation obtains two of these extensions, i.e. it does not admit decoupling. This is because `nsub(N,Q,P)` implies `nsub(H,Q,P)` according to Restriction 9, and similarly for `nsub(N,R,NP)`.

However, in view of the earlier discussion about directly conflicting subsumers, it is debatable whether the case of decoupling is an issue at all. The class that may be decoupled (`H` in the example) is always subject to the argument that it may be evenly split between the proposed alternatives (`P` and `NP` in the example), just like its superior.

8.2. Special non-decoupling situations

In view of the strong acceptance of the schema of directly conflicting subsumers, one should anyway ask whether there are some special cases where a restriction against decoupling can be defended and is supported from an application point of view. One obvious case is when the class `H` is a singleton, but this case must be treated in the way discussed in Section 2.5.

Another such case is when there is additional knowledge to the effect that the members of a class tend to behave in the same way or have similar properties. Consider, for example, the situation in an electoral college consisting of several delegations, where the members of a delegation usually vote for the same candidate (by voluntary agreement, or due to the rules governing the process). One can easily think of given information of the form "the delegation from state S usually votes for the candidate from the party P", and there is a possibility of conflicting subsumers. If the uniform voting rule is strict or if the number of exceptions is moderate then it can be encoded as a proposition of the form

```
c sub R ∨ c sub D
```

(In order to allow domain knowledge to be expressed as propositions in general, it is necessary to extend the syntax so that class identifiers can occur in propositions.) It is in special cases like this that a restriction against decoupling may be applicable.

8.3. Choice of breakpoint

Several additional examples are shown in Appendix B, but two scenarios are particularly interesting and will be discussed here. Consider first the following abstract scenario:

```
A sub B
B sub C
C sub D
D dj G
A sub G
```

Here the use of the Defeater Inheritance operation obtains two permitted extensions, one containing `nsub(A,B,C)` and one containing `nsub(A,C,D)`. The extension containing `nsub(B,C,D)` will also contain `nsub(A,C,D)` by virtue of Restriction 9, so it is not `nsub`-minimal, which is significant since otherwise the conclusion `B dsub D` would be lost. Neither of the permitted extensions is preferred over the other. Therefore, `A dsub C` is not included in the resulting extension.

Informal application of the Proportion Semantics gives the same result. Because of the symmetry, the Proportion Semantics should not support one of these choices over the other.

The lack of support for the conclusion `A dsub C` in this example is different from what is obtained in the traditional, path-based approaches to multiple inheritance, and anyone who is used to these approaches may consider this to be a fault in the present approach.

We propose that it is not, however. Consider the following argument which is a kind of defeasible *reductio ad absurdum*: if `A` were subsumed by `C` in a normal way then it should have followed that `A` is subsumed by `D`, but we know that is not the case, therefore it is not possible to conclude that `A` is subsumed by `C`. It is not difficult to construct scenarios that instantiate the abstract scenario and where the counterpart of `A dsub C` does not hold, for example as follows:

```
CitizenOfGuyana sub LivesInLatinAmerica
LivesInLatinAmerica sub SpouseHispSpeaking
SpouseHispSpeaking sub HispanicSpeaking
HispanicSpeaking dj EnglishSpeaking
CitizenOfGuyana sub EnglishSpeaking
```

where `HispanicSpeaking` is the class of those having Spanish or Portuguese as their first language, and `SpouseHisp-Speaking` is the class of those persons whose husband or wife has either of these as their first language. We would not like to infer that most citizens of Guyana (which is an English-speaking country) are married to Hispanic-speaking persons.

The general question is where, in a chain of defeasible subsumption links, shall the chain be broken if it is inconsistent to use the entire chain. Traditional path-based methods have not taken this issue into account.

8.4. Stein's Floating Conclusions Scenario

The next example is due to Stein [53] and was used by [35] as one of their two examples of *floating conclusions*. It is more complex than the extended Nixon Diamond example and can be written as follows in our notation.

```
SeedlessGrapeVine sub GrapeVine
SeedlessGrapeVine sub SeedlessThing
SeedlessThing sub NotFruitPlant
GrapeVine sub FruitPlant
GrapeVine sub Vine
Vine sub ArborPlant
FruitPlant sub NotArborPlant
FruitPlant sub Tree
ArborPlant sub Plant
Tree sub Plant
```

(Obvious `dj` literals have been omitted.) According to the traditional analysis, each extension contains a path from `SeedlessGrapeVine` to `Plant`, but these paths are different in different extensions, and there is no such path that occurs in all extensions. The conventional wisdom is that the floating conclusion is an intended one.

This example illustrates several of the issues in the present topic, and it is therefore worthwhile to study in some detail how the Defeater Inference operation applies to it. The reader is encouraged to follow it up by drawing the corresponding diagrams.

Each extension must contain defeaters that avoid inconsistencies concerning whether `SeedlessGrapeVine` is in `FruitPlant` or not, and concerning whether `GrapeVine` is in `ArborPlant` or not. Therefore, each extension must contain one of the following two literals:

```
nsub(SeedlessGrapeVine,GrapeVine,FruitPlant)
nsub(SeedlessGrapeVine,SeedlessThing,NotFruitPlant)
```

and it must also contain one of the following:

```
nsub(GrapeVine,Vine,ArborPlant)
nsub(GrapeVine,FruitPlant,NotArborPlant)
```

For both members of the second group, Restriction 9 forces additional literals for `nsub`, namely, respectively:

```
nsub(SeedlessGrapeVine,Vine,ArborPlant)
nsub(SeedlessGrapeVine,FruitPlant,NotArborPlant)
```

This gives four combinations whose contents for `nsub` are as follows:

```
nsub(SeedlessGrapeVine,GrapeVine,FruitPlant)
nsub(GrapeVine,Vine,ArborPlant)
nsub(SeedlessGrapeVine,Vine,ArborPlant)
```

and

```
nsub(SeedlessGrapeVine,GrapeVine,FruitPlant)
nsub(GrapeVine,FruitPlant,NotArborPlant)
nsub(SeedlessGrapeVine,FruitPlant,NotArborPlant)
```

and

```
nsub(SeedlessGrapeVine,SeedlessThing,NotFruitPlant)
nsub(GrapeVine,Vine,ArborPlant)
nsub(SeedlessGrapeVine,Vine,ArborPlant)
```

and

```
nsub(SeedlessGrapeVine,SeedlessThing,NotFruitPlant)
nsub(GrapeVine,FruitPlant,NotArborPlant)
NSUB(SeedlessGrapeVine,FruitPlant,NotArborPlant)
```

All these are consistent and obtain valid extensions. Number two and number four entail

```
SeedlessGrapeVine dsub ArborPlant
SeedlessGrapeVine dsub Plant
```

Number three entails

```
SeedlessGrapeVine dsub FruitPlant
SeedlessGrapeVine dsub Tree
SeedlessGrapeVine dsub Plant
```

The first combination is the most interesting one since it blocks both paths from `SeedlessGrapeVine` to `Plant`. It is minimal with respect to contents for `nsub`, like the other three, and therefore

```
SeedlessGrapeVine dsub Plant
```

is not a consequence of the given inheritance network according to the Defeater Inference operation. In fact, `Seedless-GrapeVine` is neither a `FruitPlant` nor a `NotFruitPlant` in the first extension, and this is why this extension does not allow the conclusion that it is a `Plant`.

