Dialectical Nonmonotonic Inheritance

Arthur M. Farley

CIS-TR-95-20
December 1995

Department of Computer and Information Science
University of Oregon

Dialectical
Nonmonotonic
Inheritance

Arthur M. Farley

Computer and Information Science
University of Oregon
Eugene, OR 97403 USA

ABSTRACT

We present a model of nonmonotonic inheritance based on a process of dialectical argumentation applied to mixed inheritance networks. A mixed inheritance network consists of strict and default links between nodes representing kinds or properties of objects. Given an object, represented by strict or default connections to nodes of an inheritance network, and a goal node in the inheritance network, the model argues with itself to determine inheritability of the goal with respect to that object. Burden of proof is a key parameter to this dialectical argumentation process, allowing our model to allocate risk in a flexible manner. We apply our model to several examples from the nonmonotonic inheritance literature, demonstrating its key properties.

Content Areas: automated reasoning, nonmonotonic reasoning

Word Count: 6324

Tracking Number: A061

Introduction

Inheritance reasoning systems have been studied in artificial intelligence as a means for representing and reasoning about object-related taxonomic and property knowledge. Early work on semantic networks (Quillian,1968; Sowa,1985) introduced a structural basis for this type of reasoning. Object-related concepts are represented as nodes in a hierarchical network of taxonomic, partonomic, and property links. Properties of an object or a subclass of objects that are true of (almost) all elements of a superclass need not be repeated at the sublevel, but instead are inherited as results of a search process in the network. Efficiency of representation and inference has motivated study of effective ways to determine inheritability of property and type knowledge.

The procedural semantics associated with semantic networks was criticized by Woods (1975) among others. This criticism resulted in a number of attempts to formalize the semantics of network-based notions in a more denotational fashion. One approach was to translate inheritance networks into equivalent sets of logical statements, thereby leaving behind the structural/procedural elements of semantic networks and grounding meaning in terms of more syntactic, logic-based semantics. Explorations with the logical approach resulted in the definition and application of nonmonotonic and default logics to capture the meaning of non-strict relations among taxonomic categories (Etherington, 1988; Reiter, 1980; Ginsberg, 1987).

The so-called direct approaches to inheritability have maintained inheritance networks and paths in such networks as basic theoretical elements. These approaches endeavor to define inheritability in terms of graph-theoretic properties of substructures in inheritance networks. With uncertainty introduced into the networks through the presence of possibly conflicting, non-strict (or default) links, evaluating the appropriateness of inheritability results often has proven to be an intuitive matter. A set of examples that illustrates many of the underlying issues has been presented in a recent review article (Horty, 1994); a selection of these examples will be used to illustrate and evaluate our dialectical approach.

In this paper, we introduce a new perspective on inheritability semantics, that of dialectical argumentation. Two sides, alternately arguing for and against the acceptability of a proposed inheritability relation, battle it out until one side must concede. While seeing inheritance reasoning as a process of dialectical argumentation, we also return to a procedural, direct approach, attempting to ground our semantics in terms of interactions between paths of an inheritance network. Importantly, conflicting indications, which pose a problem for logic-based systems, are just the situations that a process of dialectical argumentation is meant to address and resolve.

Inheritance Networks

We can view an argument from two perspectives: as knowledge structure and as reasoning process. Our model of argument structure is specified in a form derived from that presented by Horty (Horty, 1994). Nodes in an inheritance network, denoted by small letters, will refer to kinds of objects; we will use a small letter from the end of the alphabet (e.g., x, y, z) to represent an arbitrary node of a network. Kinds of objects correspond to sets of objects having a common designation (e.g., animal) or certain properties or parts in common (e.g., those with four legs).

Nodes of an inheritance network are interconnected by directed links, each link connecting two nodes of the network. There are four types of links, corresponding to possible combinations of strength {strict, default} and sign {positive, negative}. The link types ==> and <=/=> denote strict positive and strict negative links, respectively. These are equivalent to universally quantified conditional statements in first-order predicate logic. The link x==>y represents that every individual object of kind x is of kind y (i.e., "All x's are y's"). The link x=/=>y represents the statement "No x's are y's", which implies the reverse link, as well.

Link types --> and -/-> denote default positive and default negative links, respectively. The meaning of default links is not expressible in terms of semantics of standard logic. They capture various notions of imperfect generalizations often made in commonsense, object-related reasoning. The link x-->y represents ideas expressed by the following sentences: "Most x's are y's"; "Typical x's are y's"; or "By default, x's can be considered to be y's". These links represent a strong, but indefinite, inheritance relationship between the two kinds of objects. An example, would be b-->f, to capture the notion that "birds normally (typically, most often) fly".

