Robotica (2003) volume 21, p. 697. © 2003 Cambridge University Press
DOI: 10.1017/S0263574703005514 Printed in the United Kingdom

Book Review

DOI: 10.1017/S0263574703215514

KNOWLEDGE REPRESENTATION, REASONING AND DECLARATIVE PROBLEM SOLVING, by Chitta Baral, University Press, Cambridge, 2003, xiv+530 pp., ISBN 0-521-81802-8 (Hbk, £60.00).

ln recent decades there has been much interest in programming computers declaratively rather than procedurally. In declarative or logic programming the steps to be taken to reach the required result are not specified and instead the system is given a set of logical assertions and some sort of query. The first two paragraphs of the Preface to this book explain the point very clearly:

> "Representing knowledge and reasoning with it are important components of an intelligent system, and are two important facets of Artificial lntelligence. Another important expectation from intelligent systems is their ability to accept high level requests – as opposed to detailed step-by-step instructions, and their knowledge and reasoning ability are used to figure out the detailed steps that need to be taken. To have this ability intelligent systems must have a declarative interface whose input language must be based on logic.
>
> "Thus the author considers the all-round development of a suitable declarative knowledge representation language to be a fundamental component of knowledge based intelligence, perhaps similar to the role of the language of calculus to mathematics and physics. Taking tbe calculus analogy further, it is important that a large support structure is developed around the language, similar to the integration and derivation formulas and the various theorems around calculus."

The kind of support structure visualised is of course useful as an advanced "search engine" when coupled to databases and other sources of information. It also has exploratory and planning capabilities that are often illustrated by reference to the kind of task that might be required of a personal assistant, whether human or electronic, in for example planning a trip to anend a conference. The system might be required to find suitable combinations of flights, hotel accommodation, car hire, etc., all with verification of availability, and chosen to satisfy as far as possible the known preferences of the enquirer, presumably with attention to budget and to the complications of such things as conference discounts and frequent-flyer benefits. Such a capability is obviously useful in robots and autonomous agents in general.

Another reason for interest is that such a system can allow rapid and probably error-free prototyping, since the requirements for a new system can be expressed much more quickly and reliably in their declarative form than in terms of procedural programming.

The kind of problem-solving operation nceded has been well explored in mathematical theorem-proving, and the programming language *Prolog* has been available for some time. The new book is a comprehensive treatment of a development referred to as *AnsPmlog* (also called *A-Prolog*), or "logic programmning with answer set semantics". There is a great deal of formal mathematics and the difficulty of getting to grips with all of it is acknowledged where the use of the book as a teaching text is discussed. Only part of it is expected to be covered in an undergraduate course, and it is appropriate for a taught graduate course.

Even the meaning of "answer-set semantics", also referred to as "stable model semantics", can only be explained by invoking some mathematical theory. *AnsProlog* is more truly declarative than standard *Prolog*, since when using the latter it is necessary to consider the order in which the "literals" within a rule are listed, and therefore the order in which they will be processed. This means that progranuning in *Prolog* is partly procedural, whereas *AnsProlog* is untainted.

It is argued convincingly that *AnsProlog* should be the system of choice for practical applications both because of its versatility and because efficient software bas been developed, such that very large programs can readily be handled. A number of alternative formulations of *AnsProlog* are described, differing by the allowing or disallowing of certain operators in the heads of rules. The unrestricted version is indicated by *AnsProlog** (possibly causing confusion since it is not immediately obvious that the asterisk does not refer to a footnote!) Alternatives are indicated by replacing the asterisk with a superscripted listing of the allowed or disallowed operators, so that a superscripted "-**not**" shows that the operator "**not**" is disallowed. (The operator **not** is distinct from the negation operator "¬" because the system allows for variables having the three possibilities of true, false and unknown.)

The use of the method is illustrated with some impressive examples' including its application to a combinatorial auction, where participants can bid for any subset of objects offered, and the task of the auctioneer (or *AnsProlog* program) is to select the combination of bids that will maximise the total return subject to the condition that no object is sold more than once. Other standard combinatonal problems are used as illustrations, one of them being the deduction of the correct ownership of a pet (a zebra, in the example) found wandering, where the choice of its home, out of five possibilities, must be derived from a set of fourteen statements about the characterisitics of the five houses and their occupants.

Even more convincing of the the power of the method is a practical application to the planning of actions to be taken in a space shuttle when there is failure of components that help control maneuvering jets. This refers to a real project carried out and implemented by a NASA contractor and groups in the University of Texas.

A great deal of rele ant information appears on the website *http:www.baral.us/bookone*, including coding for the examples and a good deal of downloadable software in C++, and a set of slides in PowerPoint format that could be the basis of an introductory lecture. Probably even more helpful for a beginner is another set of slides with the title: "Answer Set Programming What it is and how to play with it", due to Aarati Parbat. An interesting point is that she is associated with Professor John McCarthy, whose involvement is noteworthy since he laid foundations for logic programming in a very early paper.[1]

It seems clear that *AnsProlog* represents the current state of the art in applicable logic programming and that this book should be seen as the definitive guide to it.

Alex M. Andrew

Reference

1. J. McCarthy, "Programs with common sense", **In:** *Mechanisation of Thought Processes* (Proceedings of a symposium in the National Physical Laboratory) (HMSO, London, 1969) **vol. 1**, pp. 75–91.