DEFAULT REASONING AS LIKELIHOOD REASONING

Elaine Rich

Department of Computer Sciences
The University of Texas at Austin

Abstract

Several attempts to define formal logics for some type of default reasoning have been made. All of these logics share the property that in any given state, a proposition p is either held to be true, it is held to be false, or no belief about it is held. But if we ask what default reasoning really is, we see that it is form of likelihood reasoning. The goal of this paper is to show that if default reasoning is treated as likelihood reasoning (similar to that of Mycin), then natural solutions emerge for several of the problems that are encountered when default reasoning is used. This is shown by presenting 7 such problems and showing how they are solved.

I. Introduction

The need for default reasoning in artificial intelligence systems is now well understood [9]. Such reasoning is required to enable systems to deal effectively with incomplete information. Several attempts have been made in the literature to define a formal logic for some type of default reasoning [5, 4, 3]. All of these logics share the property that in any given state, a proposition p is either held to be true, it is held to be false, or no belief about it is held. No intermediate situation is possible. But if we ask what default reasoning really is, we see that it is a form of likelihood reasoning. Most birds fly, so it is likely that a particular bird does so. Most people have the same hometown as their spouse, so it is likely that Mary does. In many domains, negative facts about which we have no information are true, so if we do not know whether there is a flight from Vancouver to Oshkosh, it is likely that there is not.

Once we recognize that default reasoning is a form of likelihood reasoning, it is clear that there are more than two truth values that can be associated with a proposition. Different truth values represent differing levels of belief (or confidences in) the truth of the associated proposition. This corresponds to the situation in a variety of expert systems such as Mycin [8]. We will adopt the likelihood reasoning system used in Mycin. If a proposition is known definitively to be true, then it is held to be true with a certainty factor (CF) of 1. If it is known definitively to be false, it is known with a CF of -1. If there is equal evidence for and against a proposition (including the situation in which there is no evidence either way) then the proposition has an associated CF of 0. A CF in between these values indicates that the evidence either supports or disconfirms the associated proposition but it does so with some doubt.

The goal of this paper is to show that if default reasoning is treated as likelihood reasoning, then natural solutions emerge for several of the problems that are encountered when default reasoning is used. In the rest of this paper, we will present 7 such problems and show how they can be solved through the use of certainty factors.

II. Definitions

Before doing this, however, we must introduce some notation and define a likelihood reasoning system. A standard (nonlikelihood) default rule consists of three parts: the premises, an UNLESS clause [7], and a conclusion. For example, the default rule "Most birds fly", can be represented as:[1]

$$\text{bird}(x) \Rightarrow \text{fly}(x) \text{ UNLESS } \sim\text{fly}(x) \qquad (1)$$

In "normal" default rules, the UNLESS clause is exactly the negation of the conclusion. We will restrict our attention to normal default rules for two reasons. They are adequate to describe naturally occurring defaults. And they form a more tractable logical and computational system than do nonnormal rules [6]. Given this assumption, UNLESS clauses need not explicitly be represented. All we need do is to distinguish a default rule from a standard rule. We can do that by using * as an additional premise. Thus the default rule above will be represented as

$$\text{bird}(x) \land * \Rightarrow \text{fly}(x) \qquad (2)$$

which can be read as, "If bird(x) and not fly(x) is not provable, then conclude fly(x)".

Now we extend this representation to associate with each default rule a certaintly factor (CF), which states the confidence that the system should have in the conclusion if each of the premises is known (with CF=1). So the example rule becomes

$$\text{bird}(x) \land * \Rightarrow \text{fly}(x) \qquad (\text{CF}{=}.9) \qquad (3)$$

If bird(Tweety) is known with a CF of 1, then, using this rule, fly(Tweety) can be asserted with CF=.9. Notice that the meaning of a CF attached to a universally quantified formula represents the certainty that the statement holds for any particular assignment of values to the bound variables. The certainty that the statement holds for all such bindings is usually -1, since the existence of at least one counterexample is generally known.