Objects are represented by special nodes labeled by capital letters; we will use capital letters from the end of the alphabet to represent an arbitrary object node. An object is connected to an inheritance network by the same types of links as above, representing what we know directly about the object. When connecting an object X to a node y by a strict link, i.e., X==>y (X=/=>y), we are representing that X definitely is (is not) of kind y. If we connect X to y by a default link, i.e., X-->y (X-/->y), we are representing that X most likely is (most likely is not) of kind y. This leaves open the possibility that the object X may not have the represented relationship to the kind y.

Default links have been called "defeasible" links (Pollock, 1987) as they can be defeated, or preempted, by more specific or conclusive information. We use the term "default" for these links to capture the strong connection that is intended. We reserve the term "defeasible" for the more general, somewhat weaker connection that exists between nodes interconnected by paths involving several default links in an inheritance network, i.e., the default relation is not transitive. We will discuss the construction of allowable, defeasible arguments in the next section.

We want to use an inheritance network as basis for answering questions regarding characteristics of given objects. What is directly known about an object is represented by a set of strict and default links connecting it to nodes of an inheritance network. A particular type or property of interest, i.e., a *goal*, is simply a node in the network. We want to decide whether to accept a proposed inheritance relation between a given object and goal. This will be determined by constructing arguments for and against the proposal and comparing their characteristics.

We provide an example of an inheritance network with an object connected in Figure 1. This network can be seen to represent the relationships that most birds (node b) fly (node f), all penguins (node p) are birds, and all penguins do not fly. We also represent an object (node A), which we are sure is a bird and is most likely a penguin. We use heavy lines to represent strict links and thin lines to represent default links in all figures to follow. We normally will place the goal node at the top of the networks in our figures.

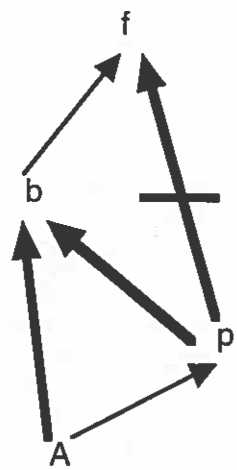

Figure 1

Arguments

As stated above, we can see an argument from two perspectives. First, an argument is a structured knowledge entity, such as when we say "give me a good argument". From this perspective, an argument will be represented as a directed path in an inheritance network, leading from an object node to a goal node. Such an argument provides a basis for believing that the given object belongs to the goal kind.

There may be arguments for and against believing a goal concept of an object based upon a given inheritance network. How the various arguments are generated and compared is our second perspective on argument, that of argument as dialectical process. This is the perspective adopted when one says "we had quite an argument". We develop a dialectical approach to argumentation in this paper, according to which arguments for and against a proposed inheritability relationship are generated alternately by opposing sides. Each side responds to the other's argument moves, until one side must concede the argument and a decision can be taken. Deciding which side goes first and when a side must concede the argument depends upon a given burden of proof, a key aspect of our model of dialectical argumentation.

We first define the structure of allowable arguments in an inheritance network, discussing their properties and interactions, and then outline a proposed process of dialectical argumentation, defining burden of proof and illustrating its role in deciding inheritability.

Argument Structure

Given an inheritance network, we are interested in defining the notion of arguments for and against a certain conclusion. An *inheritance argument* is a directed path in an inheritance network. An argument P from a *start* node x to a *finish* node y through an intermediate, possibly empty, sequence of nodes π, denoted as P(x, π, y), is such that if node v immediately precedes node u in the argument then v is directly connected to u by a link of the inheritance network. However, not all such paths constitute allowable arguments. Which paths do form allowable arguments will be defined by argument construction rules given below.

We characterize an argument in terms of its strength {strict, default, defeasible} and sign {positive, negative}. Note that we introduce a *defeasible* strength for an argument between nodes interconnected by compound paths of a particular form; defeasible arguments capture an inheritance relationship that is weaker than that established by either strict (all) or default (most) links. If P(x, π, y) is a defeasible positive argument, it represents the idea that "it is reasonable to believe that x's are y's". If this path begins at an object node X, we are saying that "X could reasonably be considered to be a y". Distinguishing between default and defeasible argument strengths will allow us to capture, in a straightforward manner, a number of inheritability results that have been judged to be intuitively correct. We will demonstrate these through examples later in the paper.

