AN INVESTIGATION INTO INFERENCE WITH RESTRICTED QUANTIFICATION AND A TAXONOMIC REPRESENTATION

Alan M. Frisch

Abstract

The goal of this investigation is to design and analyze efficien inference methods that arise when a representation language is augmente with restricted quantification and a taxonomic representation. Thes inference methods will be demonstrated by integrating them into th design of a logic-programming system and a knowledge retriever.

Cognitive Science Research Paper

Serial no: CSRP 041

The University of Sussex
Cognitive Studies Programme
School of Social Sciences
Brighton BN1 9QN

1. Introduction

The goal of this investigation is to design and analyze efficient inference methods that arise when a representation language is augmented with restricted quantification and a taxonomic representation. This paper begins by explaining what restricted quantification is, showing that it can link a traditional logical representation to a taxonomic representation such as those prevalent in semantic networks. It continues by outlining how and why this investigation will integrate taxonomic inference techniques found in semantic networks with logic-based inference techniques currently used in logic programming and knowledge retrieval.

This study will examine the design of a logic-programming language extended with restricted quantification and a taxonomic representation and of an interpreter tailored to exploit this extension. The computational advantages obtained by such a system will be characterized.

AI reasoning systems commonly contain a large corpus, or knowledge base of declarative knowledge and a set of facilities for other components of the system to retrieve this knowledge. The design and specification of such facilities is the second setting within which restricted quantification is investigated. My approach is to view retrieval as a special form of inference. Because efficiency, rather than power, is the primary constraint, the designer of a retriever needs to identify efficient inferences and specify that only these may be performed. Taxonomic inferences are prime candidates for inclusion and restricted quantification will be shown to be valuable in specifying that a retriever should make these inferences but not others. ARGOT (Allen, Frisch and Litman, 1982), a computer system that participates in natural-language dialogs, illustrates how an AI system can be organized around an inference-based knowledge retreiver that automatically reasons with restricted quantifiers and a taxonomic representation. It should be no surprise that the implementation of this kind of knowledge retriever is simplified by use of the extended logic-programming system mentioned above.

2. What is Restricted Quantification?

First-order universal quantification expresses that <u>every</u> individual in the universe has a certain property and existential quantification expresses that <u>some</u>, unspecified, individual in the universe has a certain property. So, for example, one could say about everything that satisfies T. I introduce "All x:T P" as a semantically equivalent way of re-expressing sentences of this form; the motivation for doing this will be seen. Hence, (1) can be rewritten equivalently as

(1') All x:integer x>13 v x<13 v x=13

The universal quantifier here ranges over only a subset of the domain--the set of integers--and is therefore called a restricted quantifier. A similar notation can also be introduced for restricting existential quantifiers. The symbol which expresses the restriction on a quantifier (that is, "T" in "All x:T") is called a sort symbol and the subset of the universe that it denotes is called a sort.

A representation language with restricted quantification needs a way to represent that certain relationships hold among the sorts, to express, for example, that the primes are a subset of the integers and that the primes and composites are disjoint. Furthermore, there is a need to express that certain individuals are members of certain sorts--for example, that 3 is a prime. I refer to that part of the representation that expresses such information as the taxonomic representation, to a representation language that has restricted quantification and a taxonomic representation as an RQT language, and to inferences that deal specifically with these extensions as RQT inferences. The use of a taxonomic representation that allows for the case where only partial information about the taxonomy is represented distinguishes this study from those where complete information is assumed.

These syntactic extensions to a first-order language in no way increases its expressiveness; everything that can be expressed a first-order RQT language can be expressed in a normal first-order language. Furthermore, the RQT inferences can be seen as special cases of standard logical inferences. As an example consider inheritance, the most well-known form of RQT inference performed by semantic-network systems. Given the fact that all dogs are mammals and that Jellybean is a dog inheritance can be used to infer that Jellybean is a mammal. However, as a standard logical inference this would be seen as inferring "mammal(jellybean)" from "dog(jellybean)" and "All x dog(x) -> mammal(x)" -- an instance of universal instantiation and modus ponens.

3. Logic Programming: Using Restricted Quantification in Computation

This section illustrates how a simple deductive problem can be handled better by a Prolog-style system once it is extended with restricted quantification and a taxonomic representation. I am primarily concerned with an inference system designed by Reiter (1977) that deals with the

Historically, there has been little to demonstrate the computational advantages of using restricted quantification. However, recently two theorem-proving systems employing techniques related to RQT inference have solved Schubert's Steamroller (Cohn, 1984; Walther, 1984a), a challenge problem unsolved by traditional methods. Though this is a dramatic illustration of the potential of these inference techniques a thorough analysis is needed of when, how much, and why the approach pays off. Such an analysis will be a major concern of this investigation but meanwhile the advantages can be illustrated by considering a Prolog-style theorem prover attempting to derive

(2) Exists x has(x,tires) & has(x,doors) & owns(alan,x)

from the following set of sentences:

(3) All x bicycle(x) -> vehicle(x)
(4) All x car(x) -> vehicle(x)
(5) bicycle(b1) & bicycle(b2) & ... & bicycle(bn)
(6) car(c1) & car(c2) & ... & car(cn) & car(mycar)
(7) All x vehicle(x) -> has(x,tires)
(8) All x car(x) -> has(x,doors)
(9) owns(alan,mycar)

It would first set out to solve "has(x,tires)" and eventually succeed with, for example, x=b1. Then upon failing to show "has(b1,doors)" it would backtrack to find another x such that "has(x,tires)." It would repeat this futile search until it had tried each bicycle individually. The search would then consider each car and eventually succeed with x=mycar.

