Pergamon

Information Processing & Management, Vol. 34, No. 1, pp. 35–41, 1998

0306-4573/98 $19+0.00

PII: S0306-4573(97)00036-8

NONMONOTONIC LOGIC FOR USE IN INFORMATION RETRIEVAL: AN EXPLORATORY PAPER

C. D. HURT

Professor and Director, School of Information Resources & Library Science, University of Arizona, 1515 East First Street, Tucson, Arizona, 85719 USA

(Received 27 March 1995; accepted 11 October 1995; revised 27 September 1996)

Abstract—Monotonic logic requires reexamination of the entirety of a logic string when there is a contradiction. Nonmonotonic logic allows the user to withdraw conclusions in the face of contradiction without harm to the logic string. This attribute has considerable application to the field of information searching. Artificial intelligence models and neural networks based on nonmonotonic logic have the potential for more robust findings than the use of monotonic logic alone. This paper demonstrates the power of nonmonotonic logic but does not implement the logic.

1. INTRODUCTION

Traditional information retrieval techniques assume absence of contradiction. Other ways of expressing this concept are that:

1. there is always an answer and it is somehow discrete in the universe of 'right' answers;
2. the truth will always surface and it will be recognizable as the truth.

Classic information retrieval techniques, as a necessity, often rely on a standardized first-order logic. First-order logic assumes that the information and the request for information can be defined without contradiction. A classic example of the difficulty of this position is the 'bird problem' in artificial intelligence.

1. Birds fly
2. Kiwis are birds
3. Therefore, Kiwis fly.

In standard cases, other than kiwis, the rule holds. It is the exceptions that are interesting here—penguins, birds with broken wings, and like exceptions. The problem is identified only in the more specific cases. Information professionals are faced with much of the same problem. Generally we will accept the rule that birds do fly as quickly as will those who seek our services. In seeking our services, however, the rules of the game may be changed. It is often as a direct result of the inability to rectify the veracity of general rules that information professionals are consulted. This is, in a way, higher order information service. Those who seek the services of an information professional may have already exhausted the 'normal' or the 'standard' reference sources. They are now in need of non-standard thinking. This is the utility of the bird problem that will be used throughout this paper. It is one means to force the issue of non-standard thinking. In the case of the bird problem, universal acceptance of the first-order is not possible but it is probable. The exceptions to this and the majority of other rules begin to multiply when faced with the real world. As a generalization, it is the real world that concerns information professionals and their clients.

The problem for information retrieval is much like the bird problem. The more general or non-exceptional query is not the norm. The kiwi problem is more symptomatic of the

To whom all correspondence should be addressed. Telephone: 520.621.3566. Fax: 520.621.3279. Email: cdh@u.arizona.edu

information retrieval question.

There are several solutions to the first-order logic problem for information retrieval.

1. The first solution is to enumerate the exceptions. This is the classic means of handling the problem and traditional methods were developed to do just this. An example of an enumeration technique is the *see* references in thesaurus or subject heading lists.
2. The second method is to retain as many original categories as possible but to group exceptions into a largely separate category. To some extent the Z class in the Library of Congress classification system can be seen as an exceptional class within the argument just developed. An obvious solution to the kiwi problem is to have a category of non-flying birds.
3. A third solution is to give up the assumption of first-order logic that information retrieval systems must be monotonic. (First-order logic is monotonic in the sense that it makes assumptions about the set being described in terms of completeness.) The exceptions to the set can rarely be overcome and there is always the danger of non-inclusion.

Continuing the example of the kiwi problem when the category of non-flying birds is added, do birds with injured wings count as flying birds or as non-flying birds? Depending on the state of the injury and the time factors involved,the injured bird can fall into both categories. Note that the categories are still crisp, it is the bird or the data element that is changeable.