Allowable arguments in an inheritance network, with their associated strengths and signs, are defined recursively, in the "backward" direction from a given goal node. Links in the network form direct arguments, as defined by the following argument construction rule:

Rule R1: (direct arguments)

A. given *x* ==>y, P(*x* , ø, y) is a strict positive path
B. given *x* =/=>y, P(*x* , ø, y) is a strict negative path
C. given *x* -->y, P(*x* , ø, y) is a default positive path
D. given *x* -/->y, P(*x* , ø, y) is a default negative path
E. given x<==y, P(x, ø, y) is a candidate negative path

In the above rule, *x* refers to an arbitrary kind x or object X; the symbol "ø" represents the empty sequence of nodes. Rule R1(E) captures the potential for use of modus tollens reasoning over a strict (logically sufficient) positive link in an inheritance network. If a negative link is added as "head" of such a path when it is extended, the path will become a negative path of appropriate strength. A candidate negative path on its own (i.e., not completed by a forward-directed, negative link) does not constitute an allowable argument.

We can extend an argument path by adding an element as new "head" of the path, creating a compound argument, as defined by the following construction rule:

Rule R2: (compound arguments)

A. P(x,π,y) is a strict positive path not containing *z*, then
   (i) given *z*==>x, P'(*z*,x,π,y) is a strict positive path
   (ii) given *z*-->x, P'(*z*,x,π,y) is a default positive path
B. P(x,π,y) is a strict negative path not containing *z*, then
   (i) given *z*==>x, P'(*z*,x,π,y) is a strict negative path
   (ii) given *z*-->x, P'(*z*,x,π,y) is a default negative path

C. P(x,π,y) is a candidate negative path not containing z (z), then
- (i) given z<==x, P'(z,x,π,y) is a candidate negative path
- (ii) given z =/=> x, P'(z,x,π,y) is a strict negative path
- (iii) given z -/-> x, P'(z,x,π,y) is a default negative path

D. P(x,π,y) is a default positive path not containing Z (z), then
- (i) given z==>x, P'(z,x,π,y) is a defeasible positive path
- (ii) given z-->x, P'(z,x,π,y) is a defeasible positive path
- (iii) given Z==>x, P'(Z,x,π,y) is a default positive path.

E. P(x,π,y) is a default negative path not containing Z (z), then
- (i) given z==>x, P'(z,x,π,y) is a defeasible negative path
- (ii) given z-->y, P'(z,x,π,y) is a defeasible negative path
- (iii) given Z==>x, P'(Z,x,π,y) is a default negative path

F. P(x,π,y) is a defeasible positive path not containing z, then
- (i) given z==>x, P'(z,x,π,y) is a defeasible positive path
- (ii) given z-->x, P'(z,x,π,y) is a defeasible positive path

G. P(x,π,y) is a defeasible negative path not containing z, then
- (i) given z==>x, P'(z,x,π,y) is a defeasible negative path
- (ii) given z-->x, P'(z,x,π,y) is a defeasible negative path

In rule R2, z represents either an arbitrary kind z or object Z. Rules R2(A)(i) and R2(B)(i) capture the standard chaining rules of implication in first-order logic. Rules R2(A)(ii) and R2(B)(ii) indicate that a default link followed by a strict path is a default path. Rule R2(C) represents ways in which a candidate negative path can be either continued or transformed into a strict or default negative path. Note that a negative path either ends with a single negative link or contains a single negative link followed by a candidate negative path. The two final rules, R2(F) and R2(G), indicate allowable extensions of defeasible paths.

Rules R2(D) and R2(E) represent that either a strict or a default extension of a default path between kinds becomes a defeasible argument. This captures the observation that even if most or typical x's are (or are not) y's and all or typical z's are x's, the z's that are x's may not tend to be the ones that typically are (or are not) y's, although it is reasonable to think they may be. In fact, the actual relationship between z and y may be anything from strict to non-existent. Membership in the "parent" kind is conditioned by the child kind and, thus, is not uniformly spread across the parent set; thus, the strength of the inheritance argument between z and y is weakened to defeasible, reflecting this increased uncertainty. The rules indicate that when an object is connected to a network by a strict link, we do assume a uniform distribution over the "parent" kind and adopt the strength of the existing argument for the result.

These two rules constitute a critical element in our theory of inheritance argument structure, distinguishing it from approaches that fail to note the difference between default and defeasible strengths arguments. This distinction will allow us to handle certain argument comparisons in a more straightforward manner.

