

A Theory of Nonmonotonic Inheritance Based on Annotated Logic

Krishnaprasad Thirunarayan *
Department of Computer Science and Engineering
Wright State University
Dayton, Ohio 45435

Michael Kifer†
Department of Computer Science
SUNY at Stony Brook
Stony Brook, NY 11794

September 2, 1992

Abstract

We propose a logical language for representing networks with nonmonotonic multiple inheritance. The language is based on a variant of annotated logic studied in [5, 6, 17, 18, 19, 20, 21]. The use of annotated logic provides a rich setting that allows to disambiguate networks whose topology does not provide enough information to decide how properties are to be inherited. The proposed formalism handles inheritance via strict as well as defeasible links. We provide a formal account of the language, describe its semantics, and show how a unique intended model can be associated with every inheritance specification written in the language. Finally, we present an algorithm that correctly propagates inherited properties according to the given semantics. The algorithm is also complete in the sense that it computes the set of all properties that must be inherited by any given individual object, and then terminates.

*Work supported in part by NSF grant IRI-9009587.

†Work supported in part by NSF grant IRI-8903507.

1 Preliminaries

It is customary to depict inheritance networks using diagrams where *properties* and *individual objects* are represented as nodes of the diagram. Object and property nodes are depicted as labeled rectangles and ovals, respectively. Each property node defines a *class* of individual objects that possess this property. Each link in the diagram represents one of the following four types of relationship between the nodes:

- A solid link going from an individual object o to a property node p represents an *instance-of* relationship; it says that o is a member of the class defined by p .
- A slashed link from an individual node o to a property node p represents a *non-instance* relationship; it says that o is *not* a member of the class defined by p .
- A solid link from p_1 to p_2 , where p_1 and p_2 are property nodes, denotes an *is-a* relationship; roughly, it says that the class defined by p_1 is a subset of the class of objects defined by p_2 .
- A slashed link from p_1 to p_2 denotes an *is-not-a* relationship between these property nodes; it is a statement to the effect that elements of the class defined by p_1 “usually” do not have property p_2 .

The is-a and is-not-a relationships come in two flavors: *strict* and *defeasible*; “instance-of” and “non-instance” relationships are always strict. Strict “instance-of”, “non-instance”, and “is-a” links have simple mathematical meaning based on the usual set-theoretic relations \in , \notin , and \subseteq . The strict “is-not-a” link from p_1 to p_2 says that the classes defined by p_1 and p_2 are disjoint. We will depict strict “is-a” and “is-not-a” links using bold solid and bold slashed arrows, respectively. For “instance-of” and “non-instance” links we will use plain (solid or slashed) arrows, even though these links are always strict.

The defeasible flavor of the above links is not that well understood, and there are several different views of what they should mean. The common intuition is that such a link between p_1 and p_2 expresses the fact that under normal circumstances the members of the class of p_1 are or are not members of the class of p_2 , depending on whether the relationship is “is-a” or “is-not-a”. However, certain “atypical” members of p_1 may be “excused” from being members of the class of p_2 . Defeasible links will be depicted using plain (solid or slashed) arrows.

Consider the “Penguin Triangle” example shown in Figure 1. Intuitively, the individual object Donald there should inherit the property *fly* by virtue of being a bird. However, for Tweety things are not that clear-cut. On one hand, it should fly because it is a bird. On the other hand, as a penguin, Tweety might as well inherit $\neg fly$. Thus, on the surface, the network appears to be ambiguous about whether Tweety is a flying bird.

Several works tried to address this and related problems from various angles. The main advantage of our formalism is that it is based on a well-studied logic, has model theory and a sound and complete algorithm for computing inheritance. We also argue that our

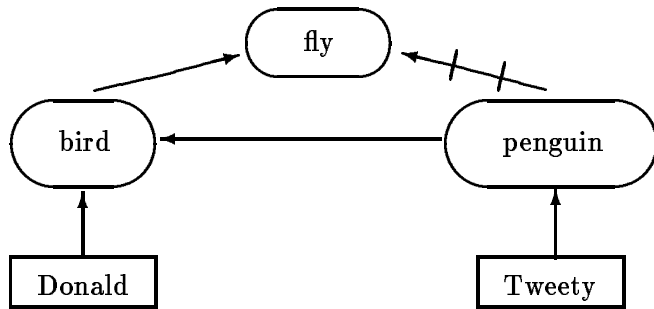


Figure 1: The penguin triangle

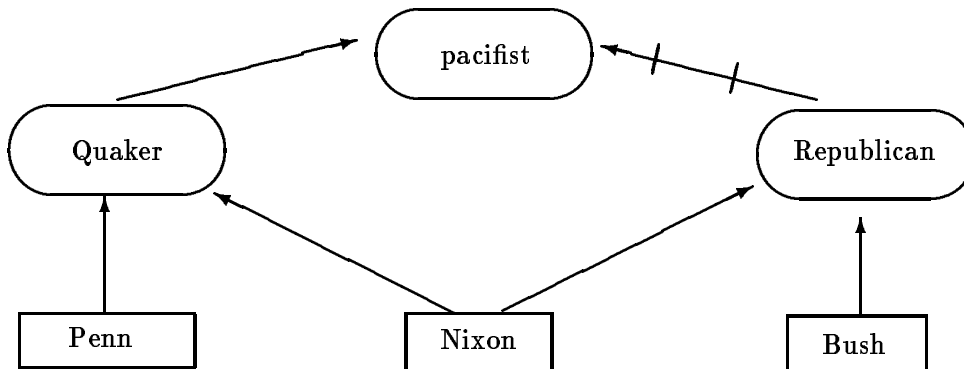


Figure 2: The Nixon diamond

language is more expressive and allows to specify inheritance network in more precise terms, disambiguating many otherwise ambiguous diagrams.

In a nutshell, we would approach the problem of Figure 1 as follows. We can say that $fly(Tweety)$ has a positive evidence “of strength” **bird** and also a negative evidence of strength **penguin**. Furthermore, from the network topology we infer that knowing that Tweety is a penguin is *more informative* than knowing that it is a bird. So we resolve the conflict by overriding the **bird**-strong inheritance of $fly(Tweety)$ by the **penguin**-strong inheritance of $\neg fly(Tweety)$, concluding that Tweety does not fly.

For another example, consider the, so called, Nixon Diamond of Figure 2. Here, Penn is a pacifist by virtue of being a Quaker, while Bush is not a pacifist because he is a Republican. However, Nixon presents a problem: as a Quaker, he should be a pacifist but, as a Republican, he should not. As before, there are conflicting conclusions regarding Nixon’s pacifism, but this time they are supported by incomparable pieces of evidence, since neither *Quaker* nor *Republican* are subclasses of one another. In this situation, we would choose to hold an “inconsistent belief” about Nixon pacifism, i.e., to become *aware* of the inconsistency

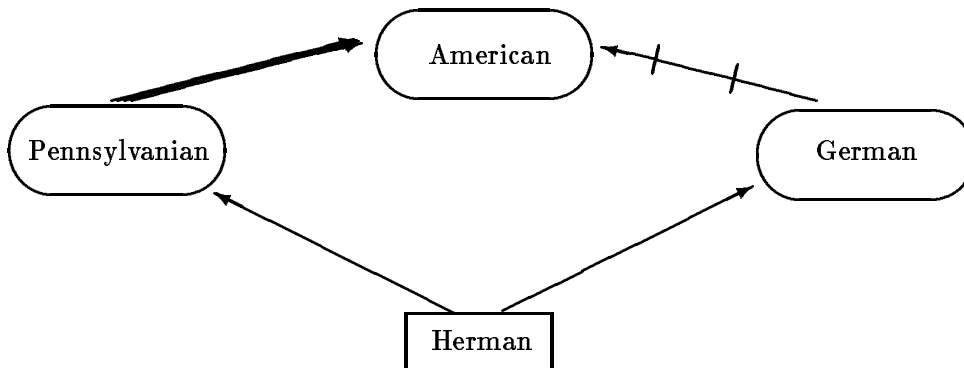


Figure 3: An unambiguous Nixon-like diamond

in our specifications. One consequence of such “awareness” is that we abstain from drawing conclusions that use $pacifist(Nixon)$ or $\neg pacifist(Nixon)$ as a premise. This approach to ambiguity is known as *skeptic* reasoning [15, 40]. A different, *credulous*, approach to handling ambiguity considers all possible worlds in which one of the conflicting pieces of evidence is sustained while the other is not. In such a world, only the inheritance that is associated with the sustained evidence takes place. Credulous approaches are known to be computationally expensive [9, 14, 22, 28, 33, 39] and will not be considered in this paper.

In the above examples, in order to inherit a property we examined various “evidences” supporting such inheritance. Actually, an evidence may not only support inheritance but also defeat it. For instance, in Figure 1, the evidence **penguin** defeats inheritance of $fly(Tweety)$, while the evidence **bird** supports it. Similarly, the evidence **penguin** supports the inheritance of $\neg fly(Tweety)$. In Figure 2, the two pieces of evidence, **quaker** and **republican**, defeat each other in case of Nixon but go unobstructed for *Penn* and *Bush*. In fact, for Nixon, the inheritance is defeated in a slightly different manner than in the case of Tweety. For Nixon, the two available pieces of evidence were *combined* and, having equal strength, they neutralized each other. The latter is based on a more general mechanism than the one used to explain the problem of Figure 1, for this allows to combine evidence obtained from different, incomparable paths in the network. To combine different pieces of evidence, we organize the space of evidence into a semi-lattice, as explained in Section 2.

An attractive aspect of our evidence-based approach is that many problems that do not have satisfactory representation in traditional inheritance networks, do have one in our formalism. Consider the following example, adapted from [16]:

- Hermann is born in Pennsylvania.
- Hermann is a native speaker of German.
- Typically, native speakers of German are not born in America.

- He who was born in Pennsylvania was also born in America.

Traditional representation of these facts gives rise to an ambiguous network of Figure 3 with the topology resembling that of Nixon diamond, except that the last “is-a” link is strict. So, we have two conflicting pieces of evidence: a *strict* one that supports the fact that Hermann is born in America; and a *defeasible* one that tells otherwise. Since strict evidence is of greater importance, it is appropriate to conclude that Hermann is American-born. To perform this kind of reasoning, conventional networks must be extended to include both strict and defeasible links. Several works, indeed, dealt with this issue [2, 8, 11, 16, 33]. The advantage of our formalism is that formalization of strict links comes at no additional cost and is just an instance of the general picture where various evidences are allowed to have different strength.