The monotonic nature of first-order logic suggests that a different approach could be used. Specifically, the problem of data elements being allowed membership in different sets depending on additional characteristics presents information retrieval problems. One strategy worth pursuing is, instead of introducing new kinds of exceptions or predicates to qualify statements, expunge the requirement for monotonicity.

In the bird problem, we can all agree that most birds fly. Assume that the statement is made that, 'Charlie is a bird'. The assumption is that Charlie is a bird and can fly as a result of being classed as a bird. If we are then told that Charlie is a kiwi, we must go back and revise our belief in Charlie's ability to fly.

The outcome is that unless the world is standardized, allowing exceptions to the general case will result in a diminution of the veracity of the general case. At some point, following sufficient numbers of exceptions, questions concerning the overall validity and veracity of the general case will arise. From a logic standpoint, these questions often arise prior to the more practical threshold in general society. The argument here is that if a more rigorous and utilitarian logic is applied then the possibility of reaching the practical threshold is also reduced. The practical significance of this in information retrieval is that, by use of a set of tools such as nonmonotonic logic, the logical and practical validity and veracity of the information retrieved is more easily retained.

Nonmonotonic logic is not a new subject. McDermott & Doyle (1980, 1982) and Reiter (1980) are good starting points for the discussion. The development of nonmonotonic logic as a means to address information retrieval problems is underdeveloped, however. Searching the general literature dealing with information retrieval for either theoretical papers or application papers produced nothing. The goal of this paper is to suggest a means of handling exceptions to universal statements in a way that will allow information professionals greater latitude and a higher success rate.

There is a parallel to the information retrieval literature that is germane to the discussion here. Nonmonotonic logic is used with good effect in the database literature. While a full discussion of the uses of nonmonotonic logic in the database arena is beyond the scope of this paper, the short discussion below will give the reader a flavor of the utility of nonmonotonic logic in another venue.

Antoniou (1996) presents a form of nonmonotonic logic, default reasoning and variants, as one means of developing systems for database searching. The approach is interesting because it addresses one of the major criticisms of nonmonotonic logic systems—the tendency to present them as fixed-point equations. A novel approach to the storage and maintenance of knowledge generated in mining databases is presented by Narayanan (1996). He suggests using nonmonotonic inheritance networks for the storage and maintenance of knowledge discovered in databases. The added value of Narayanan's approach is that use of such a technique would allow for conflicting information to be added without full or partial regeneration of the rules set for tree-based expert systems that are normally employed for knowledge storage.

Attempting to solve the fixed-point presentation problem for nonmonotonic logic, Subrahmanian *et al.* (1995) addressed the operational aspects of the semantics. They developed an efficient and stable strategy to model deductive databases. They used an approach that computed the well-founded semantics and then used an intelligent branch and bound strategy to compute the stable models. Their work effectively moves nonmonotonic logic from a semantics concern to an operational level.

In a parallel research stream, Beringer & Schaub (1993) present a version of and a discussion of nonmonotonic logic when they describe the work they did in working with Lifschitz's MBNF (minimal belief and negation as failure)logic. They developed a feasible subsystem of MBNF which can be translated into a logic built on first order logic and negation as failure, called FONF.

One of the common techniques used in nonmonotonic logic in data bases is closed world reasoning. It is also the technique used in this paper. Cadoli & Lenzerini (1994) present a detailed analysis of the computational complexity of the different forms of closed world reasoning for various fragments of propositional logic. This provides a picture of the boundary between tractability and intractability of forms of nonmonotonic reasoning.

In a counterpoint to the work done above, Darwiche & Pearl (1994) present an excellent synopsis of logical consistency in databases. Especially pertinent is the discussion of logical consistency and causal consistency in the case of nonmonotonic logic systems.

The use of nonmonotonic reasoning in data bases is not yet well used but it is well studied and the problems are generally circumscribed and understood on a theoretical basis. Uses of nonmonotonic logic beyond data bases is less well studied and the problems are less circumscribed. Because data bases structures and knowledge structures may play heavily in information retrieval as a general class of problems, the inclusion of material related to nonmonotonic logic in data bases is germane.