Given an inheritance network I, an object X, and a goal node y, an argument constructed by the rules above is *grounded* iff it starts with X and ends with y, based on the links of I and the connections of X to I. For example, given the inheritance network in Figure 1, we find grounded arguments with respect to object A and goal f, as follows:

A-/->f:: P_1(A, p, f); A-->f:: P_2(A, p, b, f), P_3(A, b, f).

There are both positive and negative arguments regarding the conclusion that object A is has property f. Thus, arguments can stand in a conflicting, or contradictory, relationship to each other. Conflict relationships between arguments have been defined by a number of previous works (Horty, 1994; Sartor, 1993; Pollock, 1987; Loui, 1987). Two arguments P(x, π1, y) and P'(x, π2, y) *directly conflict* if they differ in sign. More generally, two arguments *conflict* if one directly conflicts with a subargument of the other, i.e., one argument is of the form (π_1, x, π_2, y, π_3), the other is the form (x, π, y), and (x, π, y) and (x, π_2, y) are of opposite sign. In our example, we find the following (directly) conflicting arguments: (P_1, P_2) and (P_1, P_3).

Not all conflicts between arguments are acceptable. Two arguments from a kind x to a kind y are *inconsistent* if the arguments directly conflict and both are either strict or default in strength. This reflects the observation that if we find an argument concluding that all or typical x's are y's and another concluding that all or typical x's are not y's, there is an inconsistency in our knowledge. Two arguments from an object X to a kind y are *inconsistent* if the arguments directly conflict and both are strict in strength. X can not definitely be and not be of a kind y. However, a situation where an argument supports an object X typically being a y and X typically not being a y is not necessarily inconsistent; since X is just a single object, we may find strong arguments for both sides without implying inconsistent overlap of positive and negative kinds.

Certain pairs of arguments may stand in a stronger contrary relationship to one another than simply conflicting. In this case, one argument will be said to defeat the other. Defeat between conflicting arguments will be determined by comparative strengths. As such, we state that a strict argument is stronger than a default argument, which in turn is stronger than a defeasible argument.

One argument A *defeats* another argument B iff argument B is of the form (π_1, x, π_2, y, π_3), A is of the form (x, π, y), argument A and the subargument (x, π_2, y) are of opposite sign, and A is of greater strength than the subargument (x, π_2, y). In other words, an argument that conflicts with another argument and is of greater strength than the corresponding, directly conflicting subargument defeats the other argument. Above, argument P_1 defeats argument P_2. Our definition of argument defeat subsumes various definitions of defeat in mixed inheritance networks that have been proposed previously (Horty, 1994). By distinguishing between default and defeasible arguments, we simplify the analysis of argument defeat in inheritance reasoning.

Burden of Proof

One must decide how conservative to be when choosing whether to accept an inheritance relation. One way to address this concern would be to specify the strength of supporting argument required. Another means of allocating risk is to decide which type of error, either of commission (i.e., false positives) or omission (i.e., false negatives), one is more willing to accept. One would like to adjust this allocation depending upon estimated consequences of wrongfully accepting or not accepting an inheritance relation between a particular kind and given object.

In an attempt to address these concerns, research on inheritance reasoning has produced several variants of what have been termed credulous or skeptical reasoning systems (Horty and Thomason, 1988; Touretzky, et.al., 1987). There has been considerable discussion and some disagreement regarding appropriate definitions of such reasoners, due to semantic anomalies that have arisen under differing proposals. Our goal is not to discuss the issues surrounding these proposals; this has been done elsewhere (Horty, 1994). Rather, we present an alternative approach based on the comparison of arguments for and against an inheritance claim and the ability to allocate risk in a flexible manner through specifying burden of proof.

An *inheritance claim* is a statement as to the inheritability of a kind by an object. A positive claim will be denoted as X~~>y, while X~/~>y will denote the complementary, negative claim. Each grounded argument P(X, π, y) supports one of these two claims. Whether a claim is acceptable will depend upon whether the claim can win a dialectical argument under a given burden of proof. A burden of proof is specified as a parameter to an argument process, not as a property of the reasoning system. Specifying burden of proof as a parameter provides a degree of flexibility missing in skeptical or credible reasoners that have chosen one approach permanently.

One domain in which the notion of burden of proof has been defined and applied formally is that of legal argumentation. A different burden of proof may be mandated at each stage of a legal process or for different types of legal action. For example, arguments sufficient to indict someone need not be as convincing as those needed to convict someone of a criminal offense. When considering conviction in criminal cases, we tend to be more concerned with errors of commission (i.e., finding someone guilty when they are not) and, thus, place a high burden of proof on the side arguing for guilt.