Another weakness of traditional network representation is that there are problems whose networks are isomorphic to Nixon diamond and yet there is no inherent ambiguity in the problem. For instance, a typical renal failure patient has high blood pressure, while a typical haemorrhage patient has low blood pressure [7]. If a person is known to suffer from both renal failure and haemorrhage, then a physician would conjecture that the patient has low blood pressure. This is because haemorrhage is a stronger evidence to the effect that the person has low blood pressure than renal failure suggests otherwise. However, other approaches (e.g., [39]) would not come to this conclusion, for renal patients and haemorrhage patients are incomparable classes of entities and the aforementioned preference cannot be represented.

There is also an unpleasant *asymmetry* in the way property inheritance is specified in conventional networks. An object can inherit a property because it belongs to some class, e.g., *pacifist*, but cannot inherit properties because it *does not* belong to a class, e.g., if it is not in the class *pacifist*. For instance, in traditional networks one can say “pacifists are peace-lovers,” but cannot say “nonpacifists are war-mongers.” More precisely, we would like to be able to model what we call *complimentary inheritance*, namely, the statements of the form “*compl-p is-a q*” or “*compl-p is-a compl-q*,” where *p*, *q* are properties and *compl-p*, *compl-q* are complementary properties that do not have direct representation in the inheritance network. As we shall see in the next section, this can be remedied at no extra cost in our framework.

The paper is organized as follows. Section 2 presents our approach, first informally—in Section 2.1, and then rigorously, in Sections 2.2 through 2.4. Sections 2.5 and 2.6 show that every inheritance specification can be given a unique meaning through the notion of supported models. Section 2.7 describes an algorithm for computing this meaning. In Section 3, we illustrate the capabilities of our approach on a number of benchmark examples, while Section 4 establishes a formal link between our logic language and the language of inheritance networks. Section 5 discusses the limitations of our approach and compares it to other theories of skeptical reasoning. Section 6 discusses possible extensions and concludes the paper.

2 An Evidence-Based Theory

2.1 Informal Introduction

The idea is to associate evidence of various degree of strength with properties that might be inherited by individuals. Adapting the ideas from [12], we order different pieces of evidence on two different scales: the *information scale* and the *truth scale*. On the truth scale, an evidence can belong to exactly one of the following *categories of support*: the category of *positive* evidences, the category of *negative* evidences, and the category of *ambiguous* evidences. Informally, an evidence in the first category would support an associated belief, while an evidence in the second category would defeat it. The third category contains evidences that neither support nor defeat, but rather represent the conflict in reasoner’s beliefs.

To see how this works, suppose we know that Batman is a mammal and that typically mammals do not fly. We can then “tentatively” conclude that Batman does not fly and represent this as a fact $fly(Batman) : -\mathbf{mammal}$, meaning that there is an evidence of strength \mathbf{mammal} suggesting that Batman does not fly. The minus sign, “-”, classifies the evidence as negative. If we further learn that Batman is a bat and a typical bat should be able to fly, we may augment our knowledge base with a new fact, $fly(Batman) : +\mathbf{bat}$, which asserts that there is a positive evidence (note the “+”-sign) of strength \mathbf{bat} that suggests that Batman does fly. Since knowing that someone is a bat is more informative than knowing that it is a mammal, we order these evidences as follows: $-\mathbf{mammal} <_k +\mathbf{bat}$, where \leq_k orders facts on the information scale mentioned earlier. It should be clear that specifying such an order is tantamount to programming and thus is a responsibility of the knowledge engineer. The latter would do this either explicitly or by drawing an inheritance network and then using the algorithm of Section 4. In our case, the resulting set of facts is:

$$\begin{aligned} fly(Batman) &: -\mathbf{mammal} \\ fly(Batman) &: +\mathbf{bat} \end{aligned} \tag{1}$$

It represents the current state of our knowledge, and the problem is to define the “most appropriate” possible world consistent with that knowledge. Since $+\mathbf{bat}$ is a stronger evidence than $-\mathbf{mammal}$, we should not hold on to the tentative belief that Batman does not fly. We capture the idea of discarding the weaker beliefs via the notion of satisfaction of a fact by an interpretation. Informally, an *interpretation* is a mapping from the set of all possible beliefs to the set of evidences. A fact of the form $belief : evidence$ is *satisfied* by an interpretation \mathcal{I} if $evidence \leq_k \mathcal{I}(belief)$. For example, if $\mathcal{I}(fly(Batman)) = +\mathbf{bat}$ then \mathcal{I} satisfies $fly(Batman) : -\mathbf{mammal}$ because, as stated above, $-\mathbf{mammal} <_k +\mathbf{bat}$. On the other hand, an interpretation J such that $J(fly(Batman)) = -\mathbf{mammal}$ does not satisfy $fly(Batman) : +\mathbf{bat}$.

A *model* for a set of facts is an interpretation that satisfies every fact in the set. For instance, the interpretation \mathcal{I} above is a model for (1); J does not satisfy the fact $fly(Batman) : +\mathbf{bat}$ and hence is not a model for (1).

Suppose now that in addition to (1) we knew $fly(Batman) : -injured_bat$, where $+bat <_k -injured_bat$. Then, a model \mathcal{I} for all the three facts about Batman would have to map $fly(Batman)$ to a value such that $-injured_bat \leq_k \mathcal{I}(fly(Batman))$. This, however, still does not let us conclude that Batman cannot fly because in reality the evidence in support of $fly(batman)$ can actually turn out to be something like $+injured_yet_flying_bat$. This scenario is quite arbitrary and contains extraneous information that is not implied by the given input. To rule such a scenario out, the usual technique is to restrict attention to minimal models only.¹ For our running example, any minimal model would have to map $fly(Batman)$ to $-injured_bat$, forcing the conclusion that Batman does not fly after all.

Things are not always that rosy, though. Consider the following set of facts:

$$\begin{aligned} pacifist(Nixon) &: +quaker \\ pacifist(Nixon) &: -republican. \end{aligned} \tag{2}$$

Here, $+quaker$ and $-republican$ are incomparable pieces of evidence with respect to the information scale and neither evidence dominates the other. A way to ensure that the minimal model for (2) will still be unique is to impose a structure of an upper semi-lattice on the space of evidences, so that $<_k$ will become the ordering induced by the semi-lattice structure. For instance, we can set \top to be the least upper bound of $+quaker$ and $-republican$. Then, (2) would have a minimal model that maps $pacifist(Nixon)$ to \top . In this case, \top can be viewed as a “combined” evidence for $pacifist(Nixon)$. Least upper bounds of pairs of incomparable evidences one of which is positive and the other negative will be usually put into the category of ambiguous evidences.

As another example, consider the statement “Flying objects usually have wings.” This can be represented in our formalism as follows:

$$has_wings(X) : +flies \leftarrow fly(X) : +\perp. \tag{3}$$

Here X is a variable that is implicitly universally quantified over the domain of all individual objects; $+\perp$ is the smallest positive evidence; and “ \leftarrow ” denotes implication in our logic. We would like to use this rule to infer that Tweety has wings whenever there is a positive evidence that it flies. However, we do not want to invoke this rule when the evidence is negative or ambiguous.

In annotated logics [5, 6, 18, 19, 20, 21], a rule is enabled in an interpretation whenever its body is satisfied by that interpretation. Since so far our notion of satisfaction was determined by the information ordering $<_k$, the above rule would be enabled in too many cases. For instance, if $\mathcal{I}(fly(Batman)) = -mammal$ then \mathcal{I} would enable (3), deriving $has_wings(Batman) : -flies$, if $+\perp <_k -mammal$. Since this is precisely the opposite of what we would want, we impose one more condition on when a rule can be applied to deduce new facts.

¹In our case, a model is minimal if, considered as a function from beliefs to evidences, it is minimal among functions representing all models of the Batman example.

We will say that an interpretation *strongly satisfies* a fact of the form *belief* : *evidence* if it satisfies this fact (in the old sense) and, in addition, *evidence* and $\mathcal{I}(\textit{belief})$ belong to the same partition with respect to the truth scale (i.e., both must be either positive, negative, or ambiguous). For example, if $\mathcal{I}(\textit{fly}(\textit{Tweety})) = +\textit{sparrow}$, then \mathcal{I} strongly satisfies $\textit{fly}(\textit{Tweety}) : +\textit{bird}$, given that $+\textit{bird} <_k +\textit{sparrow}$.² We can then say that a rule is *satisfied* by an interpretation if and only if whenever its body is strongly satisfied in the interpretation, its head is also satisfied (although, not necessarily strongly). Informally, this means that in order to invoke a rule, one must have an evidence that is not merely stronger than what the rule-body requires, but also belongs to the same category of support as the corresponding fact in the rule-body.

For general logic programs, minimal models do not adequately capture the semantics and more elaborate techniques, such as establishing a preference relation among minimal models [1, 27, 32], are used. For inheritance specifications, too, minimality does not suffice. As the following example shows, the notion of strong satisfaction introduced above is non-monotonic and hence it is no longer guaranteed that any set of facts has a unique minimal model. To see this, consider

$$\begin{aligned} \textit{fly}(\textit{Batman}) &: -\textit{mammal} \\ \textit{fly}(\textit{Batman}) &: +\textit{bat} \\ \textit{has_wings}(x) &: +\textit{flies} \leftarrow \textit{fly}(x) : +\perp. \end{aligned} \tag{4}$$

There are two incomparable minimal models: one is such that only

$$\begin{aligned} \mathcal{I}(\textit{fly}(\textit{Batman})) &= +\textit{bat} \\ \mathcal{I}(\textit{has_wings}(\textit{Batman})) &= +\textit{flies} \end{aligned}$$

hold true and the other is where the facts

$$\begin{aligned} \mathcal{J}(\textit{fly}(\textit{Batman})) &= -\textit{injured_bat} \\ \mathcal{J}(\textit{has_wings}(\textit{Batman})) &= +\perp \end{aligned}$$

hold but nothing else does. The first model reflects our intuition that a stronger evidence, $+\textit{bat}$, should dominate a weaker evidence, $-\textit{mammal}$. In contrast, the second model has a fundamental flaw, as far as our intuition goes. Indeed, there is no justification for the assumption $\textit{fly}(\textit{Batman}) : -\textit{injured_bat}$, even though this fact is consistent with the rest. To filter such flawed models out, we need to formalize the concept of “supportedness,” i.e., the situation in which every fact has a valid justification. Then, the *intended* semantics of a set of inheritance specifications could be defined using minimal supported models, and it will turn out that such a model is unique for any inheritance network. The next few subsections provide a rigorous account of the ideas that so far have been exposed only informally.