This effect is due to Restriction 9, since if that restriction is not imposed then the first extension can be simply

```
nsub(SeedlessGrapeVine,GrapeVine,FruitPlant)
nsub(GrapeVine,Vine,ArborPlant)
```

The case for this alternative could maybe be made informally as follows: grapevines are fruitplants and vines; they are not arbor-plants and this is normal for fruitplants but exceptional for vines. However, seedless grapevines are an exceptional kind of grapevines which are not fruitplants, and therefore we take them to be more genuine vines than grapevines in general, and in particular we take them to be arbor-plants.

This example should therefore be seen in the context of the discussion about directly conflicting subsumers in Section 7.1. If one takes the position that every singleton and every other class that has chains of upward `sub` links to both arguments of a `dj` literal must be subsumed by one of those arguments, then the present inference method fails to obtain an intended conclusion. If one takes instead a position of allowing that the class is not subsumed by either one of the disjunction arguments, then the Defeater Inference operation gives the correct result in the present example. The latter view is in line with an informal interpretation of the Proportion Semantics and we consider it to be the correct one.

It may be argued that the very distinction between "arbor-plant" and "not arbor-plant" implies that there cannot be a third possibility. In this case the problem is that the representation for inheritance networks that has been introduced so far does not allow one to represent the complement of a given class; it only allows to represent disjunction between classes. A generalization that does allow one to represent complement classes will be outlined in the next section. However, even so, the principle of the excluded third can at most apply to individuals and to singleton classes; it does not follow that it must apply to subsumed classes in general.

9. Object-level relations and amendments for description logic

Already the earliest work on inheritance networks addressed the representation of binary relations between the objects that are members of the classes described by the network [59]. Nantajeewarawat and Wuwongse [36] propose the use of the argumentation logic of [12] for representation problems that involve the combination of nonmonotonic inheritance and the use of binary relations.

Since our approach consists of a strict part and a defeasible part, using the $\sqsubseteq$ and `dj` operators and the `dsub` and `nsub` operators, respectively, it is straightforward to extend the strict part with additional constructs. In particular, the constructs of $\mathcal{ALC}$ description logic [2] provide useful additional expressivity, and they can be added easily. This extension is often useful for representing domain-specific knowledge, and its use is illustrated by three of the examples in Appendix B. The following technical details are required for these examples, and could be used as the beginning of an amalgamation of the current approach with a variety of Description Logic.

A number of binary relations $R_1, R_2, \ldots$ are assumed over the object domain O. The interpretation mapping M is extended so that for every relation symbol or *role* R_i, $M(R_i)$ is a subset of O $\times$ O.

The syntax for inheritance networks is modified as follows. A *term* is an expression that is formed in some of the following ways:

- Each class symbol is a term.
- If A and B are terms, then $A \sqcup B$, $A \sqcap B$ and $\neg A$ are terms.
- If A is a term and R is a role, then $\forall R.A$, $\exists R.A$ and $\forall^* R.A$ are terms.

The interpretation of these expressions is the standard one; the last mentioned term is defined as $\forall R.A \sqcap \exists R.A$, i.e., it is the class for all objects x for which $R(x, y)$ holds for at least one y, and all those y are members of A.

The definition of an *inheritance network* is now extended with one component so that it is a sixtuple $\langle S, C, T, \Gamma, \Delta, \Lambda \rangle$, S and C are like before, T is a finite set of terms over C containing C itself as a subset, and the last three elements are sets of literals using T instead of C.

In the definition of the Proportion Semantics, underlying structures are restricted to those structures where all members of T are assigned a nonempty set of members. The assignment for a nonatomic term is chosen in the standard way as a function of the assignments to the term's components. This means that if T should contain a term that by definition must have an empty set of members, then the network does not have any model. The interpretations of the monotonic predicates $\sqsubseteq$ and dj are extended in the obvious way to the case of nonatomic arguments.

Axioms 1 to 8, Restriction 9 and the Defeater Inference operation can all be used with this extended representation. Additional axioms are also needed; already the analysis of Stein's Floating Conclusions Scenario showed the need for an additional rule for terms of the form $\neg A$. A full treatment of this topic is beyond the scope of the present article, but it is interesting to note that even without the addition of further rules, this representation is sufficient for a number of anecdotal scenario examples from earlier articles on this topic. In particular, the cited article [36] contains two examples, i.e. the "campus residence" scenario and the "good math student" scenario. Appendix B shows how the present approach can represent these scenarios and that the same conclusions are obtained and how an inconsistency in one part of an inheritance network is kept local and does not damage inference in other parts of the network.

We shall just make a few observations with respect to possible inference rules. Because of the importance of the disjointness predicate in our approach, the following rule is often needed when working examples:

$$A \text{ dj } B \rightarrow \forall^* R.A \text{ dj } \forall^* R.B$$

Since $(A \sqsubseteq B \sqcap C)$ follows from $(A \sqsubseteq B)$ and $(A \sqsubseteq C)$, one may be tempted to believe that the following, similar restriction should also hold:

10. $A \text{ dsub}^m B \wedge A \text{ dsub}^n C \rightarrow A \text{ dsub}^{m+n} B \sqcap C$

Actually this is merely approximately correct, as shown by the following example. Suppose 80 percent of the members of A are members of both B and C, 10 percent are only in B, 10 percent are only in C, and the value of the constant K is so large that chaining is admitted. The above formula would imply that 81 percent of the members of A are in $A \sqcap B$, which is too strong. A correct rule can be obtained by modifying the above rule so as to use another operation rather than addition.

10. Alternative approaches and discussion

The research topic of defeasible inheritance has several synonyms, such as multiple inheritance (with defaults), non-monotonic inheritance, and more. The literature on this topic is quite extensive, but with a concentration around the 1990's. Most work in this area has used networks with two kinds of links, namely positive and negative inheritance links where both kinds are defeasible. Inference has usually been defined either algorithmically, or by translation to a variety of default logic. Besides these major approaches there has been some work in more recent years using conditional logic, qualitative probabilities, and argumentation systems.