There are two aspects to specification of burden of proof: (i) which side of a claim (positive or negative) bears the burden and (ii) what level of proof is required for acceptance of the statement. The first aspect addresses whether one is more concerned about accepting false positives, where the burden is placed on the positive claim, or false negatives, where the burden is placed on the negative claim. Sometimes, one may want a good argument supporting a particular, inheritance claim before accepting it. On other occasions, where accepting a claim may have value and little risk, one may demand a good argument against the claim before denying its acceptance.

The second aspect of burden of proof, that of "proof level", is concerned with what is considered to be a good argument. Strength of required argument is only one aspect of the notion of proof level we will adopt. Proof level also will depend upon notions of defendable and justifiable argument. A *defendable argument* is a grounded argument that cannot be defeated with the given inheritance network and object representation. Such an argument has been termed *plausible* (Sartor, 1993) in a more general argument setting. A *justifiable argument* is a defendable argument with the additional condition that every argument that conflicts with it can be defeated.

We define the following proof levels:

• *scintilla of evidence (se)*: there exists a defendable argument supporting the claim;

• *preponderance of evidence (pe)*: there exist stronger arguments supporting the claim than its negation;

• *dialectical validity (dv)*: there exists a justifiable argument supporting the claim.

Winning a scintilla of evidence argument merely requires that some argument be defended against attack. In a dialectical argumentation context, this means that only defeating arguments can be considered by the opposing side; a conflicting argument of lesser or equal strength is irrelevant. Preponderance of evidence requires a means for assessing relative strengths of sets of conflicting, non-defeating arguments. Assuming a consistent inheritance network I and object description X, we are concerned about sets of conflicting, defeasible or default arguments about X. Having a greater number of independent, defeasible or default arguments for a claim represents a stronger case; when default arguments are included, they hold priority in the scoring. *Independent arguments* are those that share only start and finish nodes. Under preponderance of evidence, if the opposing side finds an equally strong argument, this can not be ignored; the argument must be counteracted, either by defeating it or by finding another argument of equal strength to rebut it. Finally, dialectical validity requires not only that one find a grounded argument that can be defended from attack but also that any argument put forth by the other side be defeated, even if the argument is weaker in strength; this reduces risk of error to the greatest extent possible.

Burden of proof plays several roles in the process of argumentation: (i) as basis for deciding the relevance of argument paths found by the active side; (ii) as basis for deciding the sufficiency of the outcome of an active side's move (i.e., whether a check condition has been realized); (iii) as basis for determining that an argument is over; and (iv) as basis for determining the winner of an argument.

Argument Process

Now that we have defined a structure for arguments as acyclic paths from object to goal nodes in an inheritance network and defined a number of important properties of and relationships between arguments, we can discuss how we can come to decide, in a particular situation, whether to ascribe property or kind y to object X. The decision will be made through a process of dialectical argumentation according to a given burden of proof. The model presented here is based upon a general model of argumentation described earlier by Freeman (Freeman, 1994; Farley and Freeman, 1995), adapted to the restricted circumstances of inheritance reasoning as defined above.

Given an object X, represented as connections to an inheritance network I, the question of interest is what claims are acceptable for a given burden of proof. The set of accepted claims constitutes the *inheritability relation* IR(X, I, B) for object X relative to inheritance network I under burden of proof B. To compute this inheritability relation, we consider each node of network I as goal and determine the acceptability of corresponding positive (X~~>y) and negative (X~/~>y) claims for the given object X through a series of dialectical argumentation processes.

A dialectical argument has two sides, where *Side-1* argues in favor of an input claim and *Side-2* against the claim, i.e., in support of its negation. The argument process begins with Side-

1 attempting to find a grounded argument for the input claim in the given inheritance network with the given object description. Search for a supporting argument proceeds from the goal node toward the object node, in a backward-directed manner, according to the construction rules above. If no grounded support can be found, the argument ends with a loss for Side-1, and the input claim is not accepted. All burdens of proof require Side-1 to construct at least one grounded argument to begin the argumentation process.

After this, the two sides alternate as *active side* of the argument. A side remains active until it succeeds in creating a check condition or runs out of possible moves and must concede the argument. A *check condition* for side S of an argument is a situation such that, if the other side can not successfully refute its relevant arguments, side S will win the argument. It is a controlling situation, requiring an adequate response from the other side. Except for the initial situation, when Side-1 must generate a grounded argument for the claim, the active side is faced with a set of *check arguments* for the other side, i.e., arguments responsible for the other side holding the check condition. The active side must refute at least one of the other side's check arguments.