²As remarked earlier, this order has to be either specified by the programmer or inferred from the inheritance network. An algorithm for the latter task is given in Section 4.

2.2 Language

A *term* is an individual constant or a variable (we do not consider theories with function symbols in this paper). An *atom* is a propositional constant or a formula $q(t)$, where q is a unary predicate and t is a term—in order to talk about inheritance, we need only 0-ary and 1-ary predicates. *Literals* are of the form $p : \tau$, where p is an atom and τ is a *priority constant* drawn from the domain of priority constants \mathcal{P} ; priority constants play the role of “pieces of evidence” used in the discussion in the previous subsection. A *rule* is an expression of the form $p : \tau \leftarrow q : \gamma$, where p, q are atoms and τ, γ are priority constants. A *fact* is a *ground* (i.e., variable-free) literal. A *clause* is either a fact or a rule.

To capture dependency relationships among predicates, we define the “depends-on” relation among predicate symbols.

Definition 2.1 Given a set of rules \mathbf{P} and a pair of predicate symbols p and q , we write $q \prec_{\mathbf{P}} p$ (read p depends on q in \mathbf{P}) if and only if either there is a rule in \mathbf{P} of the form $\bar{p} : \tau \leftarrow \bar{q} : \gamma$, or, recursively, $\bar{p} : \tau \leftarrow \bar{r} : \lambda \in \mathbf{P}$ and $q \prec_{\mathbf{P}} r$, where $\bar{p}, \bar{q}, \bar{r}$ are atoms with predicate symbols p, q , and r , respectively. When $q \prec_{\mathbf{P}} p$ and \bar{p}, \bar{q} are atoms with predicate symbols p and q , we will also write $\bar{q} \prec_{\mathbf{P}} \bar{p}$. Occasionally, we may omit the subscript \mathbf{P} , if the set of rules is clear from the context or is immaterial.

An *inheritance specification* is a set of clauses, \mathbf{P} , such that $\prec_{\mathbf{P}}$ is an acyclic partial order on predicate symbols in \mathbf{P} . Actually, our results can be generalized to the case of *locally stratified* inheritance specifications a la [32], i.e., specifications such that $\prec_{\mathbf{P}}$ is acyclic on the set of ground atoms but may be cyclic on the set of predicate symbols. This would allow specifying cyclic inheritance networks. However, such a generalization complicates the semantics somewhat, taking the definitions in the direction of [19, 21].

2.3 Priority Constants

The priority constant τ in $p : \tau$ represents the *type* and the *relative strength* of evidence in support of p . Following [3, 4, 12, 37, 38, 10], priority constants are considered along two different dimensions: in one, evidence is ordered on the basis of *truth content*; in the other, it is ordered on the basis of *knowledge* or *information content*.

The “information dimension” is defined as follows. First, the set of priority constants \mathcal{P} is organized as a complete upper semi-lattice with the least upper bound operation lub_k .³ As usual, the least upper bound operation induces an order on \mathcal{P} , which we denote \leq_k and call the *information order*.

The “truth dimension” is organized using a quasi-order on \mathcal{P} , denoted \leq_t , which induces an equivalence relation \approx_t , where $\tau \approx_t \gamma$ if and only if $\tau \leq_t \gamma$ and $\gamma \leq_t \tau$. Additionally, \leq_t

³A *complete* upper semi-lattice is a semi-lattice in which a unique least upper bound is defined for sets of elements of *arbitrary* cardinality, not just for finite sets.

induces an ordering on the equivalence classes of \approx_t as follows: if $[\tau]$ and $[\gamma]$ are equivalence classes that contain τ and γ , respectively, then $[\tau] \leq_t [\gamma]$ if and only if $\tau \leq \gamma$.

For our purposes, we need \leq_t to be a *total* quasi-order and, furthermore, such that \approx_t has only the following three equivalence classes: \mathcal{C}^- , \mathcal{C}^* , and \mathcal{C}^+ . These contain priority constants corresponding to the defeating, ambiguous, and supporting evidences, respectively. We further assume that $\mathcal{C}^- \leq_t \mathcal{C}^* \leq_t \mathcal{C}^+$ and that not only \mathcal{P} but also each of these equivalence classes is a complete upper semi-lattice with respect to the information ordering \leq_k . This implies that each such set has a unique maximum element with respect to \leq_k . We denote these maximum elements by $-\omega$, $*\omega$, and $+\omega$, respectively, and assume $*\omega = \text{lub}_k(-\omega, +\omega)$. The intent here is to regard the facts annotated with these ω -constants as beliefs that are “totally false”, “ambiguous beyond repair”, and “absolutely true”, respectively. We will see that such beliefs cannot be defeated and this is precisely the mechanism that enables us to model strict inheritance links. Also, for modeling inheritance it is sometimes convenient to have a minimum element in each class of support \mathcal{C}^- , \mathcal{C}^* , and \mathcal{C}^+ . These will be denoted $-\perp$, $*\perp$, and $+\perp$. We further define:

- $(\tau <_k \gamma)$ if and only if $(\tau \leq_k \gamma) \wedge (\tau \neq \gamma)$; and
- $(\tau \leq_{tk} \gamma)$ if and only if $(\tau \leq_k \gamma) \wedge (\tau \approx_t \gamma)$.

We illustrate priority constants by translating the penguin triangle example into our annotated logic language. (See Section 2.1 for more examples.) The statement “*Tweety is a penguin*” can be translated as a fact $\text{penguin}(\text{Tweety}) : +\omega$, where the priority constant $+\omega$ stands for “absolutely true”. The default rule “*Birds fly*” can be translated as $\text{fly}(X) : +\text{bird} \leftarrow \text{bird}(X) : +\perp$, where the priority constant $+\text{bird}$ signifies a “bird”-strong supporting evidence, and the priority constant $+\perp$ stands for the smallest positive evidence. Similarly, the default rule “*Penguins do not fly*” can be translated as $\text{fly}(X) : -\text{penguin} \leftarrow \text{penguin}(X) : +\perp$, where the priority constant $-\text{penguin}$ signifies a “penguin”-strong defeating evidence. The strict rule “*Penguins are birds*” translates to a pair of rules: (1) $\text{bird}(X) : +\omega \leftarrow \text{penguin}(X) : +\omega$, to propagate definitive evidences in support of “penguinness” as definitive evidences in support of “birdness”; and (2) $\text{bird}(X) : +\text{penguin} \leftarrow \text{penguin}(X) : +\perp$, to propagate default conclusions appropriately. Finally, we postulate that $\text{bird} <_k \text{penguin}$, which can be inferred from the network of Figure 1, as described in Section 4.

2.4 Interpretations and Models

Let \mathbf{P} be an inheritance specification. The *domain* \mathcal{D} of Herbrand interpretations of \mathbf{P} is a collection of all individual constants mentioned in \mathbf{P} ; it is often called the Herbrand *universe* of \mathbf{P} . A *Herbrand base*, $\mathcal{B}_{\mathbf{P}}$ of \mathbf{P} is a collection of all ground (variable-free) atoms p that use only the constants of \mathcal{D} and the predicate symbols mentioned in \mathbf{P} . Since every inheritance specification is finite, both \mathcal{D} and $\mathcal{B}_{\mathbf{P}}$ are finite.

A *Herbrand interpretation* \mathcal{I} of \mathbf{P} is a partial mapping from the Herbrand base, $\mathcal{B}_{\mathbf{P}}$, of \mathbf{P} to the set of priority constants \mathcal{P} . Given two interpretations \mathcal{I} and \mathcal{J} , $\mathcal{J} \sqsubseteq_k \mathcal{I}$ (resp., $\mathcal{J} \sqsubseteq_{tk} \mathcal{I}$) iff for every atom $p \in \mathcal{B}_{\mathbf{P}}$, such that $\mathcal{J}(p)$ is defined, $\mathcal{I}(p)$ is also defined and $\mathcal{J}(p) \leq_k \mathcal{I}(p)$ (resp., $\mathcal{J}(p) \leq_{tk} \mathcal{I}(p)$). An interpretation \mathcal{I} is *minimal* in a set of interpretations \mathcal{S} if and only if there is no $\mathcal{J} \in \mathcal{S}$ such that $\mathcal{J} \sqsubseteq_k \mathcal{I}$ and $\mathcal{J} \neq \mathcal{I}$; \mathcal{I} is *maximal* if and only if there is no $\mathcal{J} \in \mathcal{S}$ such that $\mathcal{I} \sqsubseteq_k \mathcal{J}$ and $\mathcal{J} \neq \mathcal{I}$.

A ground atom $p : \tau$ is *satisfied* in \mathcal{I} , denoted $\mathcal{I} \models_k p : \tau$, if and only if $\mathcal{I}(p)$ is defined and $\tau \leq_k \mathcal{I}(p)$; it is *strongly satisfied* in \mathcal{I} , denoted $\mathcal{I} \models_{tk} p : \tau$, if also $\tau \leq_{tk} \mathcal{I}(p)$.

A ground (variable-free) rule $p : \tau \leftarrow q : \gamma$ is *satisfied* in \mathcal{I} if and only if $\mathcal{I} \models_{tk} q : \gamma$ implies $\mathcal{I} \models_k p : \tau$. (Note: this is consistent with thinking of the facts as if they were rules with the empty body, which is always strongly satisfied in all interpretations.) A nonground rule $p : \tau \leftarrow q : \gamma$ is *satisfied* in \mathcal{I} if and only if all its ground instances are satisfied in \mathcal{I} . (A *ground instance* of a rule r is a ground rule obtained from r by replacing variables with constants, where different occurrences of the same variable are replaced by the same constant.)