The approach that has been used in the present work represents a radical departure from these earlier ones, both through the use of a doubt index and through the introduction of the defeater predicate nsub. A comparison of our approach with each of the earlier ones on the logical, semantic or algorithmic level is clearly not possible in the framework of this article. However, we propose that it is just as important to check a new approach for how it behaves on actual, significant examples, as to compare with earlier work on the formal level. The present section contains an admittedly incomplete review of some earlier work, where we have favored earlier work that do present their solutions for interesting scenario examples. Besides Section 8 above, Appendix B discusses a number of standard examples from the point of view of our approach.

Many early approaches to defeasible inheritance specify a computational inference procedure, in particular path-based approaches. Other approaches have been logic-based in one sense or another, and these commonly use an ordering according to "priority" or "rank". This way of resolving conflicts between defaults has a long tradition in nonmonotonic reasoning, and was in fact mentioned already in [45]. In the case of defeasible inheritance one may consider using a rank ordering on models of a given set of defeasible links as a basis for defining the nonmonotonic entailment relation, or one may define a priority ordering directly on the set of defeasible links.

Early, general overviews of the topic of multiple defeasible inheritance have been made by, for example, Pollock [37] and Etherington [15]. The logical or mathematical basis for such knowledge representation has been investigated by, for example, Thomason and Horty [55], Sandewall [46], and Simary and Loui [52].

10.1. Conventional nonmonotonic logics

In the early days of the development of nonmonotonic logic there were a number of proposals to represent defeasible inheritance directly in some of the available logics. The use of default logic for this purpose, either in its original form or

with a priority ordering on the default rules, was proposed in particular by Reiter and Criscuolo [43,42], Etherington and Reiter [14], Haugh [26], Brewka [7], and Delgrande and Schaub [11].

Gelfond and Przymusinska proposed the use of autoepistemic logic for the same purpose [18].

It is straightforward to represent simple cases of defeasible inheritance in logic programming, as has been observed by many authors and users of this programming language, but the treatment of less simple cases requires a systematic investigation like for all the other approaches. Geerts and Vermeir have studied the treatment of specificity in such logic programming representations [25].

10.2. Conditional knowledge bases

As introduced by Kraus, Lehmann and Magidor, a *conditional knowledge base* is defined as a set of *conditional assertions,* each of which is a pair of formulas [32]. For example, the conditional assertion *bird* $\mid\sim$ *flying* where *bird* and *flying* are proposition symbols will express that birds normally fly. In comparison with the approach of the present article, conditional knowledge bases lack the use of a defeater predicate and the possibility of expressing a level of doubt, but they enjoy additional expressivity in the sense that the two components in a conditional assertion may be composite formulas, and not merely proposition symbols. That additional expressivity could be matched by the extension of our approach in the direction of description logic that was outlined in the previous section.

Work on conditional knowledge bases has focused on the abstract logical properties of entailment in such structures [32] as well as their computational complexity properties [13]. We have not been able to find an account of what conclusions are obtained from conditional knowledge bases for any but the most trivial of the standard scenario examples of defeasible inheritance.

10.3. Qualitative probabilities

Goldszmidt and Pearl have proposed an approach where defeasible propositions are characterized using *infinitesimal probabilities* [19,21], see also [20]. The basic idea is to assume that each proposition in the given domain model or "knowledge base" is associated with a conventional probability, to re-express those probabilities using polynomials in a base b, and then to merely characterize each proposition by the rank of the first nonzero term in that polynomial. Technically this has the same effect as allowing b to go towards zero. This obtains a rank ordering of the propositions which in turn is used for defining the entailment relation: one uses propositions up to, but excluding the rank level where a contradiction is obtained. The relevance of this approach for concrete situations involving non-infinitesimal probabilities remains to be clarified.

Most of what was said above about conditional knowledge bases applies as well to the qualitative probabilities approach to defeasible inheritance.

10.4. Other approaches to the semantics of defeasible inheritance

Besides those mentioned in the previous subsections, there are also several other earlier proposals for defining the semantics of defeasible inheritance networks, for example by Boutilier [6], by Krishnaprasad et al. [31], by Doherty [10], by Schlechta [48], by Delgrande [8], and by Pollock [39]. The purpose of these semantics has been to clarify the meaning of a semantic network, or to define an entailment relation in semantical terms, or to characterize the differences between different approaches to semantic inheritance. They have not been used in conjunction with an axiomatic characterization of defeasible inheritance as such, or for analyzing the latter. Since they do not use a doubt index, and since they use negative arcs instead of defeater literals it would not have been possible to use one of these definitions in conjunction with the present work.

10.5. Priority-ordered links

Several authors have proposed the use of a priority ordering on default statements in order to resolve ambiguities. Baader and Hollunder [4] used this approach in the context of description logic. Goldszmidt and Pearl [21] showed how to obtain such a priority ordering automatically from the given network using the qualitative probabilities approach. Heymans and Vermeir [30] proposed a representation of nonmonotonic inheritance problems using a standard first-order logic with the addition of a priority ordering on the restrictions. Horty [28] uses (simple) examples of multiple inheritance to illustrate his approach to defaults with priorities. The early contribution of Touretzky [58] followed a similar approach.

Applications of this kind can sometimes be re-expressed in our approach without the use of a priority ordering and through the introduction of a small number of additional nodes in the inheritance network. There is no result so far concerning the range of situations where this transformation is possible. However, the (single) example used in the article by Heymans and Vermeir can in fact be rewritten in this way, as shown by the Juvenile Offender example in Appendix B.

Morgenstern [34] introduces *formula-augmented inheritance networks* where nodes in the network can have logic formulas attached to them. In her case, the priority order is also used for resolving conflicts between inherited, attached formulas, which may occur even though there is no path-level conflict in the network.

10.6. Defeasible inheritance in description logics

There have been a number of proposals for extending description logics with the possibility of defeasible conclusions, beginning with the work of Padgham and Zhang [41] and of Straccia [54]. Other contributions have been made e.g. by Baader and Hollunder [4], Donini et al. [9], Rosati [44], and Bonatti et al. [5].

A comprehensive review of these proposals is beyond the scope of the present article. Let us however comment briefly on the last-mentioned article, where Bonatti, Lutz and Wolter propose the use of circumscription together with concept-based (class-based, in our terms) abnormality predicates. This proposal differs from the previous articles since those have in most cases been based on default logic or on epistemic operators. The proposed approach is illustrated by the following introductory example, in their notation

$$\text{Mammal} \sqsubseteq \exists \text{habitatLand} \sqcup \text{Ab}_{\text{Mammal}}$$
$$\text{Whale} \sqsubseteq \text{Mammal} \sqcap \neg \exists \text{habitatLand}$$

Minimization of $\text{Ab}_{\text{Mammal}}$ obtains the desired conclusions. This approach necessitates the introduction of a large number of abnormality predicates. In order to handle representational problems such as specificity, it allows the definition of a priority ordering on these predicates. Usually this ordering coincides with the subsumption hierarchy.