When active, a side selects which argument moves to apply from a set of possible moves. To do this, a side can apply one of two primitive functions that search the inheritance network I for relevant arguments. The first is *find-arguments (X, g, s, I)*, which searches for argument paths from node X to node g of sign s in inheritance network I. The function returns a list of argument paths in decreasing order of strength, or it returns an empty list indicating no such paths exist. The other function, *find-refuting-arguments(A, I)*, finds refutations to argument A. It is equivalent to the union of all conflicting arguments for each pair of relevant nodes in argument A. Only pairs of nodes that isolate subarguments of defeasible or default strength need be considered. This function can be implemented by calls to the function *find-arguments* with parameters being relevant pairs of node, s and f, from argument A and the complement of the sign of the subargument between s and f in argument A. Only arguments of strength greater than defeasible are considered as possible, defeating arguments; counterarguments of equal strength can serve only as *rebuttals*.

Whether an argument is adequate to generate a check condition for the active side depends upon the burden of proof specified. For example, under a burden of proof of dialectical validity, Side-2 can consider both defeating and rebutting arguments in response to Side-1's arguments. If Side-2 finds a refuting argument, Side-1 must defeat Side-2's response, propose a completely new argument for its claim, or concede the argumentation process. Side-2 can continue throwing up counterarguments to a given argument, all of which Side-1 must defeat if it is to prevail under a burden of dialectical validity. If the burden of proof is preponderance of evidence, then Side-2 must generate a counterargument of strength equal to that proposed by Side-1. If it can do this, Side-1 finding another grounded argument in favor of the claim is sufficient to regain a check condition. As long as Side-1 can come up with more arguments of strength greater than or equal to those that Side-2 produces, it will prevail.

If the burden of proof is merely scintilla of evidence, Side-2 can only consider defeating arguments in response to Side-1's arguments. Side-1 need not defeat rebuttals to win the argument under this burden of proof; it must merely defend some argument from defeat; if Side-2 can defeat an argument, Side-1 can abandon it in favor of another supporting the input claim. Note we can characterize burden of proof in terms of where it places the "burden of defeat", i.e., which side must defeat the other's arguments. In the case of scintilla of evidence, the burden of defeat is on Side-2; while under a burden of proof of dialectical validity, the burden of defeat is on Side-1. Under preponderance of evidence, neither side takes on the burden of defeat. This is a free for all, where piling up more arguments of greater or equal strength in one's favor is sufficient.

We will not further formalize the dialectical argumentation process. The discussion should make it clear how each side can use the primitive operations, in response to the other side's check arguments, to attempt to find an adequate refutation and establish a check condition. We now consider a number of examples demonstrating general principles of our approach and illustrate the impacts that burden of proof has upon the argumentation process and its outcome.

Examples

We consider a series of examples illustrating our approach on several, paradigmatic networks. Consider the penguin example of Figure 1 with arguments: A~/~>f:: P_1(A, p, f); A~~>f:: P_2(A, p, b, f), P_3(A, b, f). We see that for the positive claim A~~>f, if Side-1 puts forth argument P_2, Side-2 can defeat it. Under scintilla of evidence, Side-1 can respond with P_3, which can not be defeated. In this case, both sides can win scintilla of evidence arguments, and only scintilla of evidence arguments; while the positive claim can produce two arguments, they are not independent, sharing intermediate node b. If we weaken the connection from p to f to be only a default link, we see quite a different situation. Now the positive claim has two defendable arguments, including one of default strength, i.e., P_3(A, b, f); the negative claim can only muster a single, defeasible argument P_1(A, p, f) that is defeated by the default argument. Hence, positive claim A~~>f can win arguments up through dialectical validity. On the other hand, if we change Figure 1 to have a strict link between A and p, as is often posed, we see that we can generate a strict argument for the negative claim, and only default and defeasible arguments in support of the positive claim. In this case, the negative claim A~/~>f can win arguments up through dialectical validity.