If the CF of bird(Tweety) is less than 1, that uncertainty must contribute to the final CF of fly(Tweety). This is done using the formula:

$$\text{Given an implication } P_1 \land P_2 \land \ldots \land P_n \Rightarrow C \quad (\text{CF}{=}k) \qquad (4)$$
$$\text{Then the CF attached to C will be } k \times \prod \text{CF}(P_i)$$

So if bird(Tweety) has a CF of .9, then fly(Tweety) will be asserted with a CF of $.9 \times .9 = .8$.

In any likelihood theory, contradictions will arise. There are two ways that contradictions can be handled. One is to assert the more likely proposition and to modify its CF to account for the negative evidence (by subtracting it out as is done in Mycin). The other is to exploit other information about the reasoning process to resolve the conflict and to compute an appropriate CF for the resulting assertion. As we will see, there are times when each of these is appropriate.

III. Statistical, Protypical, and Definitional Facts

The distinction between definitional facts and prototypical ones is often made (e.g., [10]). An additional distinction is sometimes

[1]Throughout this paper, all variables are assumed to be universally quantified so the quantifiers will be omitted.

made [6] between prototypical and statistical facts. Consider the facts represented by the following sentences:

People are mammals. (5)
  person(x) ⇒ mammal(x)

Most people can see. (6)
  person(x) ∧ * ⇒ sees(x)

Most Southerners are Democrats. (7)
  Southerner(x) ∧ * ⇒ Democrat(x)

(5) is a definition. It can never fail to hold. (6) is a prototypical statement. It should be assumed to hold in the absence of evidence to the contrary. (7) is a statistical statement. It, too, is a reasonable default. But it seems somehow less necessarily true than (6).

The distinction among these three kinds of facts can easily be captured using likelihoods. Definitional facts have a CF of exactly 1. So do analytical facts, which can be derived from a set of definitional facts. All other facts represent empirical observations, very few of which have no exceptions. Prototypical facts typically have fewer exceptions than do statistical ones; thus they tend to have higher CFs. Any fact with a CF of other than 1 or -1 is a default rule. It suggests something to be believed unless there is contradictory evidence. (5)-(7) can be represented in a likelihood framework as follows:

person(x) ⇒ mammal(x)	(CF=1)	(8)
person(x) ⇒ sees(x)	(CF=.99)	(9)
Southerner(x) ⇒ Democrat(x)	(CF=.8)	(10)

By representing CFs that differ from 1, we not only eliminate distinctions that are very difficult to make, but also allow more information about assertions to be represented in the knowledge base.

Notice that we can now eliminate the explicit * part of the premise for default rules since any rule with a CF other than 1 or -1 is known to be a default rule. The implicit UNLESS clause still plays an important role though. Suppose we know

∼sees(John)	(CF=1)	(11)

Since the truth of this assertion is certain, it will completely dominate all inferences derived from default rules such as (9). This corresponds to the role of the UNLESS clause in standard default logic. On the other hand, if (11) were believed with a CF of only .9, (9) would produce the assertion

sees(John)	(CF=.99)	(12)

Now two conflicting facts are believed and a conflict resolution mechanism must be applied to choose between them or perhaps to decide that no conclusion is justified. In standard default logic, the only conflict resolution technique available is time sequencing. The first conclusion to be drawn blocks later conflicting ones, regardless of the relative strengths of each of the conclusions. Using CFs instead, however, allows more rational mechanisms to be employed. See Section VII. for a further discussion of this.

IV. The Role of Multiple Premises

Multiple premises can be used to increase the CF of a rule. Consider the rule

animal(x) ⇒ defensemechanism(x,camoflage)	(CF=.4)	(13)

which can be modified to have an increased CF by the addition of 3 premises:

animal(x) ∧ color(x,c) ∧ habitation(x,y) ∧ color(y,c)
  ⇒ defensemechanism(x,camoflage) (CF=.9) (14)

Whereas (13) would never appear in a knowledge base without CFs, (14) might. If there are no CFs represented, then there must be a threshold CF value (not normally explicit or consistent) below which statements must simply be thrown out by the knowledge base creator, or the statements must be refined with additional premises until their CFs cross the threshold. By representing CFs explicitly, though, the need for an arbitrary threshold disappears.