However, the main topic of the article is not representational issues such as these, but the computational complexity of the proposed method.

One potential problem with the proposed approach is that there is likely to be a need for several abnormality predicates for each class. Mammals, for example, have a number of characteristic properties, and for many of them there are exceptions. The representation shown above will have the effect that a subclass that is exceptional with respect to one of these properties is disinherited with respect to all the others as well. The natural way to solve this would be to add a second parameter to the abnormality predicate, for instance as follows.

$$\text{Mammal} \sqsubseteq \exists \text{habitatLand} \sqcup \text{Ab}(\text{Mammal}, \text{habitatLand})$$

However, the representational and computational ramifications of such a change remain to be studied.

There is a certain resemblance between this work and ours inasmuch as both are using preference-based approaches. However, the description-logic basis of the work of Bonatti et al. leads them to "single-argument" techniques where information is attached to nodes in the network, both with respect to the abnormality predicate and with respect to the separate priority ordering for obtaining specificity. In our approach we begin by introducing predicates of two and three arguments which merely have network nodes (classes, concepts) as some of their arguments. This is a more open representation which has made possible both the three-argument defeater predicate, with its considerable expressivity, at the same time as the extension in the direction of Description Logic.

10.7. Inheritance in Semantic Web systems

Description-logic based systems in the Semantic Web area are beginning to take an interest in defeasible inheritance; see May and Kandzia [33], Bernstein and Grosof [3], Ferndriger et al. [17], and Hosain and Jamil [27].

10.8. Argumentation systems

Pollock [38] and Dung [12] have proposed the use of argumentation systems for defeasible reasoning, and in particular for defeasible inheritance. For an overview of this topic, please refer to the article by Prakken and Vreeswijk [40]. This approach differs strongly from other approaches to defeasible inheritance since the others, in spite of their differences, all view the problem as one of defining what are the conclusions from a given set of premises.

10.9. Reactive diagrams

Reactive diagrams [24] is a representation system that is very general, and which in particular claims to be able to express argumentation systems as well as inheritance diagrams. Inference in inheritance diagrams is determined through a complex inductive algorithm on paths between pairs of points in the graph. This approach is similar to ours with respect to the use of defeater links (links that invalidate other links), but it differs in other aspects: there is no underlying semantics, and no use of doubt indices.

10.10. Skeptical inheritance

Skeptical theories of inheritance are of particular interest since they produce a single solution, avoiding the need to combine accepted extensions, and thereby also avoiding the issue of floating conclusions. Skeptical inheritance approaches

have been investigated by Grégoire [22,23] and by Simonet [50]. The logical treatment of skeptical approaches to defeasible inheritance appears to be more difficult than for credulous approaches, however.

The approach that we have proposed in the present article differs from the approaches mentioned here in a number of respects, in particular, the annotation of subsumption links with doubt values, the introduction of a set of restrictions for specifying valid conclusions, and the use of a formal semantics for validating the restrictions.

11. Conclusion

The main results of this article are as follows. The use of doubt annotated links and of defeater literals have added important expressivity to defeasible inheritance networks. An axiomatic representation has been proposed for defining what are valid conclusions from a given inheritance network, consisting of a set of axioms and an additional restriction which has a heuristic and cautionary character. The Proportion Semantics has been defined, and the axioms have been shown to be sound with respect to this semantics. A possible approach to proving completeness has been described.

The Defeater Inference operation has been defined, based on the use of the axioms, the additional restriction, and a nonmonotonic operation of minimizing the extent of the defeater predicate. The motivation for, and the effects of this inference operation have been discussed for a number of important configurations of nodes in inheritance networks. The operation obtains reasonable and intended results in most cases, but not in all of them.

The definition of an underlying semantics and a set of axioms, and the verification of the soundness of the latter with respect to the former provides a foundation for the analysis of a variety of nonmonotonic inference methods for defeasible inheritance. The present article contains the first one of several possible such analyses, but it also suggests a number of directions for continued work. The following are some important topics:

- Alternative or additional heuristic restrictions, as well as modifications of the preference relation that simply minimizes `nsub`. There still remain cases where the present definition does not produce reasonable results, in particular for some cases of floating conclusions.
- Extending the representation with a modal "possibly" operator. For scenarios with directly conflicting subsumers, such as the Nixon Diamond scenario, this would make it possible to express that both the conflicting paths are possibly valid, which should help to avoid the problem shown in the Cascaded Ambiguities scenario in Section 7.2.
- Combination of the present approach with Description Logic.
- Further axioms or heuristic restrictions that apply to singleton classes.
- The use of propositions for expressing domain-specific information, and not merely for expressing axioms and heuristic restrictions.
- Robustness in the face of local inconsistencies. The solution for the "good math student" scenario in Appendix B exhibits such robustness, and it would be of interest if this is a general property of the approach.
- Properties of inheritance networks with circular structures.
- Completeness of the proposed set of axioms, or amendments to the axioms that achieve completeness.
- Algorithms for inference in defeasible inheritance networks using the approach proposed here.
- Empirical investigation of the occurrence and frequency of difficult cases for defeasible inheritance in actual knowledge bases, and in particular the occurrence of multiple off-path preclusion.

Appendix A. Axiom completeness

This appendix describes a possible approach to obtaining a completeness proof for the set of axioms that were introduced in Section 4.2.

A.1. Continuous underlying structures

A completeness result that matches the soundness result in Proposition 6 should in principle have the following form: If a literal for `dsub` holds in all the models of a consistent inheritance network N then that literal is a member of the minimal `dsub`-extension of N. This means that the underlying semantics does not allow any other conclusions with respect to subsumption than those that are obtained in the minimal extension. We cannot provide such a result, but shall make some remarks about the possibility of obtaining a completeness result.

A plausible strategy for a completeness proof is to construct, for a given, consistent inheritance network, a model where the *only* `dsub` literals that hold are the ones in the minimal `dsub`-extension of the network. We shall use the term *supermodel* for such a model. Such a construction would also solve the question that was left open above, concerning whether every consistent inheritance network has a model in the underlying semantics.

A plausible way of constructing a supermodel is to proceed iteratively, starting with a network consisting of one single class, and adding more classes successively, thereby obtaining an induction proof. This suggests that the network will have to be cycle-free in order for the proof to be feasible.

There is however a technical problem since the underlying semantics interprets classes as finite sets of objects, which means that proportions are rational numbers which are then converted to real numbers through the logarithmic transfor-

mation. For an arbitrary inheritance network it will therefore not necessarily be possible to find appropriate integers for the cardinalities of the various classes involved. The problem remains even if one should restrict doubt indices to being integers.