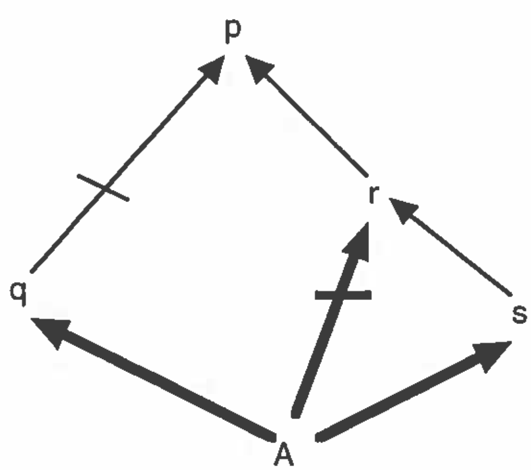

Figure 2.

Consider the slightly more complex situation given in the network of Figure 2. Here the only argument that the positive claim A~~>p can generate, i..e., P(A, s, r, p), is defeated by Side-2, using the strict, negative link between A and r. As such, the negative claim, having one defendable, argument, i.e, P'(A, q, p) can win arguments up through dialectical validity. If we weaken the negative link between A and r to default strength, then the positive argument is no longer defeated, only rebutted. In this case, either side of the claim can win scintilla of evidence arguments only. If we also weaken the strength of the link between A and s to default, again link A-/->r will defeat the only argument for the positive claim, creating a situation as the original. The same results are obtained if we also weaken the link between A and q to default strength.

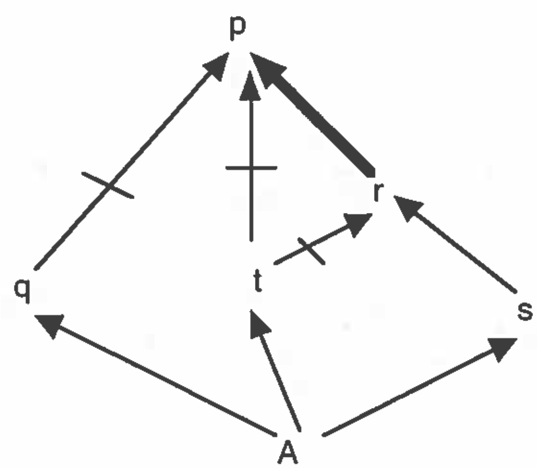

Figure 3

We make the situation a bit more complex in Figure 3. Under scintilla of evidence, both positive and negative claims win; each has a defendable argument, i.e., P(A, s, r, p) for A~~>p and P'(A, q, p) or P''(A, t, p) for A~/~>p. When arguing for the negative claim, Side-1 can find two independent arguments; thus, under preponderance of evidence, the negative claim remains acceptable. Since neither side can defeat the other's arguments, these are the only inheritability results. If we make the link between A and q or between A and t strict, then the negative claim can win arguments up through dialectical validity, having a default argument that defeats the defeasible argument in support of the positive claim. Similarly, if the link between A and s were made strict in the original network, the positive argument could win up through dialectical validity.

In Figure 3, if we make node r the goal, we have one positive, defeasible argument for object X inheriting kind r, but three defeasible arguments for the negative side of the claim. This is due to the contrapositive use of the strict link between r and p. Two of the negative arguments are independent -- P(A, t, r) and P'(A, q, p, r); thus, the negative claim A~/~>r can win arguments up through preponderance of evidence. The positive claim can only win under scintilla of evidence.

Finally, we present an example that captures an important difference between the semantics of our approach and that of some previous approaches. In Figure 4 below, we see that both positive and negative claims can marshal defendable, defeasible arguments in their favor, i.e., P(A, q, s) for A~~>s and P'(A, p, s) for A~/~>s. Thus, both sides win under a scintilla of evidence. Note that the positive argument P''(A, p, q, s) is defeated by the negative default link between p and s. This second positive argument, which includes p as an intermediary between A and q, could be considered a more complete explanation for A~~>s. Thus, one could argue that its defeat should imply defeat of the other positive argument, as well. From this perspective, adding p to the argument is giving further information as to why A is likely to be of kind (or have property) q.

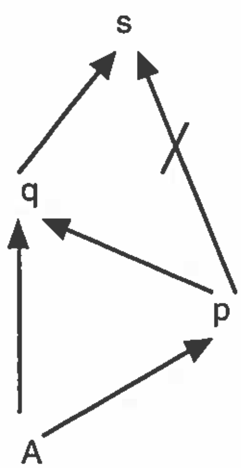

Figure 4.

Under our model, the result is just the opposite; the default link directly from A to q is what adds information to the network. It indicates that the A's that are p's also tend, for the most part, to be the p's that are q's; this is something not directly implied by the network without the default link from A to q. From this perspective, defeat of the less informed argument involving p does not impact the stronger argument bypassing p.