Assuming that the goal is to formulate a conditional statement in a way that allows consequents to be derived if and only if no contradictions can arise, we can use McDermott & Doyle's notation (McDermott & Doyle, 1980, 1982)

$\forall \times (\mathrm{Bird}(x) \wedge M\mathrm{Fly}(x) \rightarrow \mathrm{Fly}(x))$.

The expression *M*Fly (Charlie) can then be evaluated as: it is consistent with all we know about birds and Charlie that Fly(Charlie). Moving this into our example and using the notation from above: $\forall \times (\mathrm{Bird}(x) \wedge M\mathrm{Fly}(x) \rightarrow \mathrm{Fly}(x))$.Bird (Charlie)Fly (Charlie).

We can infer Fly(Charlie) provided we are also able to conclude that *M*Fly(Charlie). (The operator *M* is the addition of a rule of possibilization.) Possibilization allows us to infer *M*Fly(Charlie) provided we cannot infer $\neg$Fly(Charlie).

There are some immediate difficulties with this approach, however. Theory proving is not usually done by means of knowledge of what the provable theorems might be. The circularity and incestuous nature of the logic are the problem. McDermott & Doyle (1980) and McDermott & Doyle (1982) go far beyond the limits of this paper in demonstrating the feasibility of nonmonotonic logic. This paper does not determine the logical completeness or robustness of nonmonotonic logic. Rather, the goal is to demonstrate the efficacy and potential applicability of nonmonotonic logic in information retrieval. The desired result is completeness and soundness of the theoretical results. The difficulty is that most nonmonotonic logic systems require intensive computational techniques.

In the remainder of this paper, the use of nonmonotonic logic is suggested as a possible model for information retrieval using a semantic approach. For a finite subset of information, nonmonotonic logic completeness and soundness can be demonstrated.

2. DESCRIPTION

Continuing with the kiwi example from above, we can develop a logical subset such that

$$\Theta_0 = \left\{ \begin{array}{l} \textit{Birds Fly} \\ \textit{Kiwis do not fly} \\ \textit{Kiwis are birds} \end{array} \right\}.$$

If we were to use standard, first-order logic, the statement: $\Theta_0 \cup \{\textit{Charlie is a kiwi}\}$ would be undetermined with the knowledge that we have given in the prior statement and inconsistent with the inference that Charlie can fly. A statement of the sort: $\Theta_0 \cup \{\textit{Susan is a bird}\}$ would be entirely consistent with an inference that Susan can fly. The problem is to modify our language such that the 'Charlie' statement is consistent with our underlying theory and that, in the process, we do not disallow the 'Susan' statement.

Given our examples above, we can now extend and list the universe of facts with which we are working.

$$\theta_1 = \left\{ \begin{array}{l} Fly(x) \leftarrow Bird(x) \wedge M\ Fly(x) \\ \overline{Fly}(x) \leftarrow Kiwi(x) \\ Bird(x) \leftarrow Kiwi(x) \\ Kiwi(Charlie) \\ Bird(Susan) \end{array} \right\}.$$

If we can now assume that for the subset of the universe, Θ_1, there are only two constants, Charlie and Susan, we can now say:

$$\sum_{\Theta_1} = \{Charlie, Susan\}$$

where $\Sigma_{\Theta 1}$ is the Herbrand universe, Θ, of the example constants *Charlie* and *Susan*.

The base of facts from which we can work in developing our semantics is as follows.