One possible way of solving this problem is to allow each object to have a "mass" that is a positive real number, and to consider the proportions between the total mass of the objects in different classes, instead of just the number of objects. However, this approach can be simplified without loss of generality by omitting the objects altogether, and just assign mass to combinations of classes. The following definitions realize this approach.

A *continuous underlying structure* is a fourtuple consisting of a nonempty finite set C of class names, a mapping W (for weight) that assigns a non-negative real number to each member of 2^C while in particular assigning 0 to the empty set, and a *doubt scale factor* Φ and a *threshold K* that are defined like for the discrete underlying structures that were defined above.

A *fragment* in an underlying structure is a nonempty subset of the concept class domain in that structure. For discrete underlying structures that were defined above, the *weight* of a fragment is the set of objects that belong to exactly those classes that are included in the fragment. The *mass* of a fragment is the set of objects in all supersets of the fragment, including the fragment itself. Therefore the mass of a set of one class equals the set of members of that class, and the mass of a fragment in general is the intersection of the members of the classes in the fragment.

These concepts will now be applied to continuous structures, where each fragment has mass without having a set of members. For example, if the concept class domain C has three members a, b, and c, then $W(\{a, b\})$ shall represent the weight of the mass that is both in a and in b, but that is not in c. Similarly, the total mass for a is $W(\{a\}) + W(\{a, b\}) + W(\{a, c\}) + W(\{a, b, c\})$.

In general, consider a continuous underlying structure $\langle C, W, \Phi, K \rangle$ and let c be a member of C. The *mass of c*, written $\mathrm{m}.\{c\}$, is defined as the sum of $W(z)$ for all $z \in 2^C$ that have c as a member. If $y \in 2^C$ then $\mathrm{m}.y$ is defined as the sum of $W(z)$ for all $z \in 2^C$ where $y \subseteq z$.

Consider now in addition an inheritance network $\langle S, C, \Gamma, \Delta, \Lambda \rangle$, and a literal over C. This literal is said to *hold in* the continuous underlying structure according to the following conditions, which are the natural analogues of the definitions for discrete underlying structures:

- `c dj d` holds iff $W(z) = 0$ for all z that have both `c` and `d` as members
- `c ⊑ d` holds iff $W(z) = 0$ for all z where `c` is a member and `d` is not a member
- `c sub`^{i} `d` holds iff

$$K \geqslant i \geqslant -\Phi * \ln \mathrm{m}.\{c, d\}/\mathrm{m}.\{c\}$$

- `c dsub`^{i} `d` holds iff `c sub`^{i} `d` holds
- `nsub(c,d,e)` does not hold iff

$$\mathrm{m}.\{c, d, e\}/\mathrm{m}.\{c, d\} \geqslant \mathrm{m}.\{d, e\}/\mathrm{m}.\{d\}$$

Definition. A consistent inheritance network $\langle S, C, \Gamma, \Delta, \Lambda \rangle$ is said to be *wellformed* iff it satisfies the following conditions:

1. Its contents for the predicates `sub` and `dsub` do not contain any cycles.
2. For each of the literals `nsub(c,d,e)` in the network, there exists a path of literals for `sub` or `dsub` that proceeds from `c` to `d` and then on to `e`.

The first condition is essential; the second one will facilitate a proof but has no practical effect.

A continuous underlying structure $\langle C, W, \Phi, K \rangle$ is said to be a *supermodel* of a wellformed inheritance network $\langle S, C, \Gamma, \Delta, \Lambda \rangle$, iff it is a model for that inheritance network, and in addition the following conditions hold:

1. $W(z)$ shall equal 0 if and only if z is the empty set, or it has as members two classes c and d for which `c dj d` is a literal in Γ, or there is a literal `c ⊑ d` in Γ where $c \in z$ and $d \notin z$.
2. If Δ contains a literal of the form `c sub`^{k} `d` for a particular value of k but not for any smaller value, then the proportion of c in d shall be *exactly* $\varphi(k)$.
3. If Δ does not contain any literal of the form `c sub`^{k} `d` then the proportion of c in d shall be strictly less than 50 percent.
4. If `c sub d` and `d sub e` are in Δ and `nsub(c,d,e)` is not there, then the proportionality requirement must apply with equality.

Proposition 8. *If the minimal extension N^* of a consistent inheritance network has a supermodel U, then N^* contains all the literals for* `dsub` *that hold in U.*

Proof. This follows immediately from the definitions. $\square$

Hypothesis 9. The minimal extension of a wellformed inheritance network has a supermodel.

Discussion. The first question with respect to this hypothesis is whether the available degrees of freedom when constructing a proposed hypermodel is sufficient for admitting the number of constraints that can be imposed by the doubt indices of literals for sub, as well as by other literals. A quick calculation shows that this is in itself not a problem.

The next question is whether there exists a solution to the equations for assigning appropriate values to all variables, and whether the solution will be acceptable in particular by assigning non-negative mass to all the fragments. This can be verified with moderately cumbersome calculations for relatively simple configurations of subsumption links, but a general proof has not been found.

Proposition 10. *If Hypothesis 9 holds, and if a literal for* dsub *holds in all the continuous models of a wellformed inheritance network N, then that literal is a member of the minimal* dsub*-extension N* of N.*

Proof. Each model for N^* is also a model for N. Since the literal holds in all the continuous models of N^*, it must also hold in its supermodel. According to Proposition 8, only the members of N^* hold in the supermodel. □

A.2. Completeness using discrete underlying structures

If continuous underlying structures are used, then a completeness proof by induction over subsets of the concept class domain may use a strategy of successive decomposition of the mass assignment to model fragments. This solves the discreteness problem that was described above, but it is anyway natural to ask whether there is a way to adapt such a proof for the original, discrete underlying structures. This is in fact possible although at the expense of much additional bookkeeping. The method is as follows. First, require each doubt index to be the logarithm of a rational number, so that the corresponding probabilities can be written as a fraction between two integers. (Alternatively one could decide to write the doubt indices directly as proportion ratios and not as their logarithms.) This is an insignificant restriction from the application point of view.

A proof of the counterpart of Hypothesis 9 can then proceed in a similar way as in a proof for the original proposition. The difference is that in the original proof one can split each mass fragment into two parts when one more network node is taken into consideration in an induction step, but in the proof for the discrete case it is necessary to duplicate the space of all the objects in a way where every object obtains the same number of clones. By selecting the cloning factor it becomes possible to obtain a number of clones in each fragment that can be partitioned in exactly the proportion that is required by the rational number obtained from the doubt index. The total number of objects at the completion of this process is going to be extremely large, of course, but there is no difficulty in this besides the bookkeeping overhead.