V. Cascading Beliefs

The explicit use of CFs not only increases the information that is directly available from a knowledge base; it also increases the information associated with assertions derived in multiple steps. Consider the following:

adult(x) ⇒ employed(x)	(CF=.9)	(15)
adult(x) ⇒ ∼dropout(x)	(CF=.91)	(16)
adult(x) ∧ ∼dropout(x) ⇒ employed(x)	(CF=.99)	(17)

In (17) an additional premise has been added to refine the domain of the assertion and thus to raise its CF.

Suppose we know adult(Bill). Then, using (15), we can conclude employed(Bill) with CF=.9. We can also derive the same conclusion using (16) and (17) (by multiplying the two CFs as described in Section II.). If, on the other hand, we also know, with CF=1, ∼dropout(Bill), then, using (17), we can conclude employed(Bill) with a CF of .99.

VI. Applying the Closed World Assumption

Examples like the last one arise often if the closed world assumption is used. Consider again the last example, but suppose (16) were missing and we had no knowledge of whether or not Bill was a dropout. Then we might still like to apply (17) to conclude that Bill is employed. This can be done by using the closed world assumption to assert ∼dropout(Bill). In some restricted domains this can be done without decreasing the CF of the final assertion. In an airline database system, if flightfrom(city1,city2) is not known, then with CF=1 there is not such a flight. But in many other domains, that is not the case. In such domains, the use of the closed world assumption must affect the CF of the resulting assertion. The more knowledge that is available, the more accurately this can be done. A simple thing is to assign as the CF of the assertion ∼p the quantity (1-prob(p)). A more accurate answer can be obtained if interactions between p and other relevent predicates are known. This is, in fact, what we did by inserting (16). In addition, other sources of information can be used if they are available. For example, the absence of an assertion p that would be important if it were true more strongly suggests ∼p than does the absence of an unimportant assertion [1].

VII. Conflicting Beliefs

In many knowledge bases that contain default rules, contradictory inferences may arise through the use of more than one default rule. Consider the following example from [6]:

Typically Republicans are not pacifists.
  Republican(x) ⇒ ∼pacifist(x) (18)

Typically Quakers are pacifists.
  Quaker(x) ⇒ pacifist(x) (19)

Bill is a Republican and a Quaker.
  Republican(Bill) ∧ Quaker(Bill) (20)

There are two ways in which a nonlikelihood logic might deal with this situation. One is that described in [5] in which a default knowledge base may have one or more extensions. Each extension is a consistent set of assertions, but the set of extensions of a given knowledge base may not be consistent. In this example, there are two extensions, one containing the assertion that Bill is a pacifist and one containing the assertion that he is not. A reasoning system built around such a logic would produce one or the other extension nondeterministically.

The other nonlikelihood approach is that described in [6], in which the default rules in the knowledge base are modified to handle the interactions between them and to guarantee that, in the case of conflict, no conclusion at all will be drawn. Thus, in place of statements (18) and (19) above we must write:

Most non-Quaker Republicans are not pacifists.
Republican(x) ∧ ∼Quaker(x) ⇒ ∼pacifist(x) (21)

Most non-Republican Quakers are pacifists.
Quaker(x) ∧ ∼Republican(x) ⇒ pacifist(x) (22)

Since the hypotheses of neither of these assertions are satisfied in the case of Bill, no inference will be drawn as to whether Bill is a pacifist.

The major drawback to this approach is that, for a large knowledge base, there may be many such rule interactions and the rule writer is forced to find those interactions and to modify each rule so that it mentions explicitly all of the other rules with which it conflicts. It would be more accurate if the reasoning system itself were able to detect these interactions and to decide whether or not a particular conclusion is warranted.

A likelihood reasoning scheme can both guarantee a determinsitic result and detect rule interactions automatically. (18)-(20) are rewritten as

Republican(x) ⇒ ∼pacifist(x) (CF=.8) (23)

Quaker(x) ⇒ pacifist(x) (CF=.99) (24)

Now we can assert

pacifist(Bill) (CF=.19) (25)

by simply accepting the more likely belief but modifying its CF to account for the conflicting evidence. We do this by subtracting the CF of the conflicting assertion from the CF of the assertion that is chosen.