An inheritance specification \mathbf{P} is *satisfied* in \mathcal{I} , if all its clauses are satisfied in \mathcal{I} . An interpretation \mathcal{I} is a *model* of a set of clauses \mathbf{P} if and only if \mathbf{P} is satisfied under \mathcal{I} . A Herbrand model \mathcal{I} of \mathbf{P} is *supported* by \mathbf{P} if for every atom p such that $\mathcal{I}(p)$ is defined, we have

$$\mathcal{I}(p) = \text{lub}_k \{ \tau \mid p : \tau \leftarrow q : \gamma \text{ is a ground instance of a rule in } \mathbf{P} \text{ and } \mathcal{I} \models_{tk} q : \gamma \}.$$

Here and henceforth we postulate that the least upper bound of an empty set is undefined. In particular, this means that \mathcal{I} must be undefined on any atom that does not appear in the head of a ground instance of a rule in \mathbf{P} . Informally, \mathcal{I} is supported by \mathbf{P} if the evidences that it assigns to atoms are not stronger than what is warranted by \mathbf{P} . The reader can verify that every supported model is also \sqsubseteq_k -minimal. (The requirement that “ \prec ” is an acyclic ordering on predicate symbols is crucial for this to be true.)

2.5 Existence of Supported Models

In this section, we show that every inheritance specification has a supported model.

Lemma 2.1 *Every inheritance specification \mathbf{P} that consists of facts only admits a supported Herbrand model.*

Proof: Consider an inheritance specification

$$\mathbf{P} = \{ p_1 : \gamma_1, p_2 : \gamma_2, \dots, p_n : \gamma_n, \dots \}.$$

We construct a supported Herbrand model \mathcal{M} of \mathbf{P} as follows. For each p , $\mathcal{M}(p) = \text{lub}_k \{ \gamma_j \mid p = p_j \}$; $\mathcal{M}(p)$ is undefined if this set is empty. \mathcal{M} is a model of \mathbf{P} , as it satisfies every fact in \mathbf{P} ; it is a supported model by construction. \square

By *stratifying* a general inheritance specification \mathbf{P} relatively to the $\prec_{\mathbf{P}}$ -ordering (see Definition 2.1), a supported model can be constructed as in [1].

Lemma 2.2 *Every inheritance specification \mathbf{P} admits a supported Herbrand model.*

Proof: Define the *stratum* of a predicate symbol p as follows:

$$\text{strat}(p) = \begin{cases} 0 & \text{if } p \text{ is } \prec_{\mathbf{P}}\text{-minimal} \\ 1 + \text{max_body_strat}(p) & \text{otherwise,} \end{cases}$$

where $\text{max_body_strat}(p) = \text{MAX}\{\text{strat}(q) \mid p : \gamma \leftarrow q : \tau \in \mathbf{P}\}$.

Let \mathbf{P}_i be a subset of \mathbf{P} that contains all clauses whose head predicate is in the stratum of the level i and below. Define \mathcal{I}_i to be the following set of facts:

$$\{p : \gamma \mid \text{strat}(p) = i, p : \gamma \leftarrow q : \tau \text{ is a ground instance of a rule in } \mathbf{P} \text{ and } \mathcal{M}_{i-1} \models_{tk} q : \tau\},$$

where, for $i \geq 0$, \mathcal{M}_i is the supported model determined by the set of facts $\bigcup_{j=0}^i \mathcal{I}_j$; \mathcal{M}_{-1} is the interpretation that is undefined everywhere (or, equivalently, the supported model determined by the empty set of facts). The existence of \mathcal{M}_i follows from Lemma 2.1. Since \mathbf{P} has no recursion, it follows by induction that \mathcal{M}_i is a supported model of \mathbf{P}_i . Thus, \mathcal{M}_n , where n is the maximal stratum of \mathbf{P} , is a supported model of \mathbf{P} . \square

2.6 Uniqueness of Supported Models

We now prove that every inheritance specification has a unique supported model.

Lemma 2.3 *For an inheritance specification \mathbf{P} that consists of facts only, there exists a unique supported Herbrand model of \mathbf{P} .*

Proof: Existence follows from Lemma 2.1. The supported model is unique because the least upper bound of any set of priority constants is unique. \square

Since the above lemma associates a unique model with every set of facts, it becomes possible to represent interpretations via such sets. Sometimes it will even be convenient to slightly abuse the language and instead of defining an interpretation for a situation at hand, to present a set of facts that uniquely determines the desired interpretation.

Given an inheritance specification \mathbf{P} , we define an operator $\mathcal{T}_{\mathbf{P}}$ that maps a Herbrand interpretation \mathcal{I} into a Herbrand interpretation $\mathcal{T}_{\mathbf{P}}(\mathcal{I})$ as follows.

Definition 2.2 $\mathcal{T}_{\mathbf{P}}(\mathcal{I})$ is the supported model of the set of literals $\{p : \tau \mid p : \tau \leftarrow q : \gamma \text{ is a ground instance of a clause in } \mathbf{P} \text{ and } \mathcal{I} \models_{tk} q : \gamma\}$, which exists and is unique by Lemma 2.3.

Note that $\mathcal{T}_{\mathbf{P}}$ is not monotonic with respect to \sqsubseteq_k . For instance, let \mathbf{P} consist of a single rule $p : -1 \leftarrow q : +1$ and consider a pair of interpretations \mathcal{I} and \mathcal{J} such that $\mathcal{I}(p) = -1$, $\mathcal{I}(q) = +2$ and $\mathcal{J}(p) = -2$, $\mathcal{J}(q) = -3$. Here we assume that $\mathcal{P} = \{\pm 1, \pm 2, \pm 3\}$, $i \leq_k j$ if and only if $|i| \leq |j|$ ($|i|$ is the absolute value of i), and $i \approx_t j$ whenever they have the same sign. Clearly, $\mathcal{I} \sqsubseteq_k \mathcal{J}$. However, $\mathcal{T}_{\mathbf{P}}(\mathcal{I}) = \{p : -1\}$ while $\mathcal{T}_{\mathbf{P}}(\mathcal{J}) = \emptyset$. Fortunately, $\mathcal{T}_{\mathbf{P}}$ has a weaker property that saves the situation; namely, if $\mathcal{I} \sqsubseteq_{tk} \mathcal{J}$ then $\mathcal{T}_{\mathbf{P}}(\mathcal{I}) \sqsubseteq_k \mathcal{T}_{\mathbf{P}}(\mathcal{J})$.

Lemma 2.4 $\mathcal{T}_{\mathbf{P}}(\mathcal{I}) \sqsubseteq_k \mathcal{I}$ if and only if \mathcal{I} is a model of \mathbf{P} .

Proof: (\Leftarrow) Let \mathcal{I} be a model of \mathbf{P} , p be an atom, and $\mathcal{I}(p) = \tau$. Then $\text{lub}_k\{\lambda \mid p : \lambda \leftarrow q : \gamma\}$ is a ground instance of a rule in \mathbf{P} such that p is in the head of the rule and $\mathcal{I} \models_{tk} q : \gamma \leq_k \tau$. Thus, $\mathcal{T}_{\mathbf{P}}(\mathcal{I}) \sqsubseteq_k \mathcal{I}$.

(\Rightarrow) For every ground instance $p : \lambda \leftarrow q : \gamma$ of a rule in \mathbf{P} , if $\mathcal{I} \models_{tk} q : \gamma$ then $\mathcal{T}_{\mathbf{P}}(\mathcal{I}) \models_k p : \lambda$. As $\mathcal{T}_{\mathbf{P}}(\mathcal{I}) \sqsubseteq_k \mathcal{I}$, every such instance is satisfied by \mathcal{I} , i.e., \mathcal{I} is a model of \mathbf{P} . \square

The above lemma and the definition of supportedness yields the following result:

Lemma 2.5 $\mathcal{T}_{\mathbf{P}}(\mathcal{I}) = \mathcal{I}$ if and only if \mathcal{I} is a supported model of \mathbf{P} .

We show the existence and uniqueness of a supported model for an inheritance specification in two steps. For convenience of exposition and without loss of generality, we assume that the sets of predicate symbols that occur in the facts and of those predicates that appear in the heads of the rules are disjoint, in accordance with the tradition of separation of intensional and extensional parts of the database. It is known that any specification can be transformed into an equivalent one satisfying this assumption [43].

Theorem 2.1 *Every inheritance specification \mathbf{P} admits at most one supported Herbrand model. Equivalently, there exists at most one solution to the fixpoint equation $\mathcal{T}_{\mathbf{P}}(\mathcal{I}) = \mathcal{I}$.*

Proof: The proof is by contradiction. Let there exist two fixpoints \mathcal{F}_1 and \mathcal{F}_2 , of $\mathcal{T}_{\mathbf{P}}$ such that for some atom p , $\mathcal{F}_1(p) \neq \mathcal{F}_2(p)$. Let also p be a \prec -minimal such atom. Since predicate symbols in the facts and those occurring in the rule heads are distinct, we conclude by Lemma 2.3 that \mathcal{F}_1 and \mathcal{F}_2 must agree (as functions) on the atoms that appear as facts in \mathbf{P} . In particular, p cannot be one of the fact-atoms. By the \prec -minimality assumption about p , \mathcal{F}_1 and \mathcal{F}_2 should agree on all facts q such that $q \prec p$. Because \prec is acyclic and lub_k is a function that maps any given set of priority constants to a unique value, we conclude that $\mathcal{F}_1(p) = \mathcal{F}_2(p)$, contrary to the assumption. Hence, \mathcal{F}_1 and \mathcal{F}_2 must agree on all atoms. \square

Theorem 2.2 *Every inheritance specification \mathbf{P} has a unique supported Herbrand model $\mathcal{M}_{\mathbf{P}}$.*

Proof: Follows from the above theorem and Lemma 2.2. \square

The unique supported model for \mathbf{P} , denoted $\mathcal{M}_{\mathbf{P}}$, can be described as $\lim_{i \rightarrow \infty} \mathcal{T}_{\mathbf{P}}^i(\emptyset)$. But this does not give us an efficient way of computing the fixpoint of $\mathcal{T}_{\mathbf{P}}$ because the successive approximations change nonmonotonically. However, as in [1], we can build $\mathcal{M}_{\mathbf{P}}$ monotonically by an iterated fixpoint construction. This is illustrated by the following example. Consider the inheritance specification \mathbf{P} that uses the set $\mathcal{P} = \{\pm 1, \pm 2, \pm 3\}$ of priority constants where $i \leq_k j$ if and only if $|i| \leq |j|$ and the sign $+$ or $-$ indicates the category of support:

1. $b(a) : +3$
2. $m(a) : +3$
3. $t(X) : +3 \leftarrow m(X) : +3$
4. $f(X) : +1 \leftarrow b(X) : +3$
5. $f(X) : -2 \leftarrow t(X) : +3$
6. $l(X) : +1 \leftarrow f(X) : +1$.