A.3. Cyclic networks

The proposed approach to finding a proof for Hypothesis 9 depends on the assumption of a noncyclic inheritance network. Before trying to remove this restriction it is appropriate to consider whether cyclic networks are of interest at all.

For networks containing one or more cyclic structures formed by sub literals, two different situations arise depending on whether there is also an nsub literal whose arguments are in the cyclic structure in question, and in the direction of the sub chain. We shall refer to these as *unbroken* and *broken* cyclic structures, respectively.

Unbroken cyclic structures may occur if two or more classes represent strongly overlapping sets of objects. Networks that contain some unbroken cycles but no broken cyclic structures appear to behave similarly to cycle-free networks, or else they fail to have any models. The following schema for a correct but cyclic inheritance network is a marginal example which requires extreme choices of parameters in order to have a model. The example is given without the doubt indices:

```
a sub b
b sub a
a sub c
b sub d
c dj d
nsub(a,b,d)
nsub(b,a,c)
```

In order to construct a model for it, one must also select the values of the doubt index of each sub literal, and K must be selected in such a way that Axiom 5 is not violated. A model where all four occurrences of sub may use the same doubt index can be constructed as follows. Let a and b have an equal number of members, let two-thirds of their members be in their intersection, and let half of that intersection be part of c, and the other half be part of d. Finally, let c also contain those members of a that are outside the intersection, and similarly for d and b. This will be a model for the given inheritance network if all its doubt indexes are $-\Phi * \ln(0.33\ldots)$ or larger. Therefore, the given schema has a valid instantiation if $K = -\Phi * \ln(0.4)$, for example, but not with e.g. $K = -\Phi * \ln(0.2)$.

It is easily seen that another allocation of objects in the classes may allow a smaller doubt index in some of the literals, but at the expense of requiring a bigger doubt index in some others. This means that the definitions shown here represent, at least, a local minimum for the maximal doubt index that is required for having a model.

Recall however that a more plausible value for K would be $-\Phi * \ln(0.02)$, so for practical purposes no inheritance network with this schema has any models.

If Restriction 9 is applied to this example then one obtains at once

```
nsub(b,b,d)
```

which contradicts Axiom 5. (Notice that this does not contradict the soundness of the axioms, since Restriction 9 is not included among them.)

The case of broken cyclic structures is also problematic, as the following example shows.

```
a sub b
b sub c
c sub d
d sub a
nsx(a,b,c)
```

Restriction 9 forces the following additions

```
nsub(d,b,c)
nsub(c,b,c)
nsub(b,b,c)
```

which again contradicts Axiom 5.

These examples suggest that if cyclic structures are admitted at all in inheritance networks, they should be confined to clusters of classes within which literals for `nsub` are not asserted and cannot be inferred.

Appendix B. Issues and examples

The following are the details for a number of previously used scenario examples. The doubt annotations have been omitted, with an understanding that the doubt values are supposed to be within what is admitted by the constant K being used.

B.1. The Double Diamond

The Double Diamond structure is a well-known problem for directly skeptical approaches; it was described in [29] and was used in [35] to illustrate the phenomenon of *zombie paths*. This problem is not directly relevant for our approach since it is not a directly skeptical one, but it may anyway be of interest to check how our approach can handle it. The following is the abstract scenario:

```
A sub T
A sub P
T sub S
P sub notS
S sub R
P sub Q
Q sub notR
S dj notS
R dj notR
```

It is clear that each extension must contain either `A nsub S` or `A nsub notS`. In the former case, `A dsub Q` and `A dsub notR` are obtained, and there is no reason to block them. In the latter case, one must add one of `A nsub R`, `A nsub Q`, `A nsub S` or `A nsub notR`. The case of `A nsub S` is not of interest since a strictly smaller (by the subset predicate) set of nsub literals was obtained in the first alternative.

The derived extension therefore does not contain any conclusion beyond what is in the given inheritance network, not even `A dsub Q`, for reasons similar to the Guyana example in Section 8.3. This is well in line with the cautionary stance.

B.2. Simonet's Γ'' scenario

This scenario was used by [51] in order to show the reasons for one aspect of the approach discussed there.

```
A sub B
B sub C
C sub E
A sub D
D sub C
D sub ¬E
E dj ¬E
```

It is clear that every extension must contain `nsub(D,C,E)`. Restriction 9 requires the addition of `nsub(A,C,E)`. However, Axiom 7:2 still allows to conclude `A dsub G` obtaining a contradiction, so it is necessary to also add one of the following:

```
nsub(A,D,¬E)
nsub(A,B,E)
nsub(B,C,E)
```

The derived extension therefore does not contain any subsumption literals besides the given ones. One may consider it as strange that `B sub E` is not even obtained. However, consider by comparison the inheritance network that is obtained by removing `D sub C`, that is,

```
A sub B
B sub C
C sub E
A sub D
D sub ¬E
E dj ¬E
```

In this case it is clear that `B sub E` shall not be obtained as a conclusion, for the same reasons as in the Guyana example. If this is accepted, then there is no reason why the addition of the literal `D sub C` should make it possible to infer `B sub E`.

B.3. On-path versus off-path preemption

The issue of on-path versus off-path preemption was raised by an article by Sandewall [47] and a response by Touretzky, Horty and Thomason [57]. Additional contributions have been made e.g. by Al-Asady and Narayanan [1]. It concerns the proper treatment of inheritance networks with the following structure:

```
C sub RE
C sub CE
RE sub E
CE sub E
E sub GA
RE sub WA
WA dj GA
```

The question that has been debated is whether one should also be able to infer `C sub WA` or not, in our representation. More commonly, since most earlier systems use a type of link expressing "most A are not B", this is usually a question whether "most C are not GA" shall be inferable or not. So-called *off-path preemption* systems allow that conclusion whereas *on-path preemption* systems do not.

It is clear that `nsub(RE,E,GA)` has to be added in order to avoid a contradiction. The present example is therefore an extension of our earlier off-path preclusion example in Section 6, the difference being that the `nsub` literal was a given one there, whereas here it has to be inferred in order to avoid an inconsistency. We showed in Section 6 how the literal `nsub(C,E,GA)` is obtained by Restriction 9, and how it was anyway possible to infer `C sub GA` using Axiom 7:2 and the path through CE. This possibility is retained in the present scenario, which leads to a contradiction unless an additional `nsub` literal is assumed. There are several possibilities for the choice of that literal:

```
nsub(C,CE,GA)
nsub(CE,E,GA)
nsub(C,RE,WA)
```

which means that neither `C sub WA` nor (of course) `C sub GA` is among the consequences according to the Defeater Inference operation. This is in line with traditional on-path preemption systems.