This approach does not remove all difficulty from this problem. In particular, a fine-tuning of the CFs in the knowledge base may affect the outcome of the reasoning process. But, given a particular knowledge base, the outcome of the reasoning process can be predicted by a static analysis of the knowledge base itself. The reasoning process is not subject to the race conditions that are inherent in the multiple extension approach. Additionally, it is possible to introduce more sophisticated techniques for conflict resolution if that is important.

VIII. Property Inheritance Across ISA Links

A common form of default reasoning is the inheritance of properties in the ISA hierarchies of semantic networks. The reasoning rule that is used is not usually stated as a rule but rather is contained in the code of the network interpreter. This rule is that a node n1 can inherit any property p that is true of any other node n2 above n1 in an ISA chain unless a contradiction is found at some node that is in the chain and that is below n2 and not below n1. When the knowledge in the network is represented instead in a logical framework, it is more difficult to define a corresponding rule. But the rules of likelihood reasoning we have described can solve this problem easily, with the additional advantage that a confidence measure can be attached to the resulting inference.

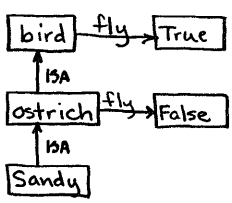

Figure 1: A Simple Semantic Network

Consider the network shown in Figure 1. Using a traditional network inference scheme, the question of whether Sandy can fly would be answered by chaining up the ISA hierarchy starting at the node labeled Sandy and stopping at the first node from which a FLY link emerges.

This same knowledge can be represented as the following set of likelihood formulae:

bird(x) ⇒ fly(x) (CF=.9) (26)

ostrich(x) ⇒ bird(x) (CF=1) (27)

ostrich(x) ⇒ ∼fly(x) (CF=1) (28)

ostrich(Sandy) (CF=1) (29)

Applying the rules of likelihood reasoning to (26)-(29), we can derive

fly(Sandy) (CF=.9) (30)

∼fly(Sandy) (CF=1) (31)

A contradictory set of beliefs has been derived. Just as in the last section, we must choose the belief with the higher CF value. But this time, rather than attaching to that belief a CF that reflects the conflict, we attach the higher value. This is done by examining the reason for the conflict, observing that the rule about ostriches is a special case of the rule about birds and thus should override the latter, and then concluding ∼fly(Sandy) with the CF suggested by the special case rule. This provides an example of the process of reasoning to resolve conflicts that was suggested in Section II..

To guarantee that likelihood reasoning will produce the same result that would be produced by chaining through an ISA hierarchy, it is necessary that what we call the *monotonic consistency constraint* (MCC) hold for the CFs in the system. MCC is defined as follows:

> If there is a rule p1(x) ⇒ p2(x) (CF=k1) and a rule p3(x) ⇒ p4(x) (CF=k2) and p3(x) ⇒ p1(x) by an ISA chain and (p2(x) ∧ p4(x)) is a contradiction, then k2>k1.

In other words, properties attached to higher nodes in an ISA chain must have lower CFs than all conflicting properties lower in the chain. This is always possible. For all ISA hierarchies, there exists an assignment of CFs that satisfies MCC. In fact, any assignment of CF values that forms a partial ordering that is isomorphic to the partial ordering formed by the ISA hierarchy is adequate. MCC is consistent with the intuition that more specific rules are more accurate than more general ones.

IX. The Failure of the Shortest Path Rule

The property inheritance problem is more complex in tangled hierarchies [2] such as the one shown in Figure 2(*a*). One way of implementing inheritance in such a network is to use the shortest path rule, but as pointed out in [6], this rule does not always work. The rule is sensitive to the insertion and deletion of nodes that should have no effect on inheritance. This is illustrated in Figure 2(*b*). The problem is that the shortest path rule counts steps in the reasoning chain as an approximation to a certainty level. But this is often a poor approximation.