We have $m \prec t \prec f \prec l$ and $b \prec f$. Subsequent applications of $\mathcal{T}_{\mathbf{P}}$ result in the following sequence of interpretations, where we represent interpretations via sets of facts, which is possible in view of Lemma 2.3.

$$\begin{aligned}
\mathcal{T}_{\mathbf{P}}^0(\emptyset) &= \emptyset; \\
\mathcal{T}_{\mathbf{P}}^1(\emptyset) &= \{ b(a) : +3, m(a) : +3 \}; && \text{(use rules 1 and 2)} \\
\mathcal{T}_{\mathbf{P}}^2(\emptyset) &= \mathcal{T}_{\mathbf{P}}^1(\emptyset) \cup \{ t(a) : +3, f(a) : +1 \}; && \text{(use rules 1 through 4)} \\
\mathcal{T}_{\mathbf{P}}^3(\emptyset) &= \mathcal{T}_{\mathbf{P}}^2(\emptyset) \cup \{ l(a) : +1, f(a) : -2 \}; && \text{(use rules 1 through 6)} \\
\mathcal{T}_{\mathbf{P}}^4(\emptyset) &= \mathcal{T}_{\mathbf{P}}^3(\emptyset) \setminus \{ l(a) : +1 \}; && \text{(use rules 1 through 6)} \\
\mathcal{T}_{\mathbf{P}}^{i+1} &= \mathcal{T}_{\mathbf{P}}^i, \text{ if } i \geq 4; \\
\mathcal{T}_{\mathbf{P}}(\emptyset) &= \mathcal{T}_{\mathbf{P}}^4(\emptyset).
\end{aligned}$$

The interpretation $\mathcal{T}_{\mathbf{P}}(\emptyset)$ represents the unique supported model of \mathbf{P} . Observe how $l(a) : +1$ in $\mathcal{T}_{\mathbf{P}}^3(\emptyset)$ was nonmonotonically withdrawn, since $\mathcal{T}_{\mathbf{P}}^4(\emptyset) \not\models_{tk} l(a) : +1$.

On the other hand, we could build the same model monotonically, by applying the rules in accordance with the \prec -order on predicate symbols in \mathbf{P} :

1. $\mathcal{M}_{\mathbf{P}}^0 = \emptyset;$
2. $\mathcal{M}_{\mathbf{P}}^1 = \{ b(a) : +3, m(a) : +3 \};$ (use rules 1 and 2)
3. $\mathcal{M}_{\mathbf{P}}^2 = \mathcal{M}_{\mathbf{P}}^1 \cup \{ t(a) : +3 \};$ (use rule 3)

4. $\mathcal{M}_{\mathbf{P}}^3 = \mathcal{M}_{\mathbf{P}}^2 \cup \{f(a) : +1, f(a) : -2\}$; (use rules 4 and 5)
5. $\mathcal{M}_{\mathbf{P}}^{i+1} = \mathcal{M}_{\mathbf{P}}^i$, if $i \geq 3$; hence
6. $\mathcal{M}_{\mathbf{P}} = \mathcal{M}_{\mathbf{P}}^3$.

Note that in (5) the rule $l(X) : +1 \leftarrow f(X) : +1$ cannot be applied since $\mathcal{M}_{\mathbf{P}}^3 \models_{tk} f(a) : +1$. This monotonic construction of the minimal supported model constitutes a basis for the inheritance algorithm in the next subsection.

2.7 Inheritance Algorithm

Let \mathbf{H} be a set of ground atoms, \mathcal{A}_i be a set of atoms whose literals are added by $\mathcal{T}_{\mathbf{P}}$ at the i th step of the iteration, and \mathcal{M}_i be the set of literals at the i th step of the iteration representing the intermediate state in the computation of the supported model for \mathbf{P} . Then:

- $\mathcal{A}_0 = \emptyset$.
- \mathcal{M}_0 is the \sqsubseteq_k -minimal interpretation (represented as a set of facts) satisfying all ground facts in \mathbf{P} .
- $\mathcal{A}_{i+1} = \mathcal{A}_i \cup \{p \mid p \text{ is a } \prec\text{-minimal ground atom in } (\mathbf{H} - \mathcal{A}_i)\}$, where “ \prec ” is the “depends on” relation defined in Section 2.
- \mathcal{M}_{i+1} is the \sqsubseteq_k -minimal interpretation (represented as a set of facts) satisfying $\mathcal{M}_i \cup \{p : \tau \mid p \in \mathcal{A}_{i+1}, p : \tau \leftarrow q : \gamma \text{ is a ground instance of a clause in } \mathbf{P} \text{ and } \mathcal{M}_i \models_{tk} q : \gamma\}$.
- $\mathcal{M}_{\mathbf{P}} = \bigcup_i \mathcal{M}_i$.

Clearly, if \mathbf{P} is a *finite* set of inheritance specifications, the above algorithm terminates, since each step involves only a finite number of operations (recall that the universe \mathcal{D} and the Herbrand base $\mathcal{B}_{\mathbf{P}}$ are finite) and the number of strata is finite, too.

Theorem 2.3 *The set of literals $\mathcal{M}_{\mathbf{P}}$ computed by the above algorithm determines a supported model of \mathbf{P} .*

Proof: Let \mathbf{P}_i denote the subset of clauses in \mathbf{P} whose heads are atoms from \mathcal{A}_i . We prove by induction on i that \mathcal{M}_i is a supported model of \mathbf{P}_i .

Basis: $j = 0$. Clearly, \mathbf{P}_0 contains only the facts of \mathbf{P} . Hence, by Lemma 2.3, \mathcal{M}_0 is the unique supported model of \mathbf{P}_0 .

Induction step: Suppose \mathcal{M}_j is the unique supported model of \mathbf{P}_j . We show that for $j = i + 1$, \mathcal{M}_j is the supported model of \mathbf{P}_j . Note that for each i , $(\mathcal{A}_{i+1} - \mathcal{A}_i) \cap \mathcal{A}_i = \emptyset$, and thus the meaning of the predicates computed in the i th iteration does not affect the meaning

of predicates computed in the preceding iterations. Thus, \mathcal{M}_i contains \mathcal{M}_{i-1} and satisfies \mathbf{P}_{i-1} . From the definition of \mathcal{M}_i , the new literals added to \mathcal{M}_i at the $(i + 1)$ th step, i.e., the literals satisfied by \mathcal{M}_{i+1} but not by \mathcal{M}_i , are obtained by taking the lub_k of all supported conclusions from all rules whose bodies are strongly satisfied in \mathcal{M}_i . By the definition of satisfaction of a rule, all rules with head predicates in $(\mathcal{A}_{i+1} - \mathcal{A}_i)$ will be satisfied in \mathcal{M}_{i+1} . Thus, \mathcal{M}_{i+1} is a model of \mathbf{P}_{i+1} and this model is supported by the construction. Hence, $\mathcal{M}_{\mathbf{P}}$ is a supported model of \mathbf{P} . \square

Complexity of this algorithm follows from the standard results about the bottom-up computation of Datalog programs [43]:

Theorem 2.4 *If the set of rules in \mathbf{P} is fixed but the set of facts may vary, the above inheritance algorithm takes time polynomial in the size of the set of facts.⁴ If the set of rules is allowed to vary, then the complexity of the above algorithm is exponential in the size of \mathbf{P} .*

3 Examples

In this section, we illustrate our approach on a number of examples. We will use the semi-lattice of Figure 4 in order to avoid long mnemonics for priority constants and also to illustrate a method for programming inheritance networks⁵ that is different from the one described in Section 4. Priority constants are now numbers marked with one of the signs “+”, “−”, or “*”. All constants with the same sign are equivalent with respect to \approx_t , i.e., $-1 \approx_t -3$, $*2 \approx_t *5$, and so on. Intuitively, annotation constants $+1, +2, \dots$ denote the degree of confidence that the corresponding atom is true; the constants $-1, -2, \dots$ denote the degree of confidence in the falsehood of the atom. The constants $*1, *2, \dots$ represent the degree of ambiguity found in the evidence. The constants $+\omega, -\omega$, and $*\omega$ represent information about “absolute” truth, falsehood, and ambiguity, respectively.

To determine whether a given individual i has property p with respect to an inheritance specification \mathbf{P} , one needs to evaluate $\mathcal{M}_{\mathbf{P}}(p(i))$, where $\mathcal{M}_{\mathbf{P}}$ is the unique supported model of \mathbf{P} . To see how this works, let us reformulate the penguin triangle example in our logic:

- $fly(X) : +1 \leftarrow bird(X) : +1$
- $fly(X) : -2 \leftarrow penguin(X) : +1$
- $bird(X) : +\omega \leftarrow penguin(X) : +\omega$
- $penguin(Tweety) : +\omega$.

⁴This is known as *data* complexity of an algorithm [41].

⁵To us, drawing inheritance networks or specifying inheritance via logic rules are just two of the many possible ways of programming taxonomies.

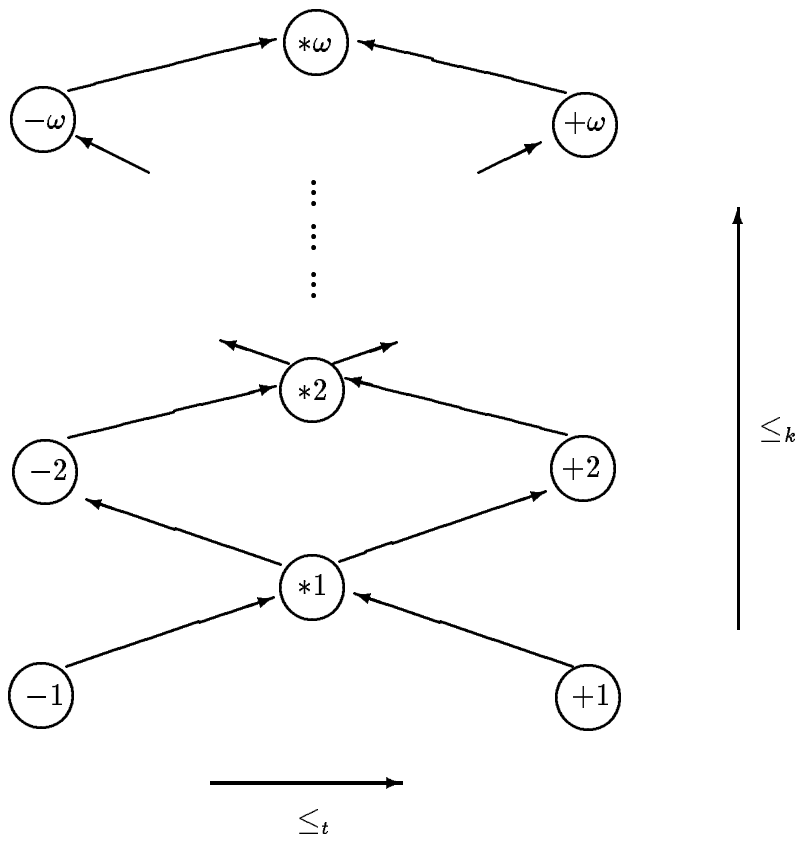


Figure 4: Information ordering for skeptical semantics

The relative magnitudes of priority constants in the heads of the rules are designed to capture the intuition that the default conclusion obtained by applying the second rule takes precedence over the conclusion obtained by the first rule. Note that the absolute magnitude of priority constants has no special significance—only the relative strength counts. Furthermore, the strength of evidence in the body has no bearing on the strength of evidence in the conclusion. It is the priority constant in the head of the rule that determines strength of the evidence contributed to the conclusion in the rule-head. The priority constant in the body is kind of a threshold that dismisses evidence that is too weak to be counted.