A little thought reveals the source of the problem in the above search space. The goal, "has(x,tires)" only succeeds once x is instantiated to the name of something that has tires, for example, b1. But x need not be b1 for the goal to succeed; it could be any vehicle. It is a gross overcommitment to restrict x to be b1 and the price is paid when the commitment is later retracted. A general, well-known guideline of problem-solving is to defer guessing (searching) as long as possible and when a guess must be made, to make the smallest commitment necessary. Unfortunately in the above representation x is either uninstantiated (totally unrestricted) or it is instantiated to an individual. There is no way to restrict x to being a vehicle without restricting it to be a particular vehicle.

By re-expressing the problem in an RQT language and using Reiter's deductive scheme the problem can be solved by a minimal-commitment strategy that avoids backtracking. The first subgoal, "has(x,tires)," would be achieved by restricting x to the set of vehicles, and likewise, the second goal by further restricting x to the set of cars. This then arising problems, there is no guarantee that it always does so. A goal of this investigation is to find precisely when a Prolog system with restricted quantification is more efficient than one with standard quantification.

Further efficiency can be gained by pre-computing much of the reasoning with the taxonomic representation that is done during unification. This investigation will look at the tradeoff between expressiveness of the taxonomic representation and the application of various methods for reasoning with it, particularly methods utilizing pre-computation.

4. Knowledge Retrieval: Using Restricted Quantification in Specification

Though there has been a growing concern for formalization in the study of knowledge representation little has been done to formalize the retrieval process. My ongoing research is an attempt to remedy this situation (Frisch and Allen, 1982). The key maneuver in this study is adopting a view of retrieval as a kind of inference. This enables the techniques of mathematical logic to be used in specifying retrieval and studying its properties. I have adapted the standard tools of proof theory and model theory to the study of retrieval (Frisch, in preparation). If retrieval is inference, then the knowledge base (KB)--in addition to being a data structure--is a representation. The user of the KB is not querying the retriever about the data structures in the KB but rather about the world that these data structures encode knowledge of.

The retrieval problem that I have studied takes the knowledge base to be a set of sentences of the first-order predicate calculus (FOPC). A query asks the retriever to retrieve a specified closed sentence of FOPC to which it responds "yes" or "no." So, for example, one could ask the retriever, "Can `UNCLE(JOHN,BILL)' be retrieved?" It is not difficult to extend this notion of query to include FOPC sentences with free variables. Such an extension enables one to ask, "What are all the x's such that `UNCLE(x,BILL)' can be retrieved?" For purposes of this exposition it suffices to consider only the first form of query.

A specification of a retriever must determine whether it says "yes" or "no" for any given KB and any given query. Just as one can speak of a sentence logically following, or being provable, from a set of sentences, one can speak of a queried sentence being retrievable from a set of sentences contained in a KB.

Though it is reasonable to demand that the retrieval process is guaranteed to terminate, inference processes, in general, have no such guarantee. This necessitates limitation of the inferences that are made

The major result of my work on retrieval to date is the specification and analysis of a retriever so severely limited that it cannot chain two facts together in order to respond affirmatively to a query. The next step in this investigation is to extend the system to perform a class of useful inferences that require such chaining but nonetheless can be controlled adequately enough to be performed automatically and efficiently during every retrieval. Specifically, the use of the RQT inferences will be considered. So, for example, the resulting system would perform inheritance but not arbitrary modus ponens.

But this presents a problem: how can a retriever be specified and implemented so that it performs inheritance but not arbitrary modus ponens when, after all, inheritance can be seen as a kind of modus ponens? The approach that semantic-network researchers take to this problem is to use a special notation to express taxonomic information. Hence the fact that Jellybean is a dog is expressed with a distinguished arc rather than with a general notation for implication. But this is precisely what is done in an RQT language. The specification of an inference system can then make use of this special notation. A proof theory can be extended with special axioms and rules of inference to handle the new expressions while a model theory can be expanded to assign meaning to the new expressions.

Once a retriever is specified, efficient algorithms are needed for its implementation. Many of the algorithms can be obtained by adapting those used in automated deduction while others must be developed specifically for the retriever. Of particular interest to this investigation is the use of the RQT inference techniques mentioned in Section 3. Ideally, this would be simple if the retriever were implemented in the RQT logic-programming system.

Thus, in the proof theory, model theory, and algorithms, restricted quantification is useful in specifying that a retriever is to reason with the taxonomic representation without necessarily specifying that it should do other inferences. Yet there are difficult problems to solve in getting the three forms of specification to coincide. As discussed in Section 3, the RQT inference techniques that I wish to use in the implementation are not, in general, complete. The problem, then, is to find the restrictions that must be placed on the representation so that the retriever's RQT inferences can be implemented by embedding them in the RQT inferences of the logic-programming system.