Assuming that the theoretical set with which we will work is Θ, then the Herbrand base Π_Θ for $\Theta = \{A(t_1, \ldots, t_n) | A$ is an n-place predicate symbol mentioned in Θ and $\{t_1, \ldots, t_n\} \subseteq \Sigma_\Theta\}$. The Herbrand base for the example Θ_1 is then:

$$\prod_{\Theta 1} = \left\{ \begin{array}{l} Fly(Charlie) \\ Bird(Charlie) \\ \overline{Fly}(Charlie) \\ Kiwi(Charlie) \\ Fly(Susan) \\ Bird(Susan) \\ \overline{Fly}(Susan) \\ Kiwi(Susan) \end{array} \right\}$$

Our largest concern is with interpretation. (We wish to determine if our interpretation of a particular set of facts is a logical consequence of an underlying theory.) For the purposes of this paper, the set noted above is a Herbrand base for examining the underlying theory, Θ_1.

A significant part of the problem of interpretation is the assigning of different interpretations to the same set of constants and symbols. Generally formulas in predicate calculus cannot be reduced to a singular formula of propositional calculus. This is the age-old problem in information retrieval of understanding the requester's possible set of successful outcomes. Determining the closeness of the fit of the requester and the searcher is a measure of the 'truth' of the logic used in determining the response to the request. This veracity is usually understood through propositional calculus. The difficulty is in moving predicate calculus to propositional calculus.

Hebrand's theorem suggests that clauses can be reduced in a systematic manner to a finite set of propositional calculus formulas (Chang & Lee, 1973). A clause is then satisfiable if the countable set of propositional calculus formulas to which it is reduced is also satisfiable. Extending the argument, and following Herbrand, in order to test whether a clause is satisfiable, we can consider the variables of the clause. This assumes that the assignment of variables is accomplished with elements from the Herbrand universe. Herbrand interpretations differ only in so far as the assignment of predicate symbols. The assignment of constants and function symbols, is fixed, however.

Let Θ be a theory, and π a set of literals. Assuming the theory operator, $T_{\Theta,\pi}$ is continuous and monotonic (Tarski, 1955), we can develop an approximation which will allow determination of the failure set for theory Θ. If ω is the limit ordinal in the lattice, then $T_{\Theta,\pi\downarrow\omega}$ is the approximation for the point associated with the theory Θ and the set π. The failure set for Θ, $Fail_\Theta$, is

$$Fail_\Theta = \prod_\Theta - T_{\Theta,\pi\downarrow\omega}$$

The calculation of the nonmonotonic extension of the theory, Θ, is

$$T_{\Theta,0\downarrow_0} = \prod_{\Theta 1} = \left\{ \begin{array}{l} Fly(Charlie) \\ Bird(Charlie) \\ \overline{Fly}(Charlie) \\ Kiwi(Charlie) \\ \overline{Fly}(Susan) \\ Kiwi(Susan) \end{array} \right\},$$

$$T_{\Theta 1,0\downarrow_1} = \left\{ \begin{array}{l} Bird(Charlie) \\ \overline{Fly}(Charlie) \\ Kiwi(Charlie) \\ Bird(Susan) \\ \overline{Fly}(Susan) \end{array} \right\},$$

$$T_{\Theta 1,0\downarrow_2} = T_{\Theta 1,0\downarrow_\omega} = \left\{ \begin{array}{l} Bird(Charlie) \\ \overline{Fly}(Charlie) \\ Kiwi(Charlie) \\ Bird(Susan) \end{array} \right\}.$$

Thus

$$Fail_{\Theta 1} = \left\{ \begin{array}{l} Fly(Charlie) \\ Fly(Susan) \\ \overline{Fly}(Susan) \\ Kiwi(Susan) \end{array} \right\}.$$

If we then construct a set of nonmonotonic literals $Nm(\Theta)$ from the example Θ) where $Nm(\Theta) = \{MA\Lambda\Theta|MA$ is mentioned in Θ, $A\Theta \in \underline{\Pi_\Theta}$, and $\hat{A} \in fail\Theta\}$ then $\underline{\text{the}}$ literals to consider in Π_Θ are *MFly(Charlie)* and *MFly(Susan)*. $\overline{Fly}(Charlie) \notin Fail_\Theta$, but $\overline{Fly}(Susan) \in Fail_{\Theta'}$, so$Nm(\Theta) = \{MFly(Susan)\}$.