Priority constants need not be restricted to numerical values. In fact, as is explained in the next section, one can simply use names of the nodes in the network as priority constants, and the information ordering will be determined by the network topology. This is illustrated below by rewriting the first two defeasible rules of the penguin triangle example, where $+\perp$ is the smallest positive evidence, as follows:

- $fly(X) : +bird \leftarrow bird(X) : +\perp$
- $fly(X) : -penguin \leftarrow penguin(X) : +\perp$.

The ordering $<_k$ is set-up to be consistent with the links of the network. Hence, since the network of Figure 1 has a directed path from *penguin* to *bird*, we postulate that $+bird <_k -penguin$ (see Section 4 for additional details). Therefore, for Tweety, to be a penguin outweighs the fact that it is a bird, yielding the conclusion that Tweety does not fly.

Our theory supports both strict and defeasible links. To “program in” the fact that some of the rules are strict, one can use rules of the form $p : +\omega \leftarrow q : +\omega$, $p : -\omega \leftarrow q : +\omega$, or $p : +\omega \leftarrow q : -\omega$. For instance, for the example in Figure 3, we would write:

- $born_in_Pennsylvania(Hermann) : +\omega$
- $native_speaker_of_German(Hermann) : +\omega$
- $born_in_US(X) : -3 \leftarrow native_speaker_of_German(X) : +1$
- $born_in_US(X) : +\omega \leftarrow born_in_Pennsylvania(X) : +\omega$.

Even though there are conflicting conclusions about Hermann’s country of birth, the conflict is resolved in favor of the U.S., for the strict conclusion derived by the last rule will prevail over the default conclusion obtained from the third rule.

To illustrate the treatment of ambiguity, consider the extended Nixon diamond.

- $republican(Nixon) : +\omega$
- $quaker(Nixon) : +\omega$
- $pacifist(X) : -1 \leftarrow republican(X) : +1$

- $pacifist(X) : +1 \leftarrow quaker(X) : +1$
- $dove(X) : +1 \leftarrow pacifist(X) : +1$
- $hawk(X) : +1 \leftarrow pacifist(X) : -1$.

Here, the unique supported model contains $pacifist(Nixon) : *1$, meaning that the evidence that Nixon is a pacifist is ambiguous. Therefore, the last two rules cannot be invoked because their bodies are not strongly satisfied in the supported model (as $pacifist(Nixon) : *1$ and $pacifist(X) : \pm 1$ belong to different support classes). Since there is no unambiguous evidence for $pacifist(Nixon)$, neither $hawk(Nixon)$ nor $dove(Nixon)$ are derived. This behaviour is similar to skeptical reasoning described in [15].

Another interesting feature of our formalism is the ability to model what we called “complimentary inheritance” in Section 1, i.e., the statements of the form “*compl-p is-a q*” or “*compl-p is-a compl-q*,” where p and q are properties directly represented in the inheritance network and *compl-p* and *compl-q* are their complimentary properties that do not have direct representation (properties r and s are complimentary if classes of entities defined by these properties are complimentary sets). For instance, the statements “unemployed persons do not pay taxes” and “a person is guilty if proven not innocent” can be represented as follows:

$$\begin{aligned} tax_payer(X) : -1 &\leftarrow employed(X) : -\perp \\ guilty(X) : +1 &\leftarrow innocent(X) : -\omega. \end{aligned}$$

To specify characteristic properties associated with a class, such as “all living things breathe” and “only adult citizens have voting rights,” we may write

$$\begin{aligned} breathes(X) : +\omega &\leftarrow living(X) : +\omega \\ breathes(X) : -\omega &\leftarrow living(X) : -\omega \\ has_voting_rights(X) : +1 &\leftarrow adult_citizen(X) : +1 \\ has_voting_rights(X) : -1 &\leftarrow adult_citizen(X) : -1. \end{aligned}$$

On the other hand, we cannot specify mutually exclusive properties. For instance,

$$\begin{aligned} dove(X) : -1 &\leftarrow hawk(X) : +1 \\ hawk(X) : -1 &\leftarrow dove(X) : +1 \end{aligned}$$

is not a legal inheritance specification, since the predicates *dove* and *hawk* (and even the corresponding ground facts) cannot be ordered with respect to “ \prec ” here.

Our logic has certain advantages over the formalism presented in [15] that accrue out of being able to order the space of evidences to help resolve ambiguity, as in the Blood Pressure example of Section 2.1.

- $Renal_Failure_Patient(Tom) : +\omega$

- $Haemorrhage_Patient(Tom) : +\omega$
- $High_Blood_Pressure(X) : +1 \leftarrow Renal_Failure_Patient(X) : +\omega$
- $High_Blood_Pressure(X) : -2 \leftarrow Haemorrhage_Patient(X) : +\omega$.

Even though the topology of this network resembles that of Nixon diamond, we may resolve the ambiguity in favor of *Tom* having low blood pressure; [15] cannot differentiate between these two cases.

Our model theory, in particular, the notion of supportedness, buys us certain advantages over [12]. Consider the following rules:

- $bird$ (Fact)
- $bird \rightarrow flies$ (Defeasible rule)
- $flies \rightarrow \neg acro$ (Strict rule)

According to [12], this specification admits two \square_k -minimal models, where the domain of truth values used in [12], its truth ordering, and the information ordering are shown in Figure 5. In one model, *flies* is true by default and *acro* is false by default; in the other, *flies* is false and *acro* is unknown.

Ginsberg notes that this does not allow one to conclude that the bird flies by default, because such conclusion would not be true in all \square_k -minimal models. He goes on to propose a criterion to prefer the first model over the second one by formalizing the notion of “strength of assumption” and discarding the second model by virtue of being based on “unreasonably strong” assumption, compared to the first model. We, on the other hand, do not run into such problem. In our formalism, the problem is represented as follows:

- $bird : +\omega$
- $flies : +1 \leftarrow bird : +1$
- $acro : -\omega \leftarrow flies : +\omega$
- $acro : -1 \leftarrow flies : +1$.

This admits a unique supported model that contains $flies : +1$ and $acro : -1$. The interpretation corresponding to the second minimal model in [12] is not supported because $flies : -\omega$ holds true (i.e., *flies* is false) does not have any justification.

An advantage of our approach is that inheritance specifications can be designed in the same style as Prolog programs. A system that executes these specifications can be thought of as an elementary Truth Maintenance System (TMS). In particular, it would store “tentative” justifications for the association of a property with an individual. When new facts are added

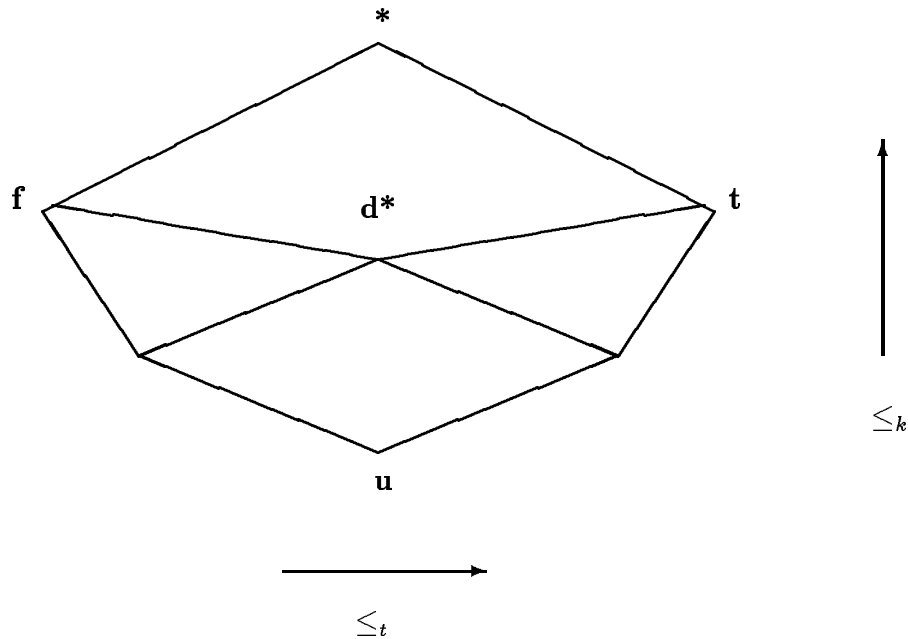


Figure 5: A bilattice for default reasoning

to the system, certain old conclusions may be nonmonotonically retracted and new conclusions asserted. The key difference is that in a general belief revision system the main concern is the inconsistency resolution procedure, which is usually quite expensive, since a TMS has to track down the possible sources of inconsistency and devise strategies for belief revision. In contrast, our use of semi-lattices leads to an inexpensive and systematic way of resolving inconsistency. In addition, when such resolution is not possible, a contradiction may be tolerated by representing it explicitly, via the priority constants that denote ambiguity.

4 Inheritance Networks as Logical Specifications

It is possible to translate inheritance networks into our logic to provide an indirect (but formal) semantics to these networks. The main question here is how to choose priority constants and design a suitable semi-lattice structure. In what follows, we outline an approach to translating inheritance networks that uses *node labels* to define priority constants. Although, perhaps, simplistic, this translation seems to work adequately. One of its features is that it automatically infers a semi-lattice structure from the network topology.