Unfortunately, comparing the scenario in Section 6 with the extension shown here, it turns out that the literal `CE sub GA` which was a consequence before, is lost when the node `WA` and the links to it were introduced. This effect is not reasonable, and it is one of the reasons why there is need for additional refinement of the Defeater Inference operation.

B.4. Heymans's "juvenile offender" scenario

Proposed by Heymans and Vermeir [30], this scenario is defined as follows.

```
GuiltyOfCrime sub ToBePunished
IsMinor sub ¬ToBePunished
Bill ⊑ GuiltyOfCrime
Bill ⊑ IsMinor
```

Heymans and Vermeir propose a priority ordering on the first two literals. In our approach the desired result can be obtained by instead adding the following literal:

```
GuiltyOfCrime ⊓ IsMinor sub ¬ToBePunished
```

which is a natural way of expressing the exception to the first literal in the scenario.

B.5. Nantajeewarawat's "campus residence" scenario

Proposed by Nantajeewarawat and Wuwongse [36], this scenario involves whether on-campus students normally stay in the Northern Dormitories or the Western Apartments. It can be expressed as follows in our approach.

```
Married ⊓ Employee sub InWestApts
Male sub InNorthDorms
Tom ⊑ Married
Tom ⊑ Employee
Tom ⊑ Male
```

In a way similar to the previous example, the desired priority can be obtained by adding the literal

```
Married ⊓ Employee ⊓ Male sub InWestApts
```

B.6. Nantajeewarawat's "good math student" scenario

Proposed by Nantajeewarawat and Wuwongse like the previous one [36], this scenario provides an example of a circular, defeasible inconsistency. It concerns a campus situation where a student with grade B in math counts as a good student, and therefore normally majors in math if math is her favorite subject; math majors tend to be (or become) grad students, and grad students in math with grade B have average ability rather than good. The formalization is as follows. The literal concerning music does not have a counterpart in the source article and has been added here.

```
MathGradeB sub AbilityGood
FavoriteMath ⊓ AbilityGood sub MathMajor
MathMajor ⊓ MathGradeB ⊓ GradStudent sub AbilityAverage
Tom ⊑ GradStudent
Tom ⊑ FavoriteMath
Tom ⊑ MathGradeB
AbilityGood dj AbilityAverage
FavoriteMath sub PracticesMusic
```

The following conclusions are candidates for being obtained:

```
Tom dsub AbilityGood
Tom dsub FavoriteMath ⊓ AbilityGood
Tom dsub MathMajor
Tom dsub MathMajor ⊓ MathGradeB
Tom dsub MathMajor ⊓ MathGradeB ⊓ GradStudent
Tom dsub AbilityAverage
```

which is defeasibly inconsistent. This inconsistency can be met by blocking any of those conclusions, using an `nsub` literal, leading to six separate permitted extensions. The derived extension does not contain any of those candidate conclusions.

Notice, however, that the conclusion `Tom dsub PracticesMusic` is present in all permitted extensions, so it is also present in the derived extension. This is an example where the inference mechanism is robust with respect to local inconsistencies.

