these implementations were significantly more efficient, but checked a somewhat narrower class of presumptions than CO-OP.

6. Damerau mentions that queries with non-empty responses can also make presumptions. This is certainly true, even in more subtle ways than noted. (For example, "What is the youngest assistant professors salary?" presumes that there is more than one assistant professor.) Issues such as these are indeed currently under investigation.

Overall, we are pleased to see that Damerau has raised some very important issues and we hope that this exchange will be helpful to the natural language processing community.

Aravind K. Joshi
Dept. of Computer and
Information Science
University of Pennsylvania
Philadelphia, Pennsylvania 19104

S. Jerrold Kaplan
Computer Science Department
Stanford University
Stanford, California 94305

Reply to Joshi and Kaplan

In general, there is little to disagree with in Joshi and Kaplan's comments, but perhaps a couple of points could be clarified.

As Joshi and Kaplan suspected (point 3), the users of this system did indeed thoroughly understand the data base. This makes quite a difference in thinking about the relative importance of facilities in a natural language query system. In particular, such users tend to check strange answers, so that a reply of "no", as in their point 4, would probably result in an additional question of "How many parcels ... ".

With regard to their remarks on implementations that incur no additional cost (points 2 and 5), I would be interested in seeing how presupposition analysis can be done without extra data base retrievals. It would seem that the system would either have to make special retrievals at marked times, as in CO-OP, or would have to make the relevant retrievals for every question so as to have the results available when needed. However, even if the execution time increase were to be zero, we still have a great many other things which we would like to add to our system before we add inference checking.

Fred J. Damerau

Book Reviews

Logic For Problem Solving

Robert Kowalski

Elsevier North Holland, New York, 1979,
287 pp., Paperback, $9.95, ISBN 0-444-00365-7.

This is a textbook introduction to logic programming. Logic programming is based on the premise that programming a task on a computer should begin with a precise formulation of what is wanted. This formulation defines the task clearly; it serves as a theory of the task which can be studied for its implications and limitations. Usually this formulation is computationally inefficient if implemented straightforwardly, but it can be reformulated so that it becomes an efficient program when interpreted by a theorem prover. In this form the logic program is closer to the theory than a PASCAL or LISP program would be, making it easier to verify its correctness and also easier to understand directly.

Logic programming has been applied mostly to formal software specifications, data base systems and problem solving, but it is being applied increasingly to natural language understanding systems [1,2,4,5,6]. In these systems axioms specify the relationship between the input text and whatever representation it is to be parsed into, and between this and whatever the output is to be (e.g., an updated database or the answer to a question). Since these axioms specify the relation between the text and its representation, they form a grammar for the text language, and, as such, are comparable to the rules in a linguist's grammar. When interpreted by a suitable theorem prover, such as a version of PROLOG, they can transform a text into its representation (and often a representation into a text) with practical efficiency.

With logic programming the computational linguist may be able to develop theories of language that are both conceptually well-organized and practical to compute, but this book includes only the most elementary introduction to natural language processing. It uses parsing as an example to show that problems can be solved in ways that correspond to top-down parsing, bottom-up parsing, or an arbitrary mixture of the two, all depending on how the theorem prover decides to

apply the axioms. But Kowalski's examples do not show how to build up a representation structure for a sentence; to learn how to do that it is necessary to consult the natural language papers cited above.

This book is organized into three parts. The first part introduces logic syntax and semantics. The notation is the clausal form of logic, in which all axioms look like implications. This form allows the elimination of 'not' as an explicit logical operator, which makes the form psychologically easier to understand; it is in fact a disguised form of the clauses used in resolution theorem proving. This introduction to logic includes a discussion about clauses and semantic networks. Clauses can be embedded in networks if the arcs corresponding to the atomic formulas are grouped together in a set, and the arcs are further grouped according to their roles. By restricting all atomic formulas to be binary relations, clauses become a linear representation of a simple network structure. This embedding of clauses is thus a practical way to build logical inference into semantic network-based knowledge systems. (cf. [3].)

The second part of the book explores various inference procedures and heuristics for logic programming and applications of logic programming to problem solving. Most of the procedures are seen to be applications to the clausal formalism of well-known heuristics and search procedures, such as pre-processing the problem-independent parts of the computation, using evaluation functions, indicating when an axiom is to be applied in a forward manner and when in a backward manner, choosing the path that branches the least, etc. This part of the book is thus an introduction to the heuristic search methods often covered in introductory courses on artificial intelligence. Most of this section limits the clauses to those having only one conclusion; these are called Horn clauses and have direct interpretations as programs and as rules for problem reduction. Several chapters discuss the problems and techniques for processing axioms in full clausal form, however, which shows that this book presents logic programming as a concept that is independent of any particular PROLOG implementation.

The last part of the book introduces more advanced topics. These include extensions of logic programming to the standard form of logic, addition and deletion of redundant goals, traps to prevent useless looping, allowing the provability of some formulas to depend on the unprovability of others, and the combining of object language with meta-language. The final chapter axiomatizes the four ways that an information system or belief system might change when a new fact is added to it. Only the top level axioms are given, however; many of the relations named in the axioms need to be further defined before there is a full theory of rational belief maintenance.

This book is intended to be a textbook that introduces the undergraduate to logic, problem solving and computer programming. Except for one chapter that compares Horn clauses to conventional programming languages, it assumes the student has no background in any of these areas. It covers many topics, but covers most of them briefly, so that one has to look up some of the many references if one wants more than an elementary treatment.

Daniel Chester, University of Delaware

References

[1] Colmerauer, A. Metamorphosis Grammars. in L. Bolc, ed., *Natural Language Communication with Computers,* Springer-Verlag, Berlin, 1978, 133-189.

[2] Dahl, Veronica. Quantification in a Three-valued Logic for Natural Language Question-answering Systems. *Proceedings of the Sixth International Joint Conference on Artificial Intelligence,* Tokyo, August 1979, 182-187.

[3] Deliyanni, Amaryllis, and Kowalski, Robert A. Logic and Semantic Networks. *Comm. ACM 22, 3,* (March 1979), 184-192.

[4] LeVine, Sharon H. Questioning English Text with Clausal Logic. *M.A. Thesis,* University of Texas at Austin, December 1980.

[5] Pereira, F.C.N., and Warren, D.H.D. Definite Clause Grammars for Language Analysis - A Survey of the Formalism and a Comparison with Augmented Transition Networks. *Artificial Intelligence 13, 3,* (May 1980), 231-278.

[6] Silva, Georgette, and Dwiggins, Don. Toward a PROLOG Text Grammar. *ACM Sigart Newsletter 73,* (October 1980), 20-25.