5. Comparison with Related Work

KL-ONE [Brachman 1979] was one of the first representation systems to

RQT systems designed by Reiter (1977) and by Minker and McSkimin (1979) both assume that the taxonomic representation has complete knowledge of the relationship between every sort and every individual. Reiter makes this assumption to obtain completeness and Minker and McSkimin to precompute all knowledge about the taxonomy. Furthermore, in the tradition of logic databases, both systems severely restrict the representation language by eliminating function symbols. My investigation is an attempt to obtain completeness and computational efficiency without assuming complete knowledge or restricting the representation language.

Cohn (1983a; 1983b) has investigated an inference system for a polymorphic sorted logic with a taxonomy and restrictions on arguments to both function and predicate symbols. Walther (1983) has worked on a similar system with a somewhat less-expressive language though it does incorporate restricted quantification and equality. Both systems require complete knowledge of the taxonomy, a restriction that I am not imposing. Neither of these systems can guarantee that resolving two clauses together results in a single clause. This is especially disquieting for logic-programming because it means that a rule can be applied to a goal in more than one way. I consider the uniqueness of resolvents to be essential and hence propose to find the conditions that ensure this. This issue is closely related to the uniqueness of most-general unifiers, an issue recently investigated by Walther (1984b).

Work on HORNE (Allen, Giuliano and Frisch, 1983; Frisch, Allen and Giuliano, 1983), an RQT logic-programming system, has led to the formulation of many of the issues discussed here. Most importantly its implementation demonstrates the feasibility of the approach and its use at the University of Rochester and many other sites demonstrates its utility. The HORNE implementation incorporates a number of effective methods for dealing with a taxonomic representation. This investigation will analyze and extend those methods.

6. Summary of Expected Results

This investigation is expected to result in the specification of RQT inference techniques and a characterization of their properties. When are these techniques more efficient than standard inference techniques? How much more efficient are they? Under what conditions are the RQT inferences complete? What is necessary to ensure the uniqueness of most-general unifiers? Further expectations include the integration of the RQT inference techniques into the design of a logic-programming system and an inference-based knowledge retriever.

Acknowledgements

I thank Tony Cohn, Bob Kowalski and Christoph Walther for their constructive comments on this paper and Roger Evans for his remarks on using RQT inference for parsing. I am especially grateful to Aaron Sloman whose detailed criticism of several early versions of the paper lead to significant revisions and the omission of many needless words.

References

Allen, J.F., A.M. Frisch and D.J. Litman, "ARGOT: The Rochester dialogue system," in *Proceedings of the National Conference on Artificial Intelligence*, August, 1982.

Allen, J.F., M.E. Giuliano and A.M. Frisch, "The HORNE reasoning system," Technical Report 126, Computer Science Dept., U. Rochester, December, 1983.

Brachman, R.J., "On the epistemological status of semantic networks," in N.V. Findler (Ed.), *Associative Networks: Representation and Use of Knowledge by Computers*. New York: Academic Press, 1979.

Brachman, R.J., R.E. Fikes and H.J. Levesque, "KRYPTON: A functional approach to knowledge representation," *Computer, 16,* 67-73, October, 1983.

Cohn, A.G., "A Note concerning the axiomisation of Schubert's steamroller in many-sorted logic," Alvey Workshop on Inference, Imperial College, Sept., 1984.

Cohn, A.G., "Mechanising a particularly expressive many sorted logic," Ph.D. Thesis, Dept. of Computer Science, Univ. of Essex, 1983a.

Cohn, A.G., "Improving the expressiveness of many sorted logic," in *Proceedings of the Third National Conference on Artificial Intelligence*, August, 1983b.

Frisch, A.M., "Using model theory to specify programs," in preparation.

Frisch, A.M. and J.F. Allen, "Knowledge retrieval as limited inference," in D.W. Loveland (Ed.), *Lecture Notes in Computer Science: 6th Conference on Automated Deduction*. New York: Springer-Verlag, 1982.

Frisch, A.M., J.F. Allen and M Giuliano, "An overview of the HORNE logic programming system," *SIGART Newsletter,* No. 84, 1983. Also in, *Logic Programming Newsletter,* No. 5, Winter, 1983/4.

Israel, D.J., "On interpreting network formalisms," *International Journal of Computers and Mathematics*, *9*, 1-13, 1983a. Also in N. Cercone (Ed.), *Computational Linguistics*, Oxford: Pergamon, 1983a.

Israel, D.J., "On the semantics of taxonomy-based semantic networks," 1983b.

McSkimin, J.R., and J. Minker, "A predicate calculus based semantic network for deductive searching," in N.V. Findler (Ed.), *Associative*

Walther, C., "Unification in many-Sorted theories," *Proc. of the Sixth European Conference on Artificial Intelligence*, 1984b.

Walther, C., "A many-sorted calculus based on resolution and paramodulation," *Proc. of the Eight International Joint Conference on Artificial Intelligence*, August, 1983.