The decision not to include the weakening of the inheritance relation over multiple default links and to view default strength arguments as transitive has caused many of the reported problems in determining intuitively acceptable meanings for inheritance networks. In particular, it has led to counterexamples based on particular node labelings that claim to demonstrate network semantics are incorrect. What is missing from these purported counterexamples are the direct, default links between nodes for other, implicitly known, but unrepresented knowledge. Inclusion of these additional links would change the strengths of relevant arguments and, thereby, lead to the different patterns of inheritability that proposers of the counterexamples note.

Given our model of argument strength, the notion of network stability (Horty, 1994), whereby adding default links corresponding to defeasible inheritance results does not change network semantics, is misinformed. Only if a defeasible link were added, i.e., weaker than a default link, could we agree with the notion of stability. Otherwise, adding direct default links between nodes currently connected only by defeasible arguments is an addition of specific information that should, and in our model does, change network semantics.

Conclusion

In this paper, we have proposed a rethinking of inheritance reasoning systems and their semantics based directly on mixed inheritance networks. The key elements of our approach that distinguish it from earlier proposals are (i) treating inheritance reasoning as a form of dialectical argumentation; (ii) introducing burden of proof as a means for allocating risk and determining acceptability of inheritability claims; (iii) eliminating transitivity of the default relation, thereby weakening the strength of default arguments to defeasible when extended.

We believe our approach subsumes many of the positive properties of the network-based methods that have been proposed previously and clarifies or eliminates other issues that have caused this area of inquiry difficulties in the past. In particular, discussions concerning the nature of credulous and skeptical systems, e.g., which is the most appropriate and how to define them properly, are replaced by a definition of burden of proof in terms of defendable or justifiable arguments and relative strengths among conflicting arguments. The ability to set the burden of proof as a parameter reflecting risk acceptability in a particular situation, rather than attributing a fixed property (either skeptical or credulous) to the reasoning system, provides needed flexibility in making inheritance decisions.

References

Etherington, D. *Reasoning with Incomplete Knowledge*, Morgan Kaufmann: Menlo Park, 1988.

Farley, A.M. and Freeman, K. "Burden of proof in legal argumentation", in *Proceedings of Fifth International Conference on Artificial Intelligence and Law*, Washington, DC, 1995.

Freeman, K. *A Computational Model of Dialectical Argumentation*, Ph.D. Thesis, Computer and Information Science Department, University of Oregon, 1993.

Ginsberg, M. (ed.), *Readings in Nonmonotonic Reasoning*, Morgan-Kaufmann: San Mateo, CA, 1987.

Horty, J.F. "Some direct theories of nonmonotonic inheritance", in Gabbay, D.M., Hogger, C.J, and Robinson, J.A., *Handbook of Logic in Artificial Intelligence and Logic Programming*, Oxford Press : New York, 1994.

Horty, J.F. and Thomason, R. "Mixing strict and defeasible inheritance", in *Proceedings AAAI-88*, 427-432, 1988.

Loui, R. "Defeat among arguments: a system of defeasible inference", *Computational Intelligence*, 3, 100-106, 1987.

Pollock, J. "Defeasible reasoning", *Cognitive Science*, 11, 481-518, 1987.

Quillian, J.R. "Semantic memory", in M. Minsky(ed.),*Semantic Information Processing*, 227-270, 1968.

Reiter, R. "A logic for default reasoning", Artificial Intelligence, 13, p81-132, 1980.

Sandewall, E. "Nonmonotonic inference rules for multiple inheritance with exceptions", *Proceedings of the IEEE*, 74: 1345-1353, 1986.

Sartor, G. "A simple model for nonmonotonic and adversarial legal reasoning", in *Proceedings of Fourth International Conference on Artificial Intelligence and Law*, 192-201, 1993.

Sowa, J.F. *Conceptual Structures*, Addision-Wesley: Reading, MA, 1984.

Sowa, J.F. (ed.) *Principles of Semantic Networks*, Morgan-Kaufman: Menlo Park, CA, 1992.

Touretzky, D., Horty, J.F., and Thomason, R. "A clash of intuitions: the current state of nonmonotonic multiple inheritance systems", in *Proceedings of IJCAI-87*, 476-482, 1987.

Woods, W. "What's in a link: foundations of semantic networks", in Bobrow, D. and Collins, A. (eds.) *Representation and Understanding: Studies in Cognitive Science*, p 35-82, Academic Press: New York, 1975.