The book contains definitions and theorems on almost every page, and many of them are generalizations of results related to Horn clause logic programming. Reinterpretation of some well known results of mathematical logic in this domain is quite interesting, for example, the Church–Rosser Property of SOS and the Halting Problem. Although the author provides a very fluid introduction, the rest of the material is very theoretical and difficult to follow, due to the lack of enough motivating examples.

Reviewed by Subrata K. Das, Queen Mary and Westfield College, Mile End Road, London E1, UK

Formal specification of complex reasoning systems edited by Jan Treur and Thomas Wetter, Ellis Horwood, Chichester, 1993, pp 282, £35.95, ISBN 0-13-336-785-1.

In its youth, KBS research was vigorous and dynamic, but with a disturbing tendency to impulsiveness and ill-considered assertion. With the onset of middle-age, the confidence of adolescence has given way to a more introspective mood, with an accompanying desire to be taken seriously by the wider research community. Two of the most obvious role models are software engineering (with its emphasis for standardized methodologies and tools for program development) and computational logic (which supplies methods for describing and proving properties of systems). This book's primary theme is the use of computational logic for the specification of KBS, but this is intertwined with a methodological subtheme which subtly influences the authors' choices of formal systems. The general approach is to provide a snapshot of recent research across a variety of research groups. Eight different systems are described and, to provide a platform for comparison, a standard example is used by each system. Although the example is (of necessity) quite small, it provides an invaluable anchoring point for the more abstract technical discussions.

Although the book is organized as a series of collected papers, sandwiched between an introduction to the running example and a conclusion which draws comparisons between the systems, it is more interesting to review it by drawing out some of the dominant issues which surface sporadically throughout. The most noticeable of these is the choice of specification language, which varies considerably between systems. It is ironic (given that these languages are intended to rationalize KBS design) that it is difficult to provide a direct comparison of them, although a brave attempt is made in the concluding chapter. This is an example of the subtle influence of methodology on formal methods, since the differing "views of the world" taken by each of the research groups has clearly exerted a strong influence on the shape of their formal systems. This divergence is (I suspect) further emphasized by the high level of expertise of each group, which makes it possible for them to select the formal framework which is "just right" for their particular approach. As a result, we are presented with a range of specialized logical frameworks, tuned to each research group's interests. Although some of these sit directly on top of established specification languages (such as OBJ3), most are blends of different varieties of computational logic, applied to different stages of KBS development—for example, $K_{BS}SF$ uses both an algebraic specification language and first order predicate calculus. Many of the systems described are designed to accommodate this diversity using various types of modularity—with refinement between modules in MILORD; design by composition of modules in DESIRE and communication by bridge rules between modular theories in MC.

Along with the emphasis for modularity comes the desire to stratify the specification task in some way. Earlier work on "semi-formal" KBS design methodologies (KADS in particular) has exerted a strong influence on the solutions proposed here. ML^2 explicitly takes KADS as its starting point and offers formal accounts of the domain, inference and task levels of development—the general idea being that the inference level transfers information to and from the domain level whilst the task level overlays control information on the inference level. A similar approach is taken in the KARL system, although using a more object-oriented style of specification and with more weight given to the visualization of each level using diagrammatic notations. Both ML^2 and

KARL have the feel of "logical modelling" exercises, in which the intention is to produce an elegant and rigorous formal account of some, hitherto informal, methodology. A more pragmatic line is taken in the AIDE system, which is more clearly in the lineage of conventional expert system shells, giving a separation between strategic, domain and case-specific knowledge in a style reminiscent of NEOMYCIN.

Since this book is a description of work in progress, it is unreasonable to expect firm conclusions about the contribution which each of the formal systems might make to KBS development. However, the concluding chapter is effective in drawing together the main lessons from each of the (independent) earlier chapters. Of particular interest are the open problems which are highlighted at the end of the book, since these are shared with the wider software engineering, formal specification and logic programming communities. In summary, these are: integration of system components; describing systems which involve database updates; merging with other software development environments; providing reliable formal mappings between specification languages; coping with (sometimes unexpected) interactions with the external environment. To these we might add the need to demonstrate the benefits of formal frameworks to KBS design problems of realistic size, since the "goodness of fit" between the abstract design of a formal framework and the reality of its use in KBS design is an empirical, as well as theoretical, issue. One of the virtues of this book is that it shows a willingness to address the problem—a trend which, I believe, stems from the growing openness of specialists in the KBS formal methods community. I expect that this book will be helpful in making such techniques more accessible to the wider research community and recommend it as an introduction to formal methods in KBS.

Reviewed by Dave Robertson, Department of Artificial Intelligence, University of Edinburgh, 80 South Bridge, Edinburgh, UK

Genetic algorithms for machine learning edited by John J. Grefenstette, Kluwer Academic, USA, 1993, £72.50, pp 163, ISBN 0-792-39407-0.

This book is concerned with Genetic Algorithms (GA) and their use in machine learning. It is a reprint of the special issue of *Machine Learning* journal (**13**, No. 2–3). The same five articles in the journal are included in the book, dealing with different aspects of GAs applied to machine learning. The first two articles are about the representation scheme of genetic operators and describe two contrasting approaches. The first approach is to utilize traditional string representation while the second one is to use symbolic rules for problem solving. The third article describes a system called COGIN, showing how to utilize GAs for competition-based learning. In the fourth article, the performance of GA-based reinforcement learning is elaborated and it is compared with a neural network-based learning procedure. The last paper is somewhat different in scope. It is rather a theoretical work, and introduces a tutorial on how to define a good fitness function. It relates the results to machine learning at the end anyway.

I should admit that the book includes very thoughtful ideas, and I have learned something while reading it. It is mainly well organized (apart from two page numbers on one page and a poor index table), and the papers are carefully selected. The results illustrated in the papers are clear enough to understand the ideas presented. They also indicate some future research directions. For example, although the first paper by De Jong et al. describes a learning procedure for a single-class problem, it can direct the reader to think on how to apply the methodology in multi-class problems concerned with real time events in a dynamic environment. The second paper by Janikow provides some ideas to develop justification and validation procedures for learning systems which I believe, is usually ignored in machine learning research. As the authors agree, the third paper by Greene and Smith provides ideas on how to develop a general symbolic induction algorithm for solving continuous-valued classification problems. The fourth paper by Whitley et al. compares reinforcement learning procedures on a single control problem (pole balancing). The results indicate that