Defining an approximation is only one part of the puzzle. To be useful, the approximation must be able to determine logical consequences. In short, what we have so far is theory and to satisfy the intentions of the paper, some means of making this theory work in information retrieval is necessary. Information retrieval is most often concerned with clauses. Clauses in this context are defined as groupings of elements that, in themselves, are searchable. It is the elements in some combination that are the most interesting, however. Extending the arguments made already, we can investigate the ability of the system to handle clauses.

Given a set of clauses, Θ, and a goal G, the clauses and goal are unsatisfiable if and only if either

1. G contains the ground literal $\leftarrow B$ and $B \in T_\Theta, Nm(\Theta)\uparrow\omega$
2. G contains the ground literal $\leftarrow B_1 \wedge \ldots \wedge Bn$ and $\Theta \cup \{\leftarrow Bi\}$ is unsatisfiable, $1 \leq i \leq n$,
3. G contains the clause $\leftarrow B_1 \wedge \ldots \wedge Bn$ and $\exists\theta$ such that $B_1\theta, \ldots, Bn\theta$ are ground, and $\Theta \cup \{B_1\theta \wedge \ldots \wedge Bn\theta\}$ is unsatisfiable.

Let C be a clause. A is a logical consequence of a set of clauses, Θ, $\Theta \models^{nm} A$ if and only if $A \neq MC$ and $\Theta \cup \{\leftarrow A\}$ is unsatisfiable, or$A \in Nm(\Theta)$.

As an illustration, we can use the kiwi example again.

$$T_{\Theta 1, Nm(\Theta)\uparrow 0} = 0$$

$$T_{\Theta 1, Nm(\Theta)\uparrow 1} = \left\{ \begin{array}{l} Kiwi(Charlie) \\ Bird(Susan) \end{array} \right\}$$

$$T_{\Theta 1, Nm(\Theta)\uparrow 2} = T_{\Theta 1, Nm(\Theta)\uparrow \omega} = \left\{ \begin{array}{l} Kiwi(Charlie) \\ Bird(Charlie) \\ \overline{Fly}(Charlie) \\ Bird(Susan) \\ Fly(Susan) \end{array} \right\}$$

We are interested specifically in whether *Fly(Charlie)* is a logical consequence of Θ_1. In terms of nonmonotonic logic we are interested in whether $\Theta_1 \models^{nm}$ *Fly(Charlie)*. Since *Fly(Charlie)* is ground, this is true just in case *Fly(Charlie)* $\in T\Theta 1, Nm(\Theta)\uparrow\omega$, which it is not since *Fly(Susan)* $\in \Theta 1, Nm(\Theta)\uparrow\omega, \Theta_1 \models^{nm}$ *Fly(Susan)*.

Note the difference between predicate negation and the negation symbol of standard first-order logic. As far as negation is concerned, the logic presented here functions in the same manner as unaugmented Horn-clause logic. Negation can easily be added to the logic presented here by completing the theory Θ. The negation as failure rule will allow deduction of negative information closer to the standard first-order meaning. Care must be taken to differentiate negation as failure and predicate negation in the logic developed here. The two cases involve different operators and different operations.

The kiwi example is a means to elucidate the potential for nonmonotonic logic techniques. In general, there is a class of problems in information retrieval that has as its core the problem of non-crispness. That is, there is some ambiguity to the set relationships that develop and then are extended via logic systems. Information retrieval, in its infancy, was concerned with the 'one best answer.' As information retrieval as a field of study matured, the acceptance of fuzzy set solutions to information retrieval problems was generally acknowledged.