References

[1] R. Al-Asady, A. Narayanan, More notes on 'a clash of intuitions', in: Proceedings of the IJCAI Conference, 1993, pp. 682–687.
[2] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, P.F. Patel-Schneider, The Description Logic Handbook: Theory, Implementation, Applications, Cambridge University Press, 2003.
[3] Abraham Bernstein, Benjamin N. Grosof, Beyond monotonic inheritance: Towards semantic web process ontologies, Technical Report 190, Center for e-Business at MIT, 2003.
[4] F. Baader, B. Hollunder, Priorities on defaults with pre-requisites and their application in treating specificity in terminological default logic, Journal of Automatic Reasoning 15 (1995) 41–68.
[5] Piero Bonatti, Carsten Lutz, Frank Wolter, Description logics with circumscription, in: Proceedings of the KR Conference, 2006, pp. 400–410.
[6] Craig Boutilier, A semantical approach to stable inheritance reasoning, in: Proceedings of the IJCAI Conference, 1989, pp. 1134–1139.
[7] Gerhard Brewka, Adding priorities and specificity to default logic, in: Proceedings of JELIA Conference, 1994, pp. 247–260.
[8] James P. Delgrande, A preference-based approach to default reasoning: preliminary report, in: Proceedings of the AAAI Conference, 1994, pp. 902–908.
[9] F.M. Donini, D. Nardi, R. Rosati, Description logics of minimal knowledge and negation as failure, ACM Transactions on Computational Logic 3 (2002) 177–225.
[10] Patrick Doherty, A semantics for inheritance hierarchies with exceptions using a logic of preferential entailment, in: Proceedings of the SCAI Conference, 1989.
[11] J.P. Delgrande, T.H. Schaub, A general approach to specificity in default reasoning, in: Proceedings of the KR Conference, 1994, pp. 146–157.
[12] P.M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning and logic programming and N-person games, Artificial Intelligence 76 (1995) 321–357.
[13] Thomas Eiter, Thomas Lukasiewicz, Default reasoning from conditional knowledge bases: Complexity and tractable cases, Artificial Intelligence 124 (2000) 169–241.
[14] David W. Etherington, Raymond Reiter, On inheritance hierarchies with exceptions, in: Proceedings of the AAAI Conference, 1983, pp. 104–108.
[15] David W. Etherington, Reasoning with Incomplete Information, Pitman, 1988.
[16] Scott Fahlman, NETL: A System for Representing and Using Real-World Knowledge, MIT Press, 1979.
[17] Simon Ferndriger, Abraham Bernstein, Jin Song Dong, Yuzhang Feng, Yuan-Fang Li, Jane Hunter, Enhancing semantic web services with inheritance, in: Proceedings of the ISWC Conference, in: LNCS, vol. 5318, Springer, 2008, pp. 162–177.
[18] M. Gelfond, H. Przymusinska, Formalization of inheritance reasoning in autoepistemic logic, Fundamenta Informaticae (1990) 403–443.
[19] Moisés Goldszmidt, Judea Pearl, System-Z$^+$: A formalism for reasoning with variable-strength defaults, in: Proceedings of the AAAI Conference, 1991, pp. 399–404.
[20] Hector Geffner, Judea Pearl, Conditional entailment: Bridging two approaches to default reasoning, Artificial Intelligence 53 (1992) 209–244.
[21] Moisés Goldszmidt, Judea Pearl, Qualitative probabilities for default reasoning, belief revision, and causal modelling, Artificial Intelligence 84 (1996) 57–112.
[22] E. Grégoire, Skeptical theories of inheritance and nonmonotonic logics, in: Methodologies of Intelligent Systems, 1989, pp. 430–438.
[23] E. Grégoire, Skeptical inheritance can be more expressive, in: Proceedings of the ECAI Conference, 1990, pp. 326–332.
[24] Dov M. Gabbay, Karl Schlechta, Defeasible inheritance systems and reactive diagrams, Logic Journal of the IGPL 17 (2009) 1–54.
[25] P. Geerts, D. Vermeir, A nonmonotonic reasoning formalism using implicit specificity information, in: Logic Programming and Nonmonotonic Reasoning, 1993, pp. 380–396.
[26] B.A. Haugh, Tractable theories of multiple defeasible inheritance in ordinary nonmonotonic logics, in: Proceedings of the AAAI Conference, 1988, pp. 421–426.
[27] Shazzad Hosain, Hasan Jamil, Empowering owl with overriding inheritance, conflict resolution and non-monotonic reasoning, in: Proceedings of the AAAI Spring Symposium, 2009, pp. 53–58.
[28] John F. Horty, Defaults with priorities, Journal of Philosophical Logic 36 (2007) 367–413.
[29] John F. Horty, Richmond H. Thomason, David S. Touretzky, A skeptical theory of inheritance in nonmonotonic semantic networks, Artificial Intelligence 42 (1990) 311–348.
[30] S. Heymans, D. Vermeir, A defeasible ontology language, in: Proceedings of the DOA/CoopIS/ODBASE Confederated International Conferences, in: LNCS, vol. 2519, Springer, 2002, pp. 1033–1046.
[31] Thirunarayan Krishnaprasad, Michael Kifer, David Warren, An evidence-based framework for a theory of inheritance, in: Proceedings of the IJCAI Conference, 1989, pp. 1093–1098.
[32] Daniel Lehmann, Menachem Magidor, What does a conditional knowledge base entail? Artificial Intelligence 55 (1992) 1–60.
[33] Wolfgang May, Paul-Thomas Kandzia, Nonmonotonic inheritance in object-oriented deductive database languages, Technical Report 114, Institut für Informatik, Universität Freiburg, 1999.
[34] Leora Morgenstern, Inheritance comes of age: Applying nonmonotonic techniques to problems in industry, Artificial Intelligence 103 (1998) 237–271.
[35] David Makinson, Karl Schlechta, Floating conclusions and zombie paths: two deep difficulties in the 'directly skeptical' approach to defeasible inheritance nets, Artificial Intelligence 48 (1991) 99–209.
[36] Ekawit Nantajeewarawat, Vilas Wuwongse, Defeasible inheritance through specialization, Computational Intelligence 17 (2001) 62–86.
[37] J.L. Pollock, Defeasible reasoning, Cognitive Science 17 (1987) 481–518.
[38] J. Pollock, Cognitive Carpentry. A Blueprint for How to Build a Person, MIT Press, 1995.
[39] John L. Pollock, A recursive semantics for defeasible reasoning, in: Guillermo Simari, Iyad Rahwan (Eds.), Argumentation in Artificial Intelligence, Springer-Verlag, 2009, pp. 173–197.
[40] H. Prakken, G. Vreeswijk, Logics for defeasible argumentation, in: D. Gabbay, F. Günthner (Eds.), Handbook of Philosophical Logic, vol. 4, second ed., Kluwer Academic Publishers, 2002, pp. 219–238.
[41] L. Padgham, T. Zhang, A terminological logic with defaults: A definition and an application, in: Proceedings of the IJCAI Conference, 1993, pp. 662–668.
[42] Ray Reiter, G. Criscuolo, On interacting defaults, in: Proceedings of the IJCAI Conference, 1981, pp. 270–276.
[43] Ray Reiter, A logic for default reasoning, Artificial Intelligence 13 (1980) 81–132.
[44] R. Rosati, On the decidability and complexity of integrating ontologies and rules, Journal of Web Semantics 3 (2005) 61–73.

[45] Erik Sandewall, An approach to the frame problem, and its implementation, in: Machine Intelligence, vol. 7, Edinburgh University Press, 1972, pp. 195–204.
[46] Erik Sandewall, A functional approach to nonmonotonic logic, Computational Intelligence 1 (4) (1985). Also in the Proceedings of the IJCAI Conference, 1985.
[47] Erik Sandewall, Nonmonotonic inference rules for multiple inheritance with exceptions, Proceedings of the IEEE (October 1986).
[48] Karl Schlechta, Semantics for defeasible inheritance, in: Proceedings of the ECAI Conference, 1990, pp. 594–597.
[49] Karl Schlechta, Directly skeptical inheritance can not capture the intersection of extensions, Journal of Logic and Computation 3 (1993) 455–467.
[50] Geneviève Simonet, RS theory: a really skeptical theory of inheritance with exceptions, in: Proceedings of the ECAI Conference, 1990, pp. 615–626.
[51] Geneviève Simonet, On Sandewall's paper: Nonmonotonic inference rules for multiple inheritance with exceptions, Artificial Intelligence 86 (2) (1996) 359–374.
[52] G.R. Simary, R.P. Loui, A mathematical treatment of defeasible reasoning and its implementation, Artificial Intelligence 53 (1992) 125–257.
[53] L.A. Stein, Resolving ambiguity in non-monotonic inheritance hierarchies, Artificial Intelligence 55 (1992) 259–310.
[54] U. Straccia, Default inheritance reasoning in hybrid KL-ONE-style logics, in: Proceedings of the IJCAI Conference, 1993, pp. 676–681.
[55] Richmond H. Thomason, John F. Horty, Logics for inheritance theory, in: Michael Reinfrank, Johan de Kleer, Matthew L. Ginsberg, Erik Sandewall (Eds.), Non-Monotonic Reasoning and Reason Maintenance, Springer, 1988.
[56] Richmond H. Thomason, NETL and subsequent path-based inheritance theories, Computers and Mathematics with Applications 23 (1992) 179–204.
[57] David S. Touretzky, John F. Horty, Richmond H. Thomason, A clash of intuitions: The current state of nonmonotonic multiple inheritance systems, in: Proceedings of the IJCAI Conference, 1987, pp. 476–482.
[58] David S. Touretzky, Implicit ordering of defaults in inheritance systems, in: Proceedings of the AAAI Conference, 1984, pp. 322–325.
[59] David S. Touretzky, The Mathematics of Inheritance Systems, Pitman, 1986.
[60] David S. Touretzky, Richmond H. Thomason, John F. Horty, A skeptic's menagerie: conflictors, preemptors, reinstators and zombies in nonmonotonic inheritance, in: Proceedings of the IJCAI Conference, 1991, pp. 478–483.
[61] Xianchang Wang, Jia-Huai You, Li Yan Yuan, A default interpretation of defeasible network, in: Proceedings of the IJCAI Conference, 1997, pp. 156–161.