Define the *evidence set* corresponding to a network to be $\mathcal{E} = \{+, -\} \times (\{\perp, \omega\} \cup \mathbf{C})$, where \mathbf{C} is a set of names of property nodes in the network. The first component of an element

of \mathcal{E} specifies the class of support, while the second component captures the information content. The constants $+\perp$ and $-\perp$ represent the minimum positive and negative evidence; $+\omega$ and $-\omega$ represent the maximum positive and negative evidence, respectively. The “is-a” and “is-not-a” relationships between property nodes in the network determine the ordering $<_k$ on \mathcal{E} : if q is-a p then $\varepsilon p <_k \delta q$, where ε and δ stand for one of the signs, “+” or “-”, and p, q are constants in \mathbf{C} that correspond to the property nodes p and q . For instance, we would have $+\text{bird} <_k -\text{penguin}$ for the network in Figure 1, because *penguin* is-a *bird*. We also define $\pm\perp <_k \pm p <_k \pm\omega$ for every $p \in \mathbf{C}$.

We define the domain \mathcal{P} of priority constants to be the set of all *reduced* subsets of \mathcal{E} , i.e., subsets that do not contain $<_k$ -comparable elements. The $<_k$ -order on \mathcal{P} is defined so that $\pi_1 <_k \pi_2$ if and only if for every $e \in \pi_1$ there is $e' \in \pi_2$ such that $e <_k e'$. The least upper bound, lub_k , is determined by taking the set-theoretic union of the elements of \mathcal{P} and then reducing the result with respect to $<_k$. The class of support \mathcal{C}^+ (resp., \mathcal{C}^-) consists of those elements of \mathcal{P} that are composed of only positive (resp., negative) elements of \mathcal{E} ; the sets in \mathcal{C}^* are mixed, i.e., they contain positive as well as negative elements of \mathcal{E} .

As an illustration, consider the penguin triangle example. Having inferred $\text{fly}(\text{Tweety}) : \{+\text{bird}\}$ and $\text{fly}(\text{Tweety}) : \{-\text{penguin}\}$, we can see that only the latter survives, since $\{+\text{bird}\} <_k \{-\text{penguin}\}$. For an example with ambiguity, consider Nixon diamond. The two evidences here are $\{-\text{republican}\}$ and $\{+\text{quaker}\}$ and their lub_k is $\{-\text{republican}, +\text{quaker}\}$. Because $-\text{republican}$ and $+\text{quaker}$ are incomparable on the information scale and have different signs, the resulting evidence belongs to \mathcal{C}^* and is, therefore, ambiguous.

In general, given an inheritance diagram, we translate it into our logical representation and then take the unique supported model of that representation for the meaning of this diagram. The translation is performed as follows. We associate a constant i with each individual node i and a unary predicate p with each property node p . Each such node also has associated priority constants $+p$ and $-p$, as explained earlier. Note that this algorithm uses names of properties for priority constants, rather than just numbers. Therefore, for well-designed networks, priority constants would have meaningful mnemonics, such as **bird**, **quaker**, etc.

A positive (resp., negative) link from an individual node i to a property node p is represented by the fact $p(i) : \{+\omega\}$ (resp., $p(i) : \{-\omega\}$). The translation of a positive strict link from a property node p to a property node q is the following *pair* of rules:

$$\begin{aligned} q(X) : \{+\omega\} &\leftarrow p(X) : \{+\omega\} \\ q(X) : \{+p\} &\leftarrow p(X) : \{+\perp\}. \end{aligned}$$

The first rule above propagates strict facts through positive strict links, yielding strict conclusions. The second rule propagates defeasible facts through strict links, yielding defeasible conclusions. Similarly, a strict negative link from p to q would be represented as a pair of rules:

$$\begin{aligned} q(X) : \{-\omega\} &\leftarrow p(X) : \{+\omega\} \\ q(X) : \{-p\} &\leftarrow p(X) : \{+\perp\}. \end{aligned}$$

A defeasible positive link from a property node p to a property node q is translated as follows:

$$q(X) : \{+\mathbf{p}\} \leftarrow p(X) : \{+\perp\}.$$

Finally, a defeasible negative link from p to q is represented as:

$$q(X) : \{-\mathbf{p}\} \leftarrow p(X) : \{+\perp\}.$$

5 Discussion

In this section we discuss a number of limitations of our approach and briefly compare its behaviour vis a vis other skeptical inheritance theories. First, we note that the translation introduced in the previous section works well for networks whose negative links go into terminal nodes only, i.e., nodes with the out-degree 0. For more general networks—as the following examples show—constructing a suitable semi-lattice may involve global analysis of the network to determine the so called “specificity” relation. In fact, it is possible to integrate the following notion of “local specificity” with the inheritance computation [25]:⁶ If the fact $p(q)$ holds (that is, q inherits p) and some individual i inherits property r through p and property $\neg p$ through q , then the priority constant q is more informative than the priority constant p as far as arbitrating this conflicting multiple inheritance is concerned.

Apart from showing the non-locality of our translation, the next two examples suggest that it is unlikely that there is a modular translation from the language of inheritance networks into our formalism. For instance, Figure 6(b) is obtained from Figure 6(a) by adding a negative link that directly connects nodes p and r . In Figure 6(a), since p is-a q and q is-a r , the property p is more specific than r as far as the node a is concerned. Hence, the node a inherits $\neg s$ from p and does not inherit s from r . In contrast, in Figure 6(b), p is-not-a r and hence p is not a more specific property of a than r is, and now there is an ambiguity regarding whether or not a should inherit s . Translating the network of Figure 6(b) into an evidence-based inheritance specification according to the algorithm of Section 4, we see that the above change in the specificity relationship between p and r with respect to a results in a change of priority constants in the heads of the rules corresponding to the in-links of s . This means that, a nonmonotonic change in the specificity relationship due to link-updates can potentially lead to scrapping and then recompiling a large chunk of the translation.

For another example, consider Figure 7(a). Here p is-a q and q is-a r and hence p is a more specific property than r as far as the node b is concerned. Since we can view the links from a and b to r as being redundant, both a and b inherit $\neg s$ from p . On the other hand, in Figure 7(b), a is-not-a q . So the link from a to r can be interpreted as an independent evidence that a is-a r . This gives rise to an ambiguity as to whether a is-a s . Indeed, it is

⁶A precise characterization of the relationship and the differences between the notions of specificity espoused in [39], [15] and [25] has been studied in [26], where it is shown that [39] theory is *nonlocal*, [15] is *ground local*, and the theories in [25] are *local*.

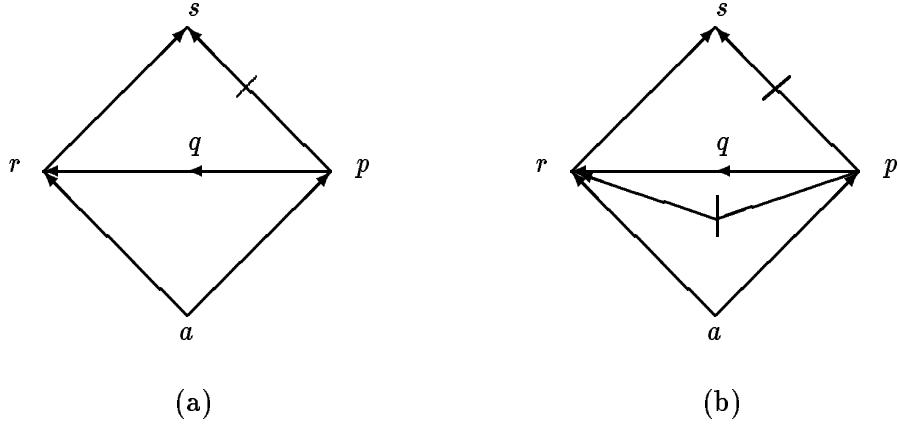


Figure 6: Link-update changes specificity relation

not clear how to disambiguate the conflicting pieces of evidence, one of which instructs us to inherit s through r while another suggests that inheritance of $\neg s$ through p should take place. Thus, even though a and b both possess properties p and r , p is more specific than r with respect to a , while p and r are incomparable in that respect relatively to b . Again, a change in the net has led to a non-modular change in the translation.

We conclude this section by comparing our approach with the so called “ideally skeptical” reasoner. An *ideally skeptical reasoner* believes only in those facts that are true in all *credulous* models [36] associated with any given inheritance specification. Our evidence-based theory is different in that respect, since there may be facts lying in the intersection of all credulous models and yet not supported in the unique intended model of Section 2.6. For instance, consider the following variant of an example from [13].

- $politically_motivated(X) : +1 \leftarrow pacifist(X) : +1.$
- $politically_motivated(X) : +1 \leftarrow pacifist(X) : -1.$
- $pacifist(X) : +1 \leftarrow quaker(X) : +1.$
- $pacifist(X) : -1 \leftarrow republican(X) : +1.$
- $quaker(penn) : +\omega.$
- $republican(bush) : +\omega.$
- $quaker(nixon) : +\omega.$
- $republican(nixon) : +\omega.$

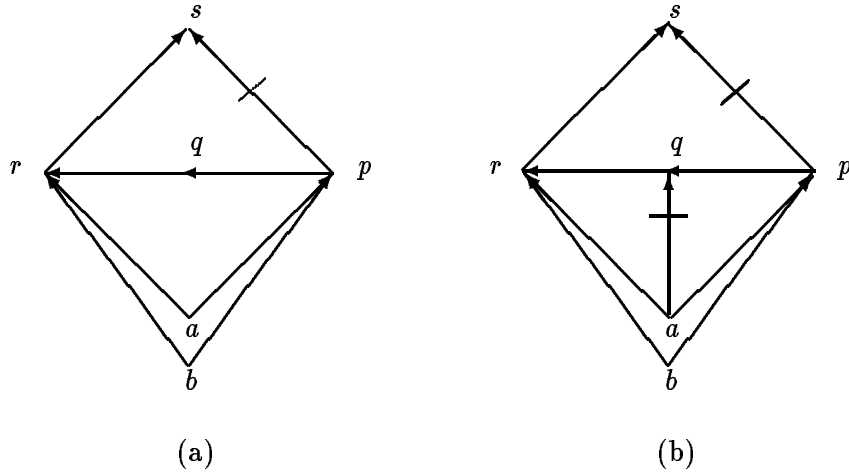


Figure 7: A more subtle change in specificity relation

The minimal supported model for this example satisfies $\textit{politically_motivated}(\textit{penn}) : +1$ and $\textit{politically_motivated}(\textit{bush}) : +1$, because it satisfies $\textit{pacifist}(\textit{penn}) : +1$ and $\textit{pacifist}(\textit{bush}) : -1$, respectively; the fact $\textit{politically_motivated}(\textit{nixon}) : +1$ is not satisfied, though. On the other hand, the latter fact belongs to all credulous models of the network corresponding to the above inheritance specification, for in every such model either $\textit{pacifist}(\textit{nixon})$ or $\neg\textit{pacifist}(\textit{nixon})$ holds true.