Nonmonotonic logic addresses the operational side of the information retrieval problem. One may define a set as fuzzy and deal with probabilistic responses to set algebra, but the underlying logic used was first-order logic. Nonmonotonic logic systems allow the use of fuzzy sets systems while also allowing the possibility of contradiction in the outcome. As a logic system, nonmonotonic logic allows the user to work through an information retrieval problem, reach a solution that was perhaps unintended and contradictory to the initial state, and still be able to interpret the results in a sound and complete manner.

3. DISCUSSION AND CONCLUSION

1. The use of standard, first-order logic requires the absence of contradiction. Traditionally, the process used in information retrieval is to do one of the following. Negate an entire strategy when faced with contradiction.
2. The second strategy is to ignore the requirement in monotonic logic of lack of contradiction and proceed.

Both have significant faults. One of the starting points for this paper was the realization that while most information retrieval techniques operate on a first-order logic model, actual searching sometimes assumed a nonmonotonic process. This latter point is especially true in the cases where searches are complex. The interim stages of searching, which can be termed middle or undetermined stages, are often points at which search strategies change as a result of preliminary results.

The problem associated with negation in searching large scale databases can be dealt with to some extent with nonmonotonic logic. Although the process demonstrated here did not employ negation, it can be added without difficulty. Negation as failure is a common concept in database logic (Clark, 1978). Negation in nonmonotonic logic is not always failure in the sense that it is a failure in predicate logic. In fact, it can be argued that negation in nonmonotonic logic structures will yield significant information (Etherington, 1987).

Nonmonotonic logic offers a potential solution to the logical problem faced with first-order models. Perhaps of greatest importance, it will allow the designers of search engines greater latitude in developing tools that work in less than ideal conditions. Work in this area is already in progress (Apt & van Emden, 1982).

Nonmonotonic logic is also useful in the formulation of retrieval queries and in the teaching of retrieval techniques. The reliance on first-order logic is natural but, as information requests become more complicated and data more replete, techniques such as nonmonotonic logic will allow greater relevance and higher precision of searching.

REFERENCES

Antoniou, G. (1996). Operational characterizations of extensions in some logics for default reasoning. *Information Sciences, 89*, 261–273.

Apt, K. R., & van Emden, M. H. (1982). Contributions to the theory of logic programming. *Journal of the ACM, 29*, 841–862.

Beringer, A. & Schaub, T. (1993) Minimal belief and negation as failure: a feasible approach, *Proceedings of the National Conference on Artificial Intelligence, Menlo Park*, CA: AAAI.

Cadoli, M., & Lenzerini, M. (1994). Complexity of propositional closed world reasoning and circumscription. *Journal of Computer and System Sciences, 48*, 255–310.

Chang, C. & Lee, R. (1973) *Symbolic logic and mechanical theorem proving*, New York: Academic Press.

Clark, K. L. (1978) Negation as failure. In H. Gallaire & J. Minker (Ed.), *Logic and databases*, New York: Plenum.

Darwiche, A. & Pearl, J. (1994) Symbolic causal networks, *Proceedings of the 12th National Conference on Artificial Intelligence*, Menlo Park, CA: AAAI.

Etherington, D. W. (1987). Formalizing nonmonotonic reasoning systems. *Artificial Intelligence, 31*, 41–85.

McDermott, D., & Doyle, J. (1980). Nonmonotonic logic I. *Artificial Intelligence, 13*, 27–39.

McDermott, D., & Doyle, J. (1982). Nonmonotonic logic II. *Journal of the ACM, 29*, 33–57.

Narayanan, A. (1996). Revisable knowledge discovery in databases. *International Journal of Intelligent Systems, 11*, 75–96.

Reiter, R. (1980). A logic for default reasoning. *Artificial Intelligence, 13*, 81–132.

Subrahmanian, V. S., Nau, D., & Vago, C. (1995). WFS plus branch and bound equals stable models. *IEEE Transactions on Knowledge and Data Engineering, 7*, 362–377.

Tarski, A. (1955). A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics, 5*, 285–309.