Likelihood reasoning solves this problem by using certainty factors directly. The use of a rule with a CF of 1 has no effect on the CF of the resulting assertion. The use of a rule with some other CF does. The knowledge of Figure 2(*a*) is represented as:

bird(x) ⇒ ∼tame(x) (CF=.8) (32)

ostrich(x) ⇒ bird(x) (CF=1) (33)

pet(x) ⇒ tame(x) (CF=.95) (34)

ostrich(Sandy) (CF=1) (35)

pet(Sandy) (CF=1) (36)

The knowledge of Figure 2(*b*) is represented as

bird(x) ⇒ ∼tame(x) (CF=.8) (37)

pet(x) ⇒ tame(x) (CF=.95) (38)

rarepet(x) ⇒ pet(x) (CF=1) (39)

rarepet(Sandy) (CF=1) (40)

bird(Sandy) (CF=1) (41)

From both (32) - (36) and (37) - (41) we derive:

~tame(Sandy) (CF=.8) (42)

tame(Sandy) (CF=.95) (43)

We assert (43) since it has the higher CF. The insertion and deletion of rules with CFs of 1 did not affect the derivation.

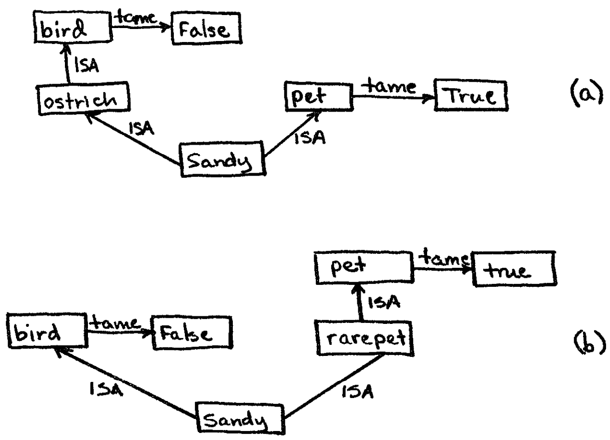

Figure 2: A Tangled Network

X. Conclusion

Default reasoning is likelihood reasoning and treating it that way simplifies it. This is important in problem-solving systems, which must manipulate their knowledge, incomplete as it may be, to construct the best possible solution to a problem. Thus, for example, it is important to be able to choose to believe the most likely of a set of conflicting assertions rather than simply to believe nothing. There are, of course, other domains in which this is not true. It may be desirable for a database system, for example, to assert nothing in such a situation and to allow the user of the database to decide what to do next.

It is important also to keep in mind that the introduction of likelihoods poses new questions such as how best to assign CFs when there are conflicts. But these questions are central to the process of default reasoning and should be addressed.

References

1. Collins, A, E. H. Warnock, N. Aiello & M. Miller. Reasoning from Incomplete Knowledge. In *Representation and Understanding*, D. G. Bobrow & A. Collins, Eds., Academic Press, New York, 1975.

2. Fahlman, S. E.. *NETL: A System for Representing and Using Real-World Knowledge.* MIT Press, Cambridge, Mass., 1979.

3. McCarthy, J. "Circumscription-A Form of Non-Monotonic Reasoning." *Artificial Intelligence 13* (Apr. 1980).

4. McDermott, D. & J. Doyle. "Non-Monotonic Logic I." *Artificial Intelligence 13* (Apr. 1980).

5. Reiter, R. "A Logic for Default Reasoning." *Artificial Intelligence 13* (Apr. 1980).

6. Reiter, R. & G. Criscuolo. Some Representational Issues in Default Reasoning. Dept. of Compute Science, The University of British Columbia, 1980.

7. Sandewall, E. An Approach to the Frame Problem and its Implementation. In *Machine Intelligence 7*, B. Meltzer & D. Michie, Eds., Univ. of Edinburgh Press, 1972.

8. Shortliffe, E. H.. *Computer-based Medical Consultations: MYCIN.* Elsevier, New York, 1976.

9. Winograd, T. "Extended Inference Modes in Reasoning by Computer Systems." *Artificial Intelligence 13* (1980).

10. Woods, W. A. What's in a Link: Foundations for Semantic Networks. In *Representation and Understanding*, D. G. Bobrow & A. Collins, Eds., Academic Press, New York, 1975.