In this example, our theory agrees with [15] rather than [36]. To a certain extent, it is a matter of taste as to which theory is more natural. An ideal skeptic reasons in the intersection of all possible worlds and is willing to draw conclusions based on inconsistent beliefs. For instance, in the above, $\textit{pacifist}(\textit{nixon})$ is such a belief and yet an ideal skeptic would use it to conclude that Nixon is politically motivated. In contrast, we and [15] would hold back the judgement until the situation clears. In each case this happens for somewhat different reasons, but in both instances the law of excluded middle does not hold, even at the level of models. For instance, it is entirely possible for Nixon to be neither a pacifist nor a non-pacifist in a minimal supported model.

6 Conclusion and Future Work

We proposed a new logic framework for representing inheritance networks, which provides a convenient way of encoding an important fragment of common-sense reasoning. In particular, we can represent class-subclass hierarchies, mixed inheritance networks with strict and defeasible links, preferential inheritance networks, and other phenomena. The framework also supports inheritance through paths with negative links and skeptical reasoning.

An important property of our theory of inheritance is the *locality* of its semantics. This means that whether or not a specific object inherits a certain property depends only on the *neighboring* properties inherited by that object. In terms of our logic, q is a neighboring property of p , where p and q are predicate symbols, if and only if $q \prec p$ and there is no r such that $q \prec r \prec p$. Locality affects both conceptual and computational aspects of our theory [30, 34]. However, this also means that we cannot mimic *nonlocal* approaches, such as [15], closely enough.

One possible extension of our logic framework is to allow networks where inheritance conflicts are resolved using, so called, *credulous* reasoning [40]. This can be achieved by weakening our constraints on the domain of priority constants and, instead of insisting on the existence of unique least upper bounds, let every set of priority constants to have several minimal upper bounds. Another extension that was briefly mentioned in Section 2.2 is to allow the “depends-on” relation to be cyclic on predicate symbols. This will let us incorporate cyclic inheritance networks and contrapositive reasoning⁷ [28, 31]. However, extending the class of inheritance specifications to locally stratified programs may not be enough for handling contraposition, and an adaptation of the well-founded semantics [42] may be needed.

References

- [1] K. R. Apt, H. A. Blair and A. Walker, Towards a theory of declarative knowledge, in: Jack Minker (ed.), *Foundations of Deductive Databases and Logic Programming* (Morgan Kaufman, 1987) 89–148.
- [2] F. Bacchus, A modest, but semantically well-founded, inheritance reasoner, in: *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (1989) 1104–1109.
- [3] N. Belnap, How a computer should think, in: G. Ryle (ed.), *Contemporary Aspects of Philosophy* (Oriel Press, 1977) 30–56.
- [4] N. Belnap, A useful four-valued logic, in: J. Dunn and G. Epstein (ed.), *Modern uses of multi-valued logic* (D. Reidel, 1977) 8–37.
- [5] H. A. Blair and V. S. Subrahmanian, Paraconsistent logic programming, in: *Lecture Notes in Computer Science* **287** (1987) 340–360.
- [6] H. A. Blair and V.S. Subrahmanian, Paraconsistent logic programming, in: *Theoretical Computer Science* **68** (1989) 135–154.
- [7] A. Borgida, Modeling Class hierarchies with contradictions, in: *Proceedings of SIGMOD’88 Conference* (1988) 434–443.

⁷This would account for the following kind of derivation. Let p is-a q and r is-not-a q are both strict links. Then r is-not-a p .

- [8] J. P. Delgrande, An approach to default reasoning based on a first-order conditional logic: revised report, in: *Artificial Intelligence* **36** (1988) 63–90.
- [9] D. Etherington and R. Reiter, On inheritance hierarchies with exceptions, in: *Proceedings of the Second National Conference on Artificial Intelligence* (1983) 104–108.
- [10] M. Fitting, Bilattices and the semantics of logic programming, in: *Journal of Logic Programming*, 1991.
- [11] H. Geffner and J. Pearl, A framework for reasoning with defaults, in: *Proceedings of Society for Exact Philosophy Conference* (1988).
- [12] M. L. Ginsberg, Multi-valued logics: A Uniform Approach to Reasoning in Artificial Intelligence, in: *Computational Intelligence* **4** (1988) 265–316.
- [13] M. L. Ginsberg, A local formalization of inheritance (1988) Unpublished Manuscript.
- [14] B. Haugh, Tractable theories of multiple defeasible inheritance in ordinary nonmonotonic logics, in: *Proceedings of the Seventh National Conference on Artificial Intelligence* (1988) 421–426.
- [15] J. Horty, R. Thomason and D. Touretzky, A skeptical theory of inheritance in nonmonotonic semantic networks, in: *Proceedings of the Sixth National Conference on Artificial Intelligence* (1987) 358–363.
- [16] J. Horty, R. Thomason, and D. Touretzky, Mixing strict and defeasible inheritance, in: *Proceedings of the Seventh National Conference on Artificial Intelligence* (1988) 427–432.
- [17] M. Kifer, and A. Li, On the semantics of Rule based expert systems with uncertainties, in: *Proceedings of the Second International Conference on Database Theory* Springer Verlag LNCS **326**, Bruges, Belgium (1988) 102–117.
- [18] M. Kifer, and E. L. Lozinskii, RI: A logic for reasoning with inconsistency, in: *Proceedings of the Fourth IEEE Symposium on Logics in Computer Science* Asilomar, CA (1989) 253–262.
- [19] M. Kifer and E.L. Lozinskii, A logic for reasoning with inconsistency, in: *Journal of Automated Reasoning*, Kluwer Academic Publ. (1992). to appear.
- [20] M. Kifer and V.S. Subrahmanian, On the expressive power of annotated logic programs, in: *Proceedings of North American Conference on Logic Programming*, Cleveland, Ohio, MIT Press (1989) 1069–1089.
- [21] M. Kifer and V.S. Subrahmanian, Theory of generalized annotated logic programming and its applications, in: *Journal of Logic Programming* (1992). to appear.

- [22] T. Krishnaprasad, M. Kifer, and D.S. Warren, On the circumscriptive semantics of inheritance networks, in: Z. Ras and M. Zemankova (eds.), *Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems* North Holland (1989) 448–457.
- [23] T. Krishnaprasad and M. Kifer, An evidence-based framework for a theory of inheritance, in: *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (1989) 1093–1098.
- [24] T. Krishnaprasad, M. Kifer, and D.S. Warren, On the declarative semantics of inheritance networks, in: *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (1989) 1099–1103.
- [25] T. Krishnaprasad, The Semantics of Inheritance Networks, *Ph.D. Dissertation*, State University of New York at Stony Brook (1989).
- [26] Krishnaprasad Thirunarayan, Locality in Inheritance Networks, *Technical Report #09/91*, Wright State University, Dayton (1991)
- [27] V. Lifschitz, On the Declarative Semantics of Logic Programs with Negation, J. Minker (ed), in: *Foundations of Deductive Databases and Logic Programming*, (Morgan-Kaufmann, Los Altos, CA, 1988) 177–192.
- [28] L. Padgham, A model and representation for type information and its use in reasoning with defaults, in: *Proceedings of the Seventh National Conference on Artificial Intelligence* (1988) 409–414.
- [29] L. Padgham, Negative reasoning using inheritance, in: *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (1989) 1086–1092.
- [30] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, (Morgan Kaufman, San Mateo, 1988).
- [31] D. Poole, A logical framework for default reasoning, in: *Artificial Intelligence* **36** (1988) 27–47.
- [32] T.C. Przymusiński, On the declarative semantics of deductive databases and logic programs, in: J. Minker (ed), *Foundations of Deductive Databases and Logic Programming*, (Morgan Kaufmann, Los Altos, CA, 1988) 193–216.
- [33] H. Przymusińska and M. Gelfond, Inheritance hierarchies and autoepistemic logic, in: Z. Ras and M. Zemankova (eds.), *Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems*, (North Holland, 1989) 419–429.
- [34] B. Selman and H.J. Levesque, The tractability of path-based inheritance, in: *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (1989) 1140–1145.

- [35] L. Shastri, Default reasoning in semantic networks : a formalization of recognition and inheritance, in: *Artificial Intelligence* **39** (1989) 283–355.
- [36] L. A. Stein, Skeptical inheritance: Computing the intersection of credulous extensions, in: *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (1989) 1153–1158.
- [37] R. Thomason, J. Horty, and D. Touretzky, A calculus for inheritance in monotonic semantic nets, in: Z. Ras and M. Zemankova (eds.), *Proceedings of the Second International Symposium on Methodologies for Intelligent Systems* (North Holland, 1987) 280–287.
- [38] R. Thomason and J. Horty, Logics for inheritance theory, in: M. Reinfrank, J. de Kleer, M. Ginsberg, and E. Sandewall (eds.), *Non-Monotonic Reasoning*, Springer-Verlag (1989).
- [39] D. Touretzky, *The Mathematics of Inheritance Systems* (Morgan Kaufmann, Los Altos, 1986).
- [40] D. Touretzky, J. Horty, and R. Thomason, A clash of intuitions: the current state of nonmonotonic multiple inheritance systems, in: *Proceedings of the Tenth International Conference on Artificial Intelligence* (1987) 476–482.
- [41] M.Y. Vardi, The complexity of relational query languages, in: *Proceedings of 14th ACM Symposium on Theory of Computation* (1982) 137–146.
- [42] A. Van Gelder, K.A. Ross and J.S. Schlipf, The Well-Founded Semantics for General Logic Programs, *Proc. of ACM SIGMOD-SIGACT Symp. on Principles of Database Systems* (1988) 221–230.
- [43] J.D. Ullman, *Principles of Database and Knowledge-Base Systems* (Computer Science Press, Rockville, MD, 1988).