the like, and they contain a lot of information about how to implement efficient logical reasoners. The section on uncertain knowledge contains an excellent introduction to probabilistic reasoning and belief networks. Moreover, it introduces decision theory covering topics like multiattribute utility functions, decision networks, sequential decision problems and dynamic decision networks. The section on learning, one of my favourites, present all sorts of approaches ranging from "subsymbolic" back-propagation learning in neural nets, genetic algorithms, decision tree learning, explanation based on learning to inductive logic programming, and it puts all these approaches into perspective.

The book also contains very valuable information about how specific approaches and techniques were used in real applications, how successful they were—or why they failed. The planning section is a particularly good example of this. With this information the reader gets a pretty good feeling of what can be done at the moment, and where the big problems are.

A further important aspect distinguishing this book from others is the common unifying perspective under which all the different approaches are presented. The authors view AI as the science of intelligent agent design. Under this view all the bits and pieces from various subfields of AI fit together very nicely. For novices this provides a lot of orientation. Advanced researchers get the great feeling that what they do is not only relevant to them and their little AI subcommunity.

The book is very well-written and clear with an excellent balance between motivation, formalisation and application. To make the underlying ideas precise the authors use easily understandable pseudo code throughout the book. Actual Common Lisp implementations of the presented algorithms are available via the Internet. The authors show a great ability to invent illustrating and entyertaining examples—often reappearing in several chapters—and their style of writing is very amusing. It's just fun to read this book.

After all this enthusiasm for the book, is there any wish left open? There is. I don't want to mention the few mistakes one finds, rather unsurprisingly given this is the first edition of a book of over 900 pages (of course, consistency of first order logic is *not* semi-decidable, contrary to what is stated on p277). Number 1 on my wish list is a more adequate treatment of nonmonotonic reasoning. Much excellent work has been done in this area in recent years and interesting insights have been gained. All Russell and Norvig, basically, have to say about this is: theoretically interesting but practically irrelevant. I think this is too much of an oversimplification and the topic of nonnumerical defeasible reasoning deserves more than one page in a book like this. I would hope to see one or two extra chapters on this topic in a future edition.

Anyway, there can be no doubt that this is the best general AI textbook available today. If the quality of textbooks mirrors the matureness of a field AI is in much better shape than many of us may have thought. The book is a highly valuable source of information, not just for newcomers. Given its reasonable (if not cheap) price there is a pretty good chance that this wll become the AI bible of the next decade.

Reviewed by Gerd Brewka, Technische Universität Wien, Austria

Concept formation and knowledge revision by Stefan Wrobel, Kluwer Academic, Netherlands, 1994, pp 240, £57.75, ISBN 079239500X.

The construction and maintenance of a knowledge base is a vast and sometimes confusing area of AI and machine learning, engulfed in specific theories and representations from various related fields.

Wrobel presents a holistic approach to the symbolic representation of knowledge, concept formation and updating methods for knowledge bases. The book has an interdisciplinary view and draws from close fields such as psychological studies on human learning, logical formalisms, computational learning and existing machine learning theories. In particular, the author's system, MOBAL, bases its knowledge acquisition system on the architecture described in this book.

In the seven chapters of the book, the approach to creating a knowledge acquisition model is detailed from its basis in psychological human learning trends to the logical representation of knowledge through to the creation and updating of concepts.

Chapter 1 is an introductory chapter describing the necessity and context of the work presented in the book.

The second chapter examines and classifies human concepts according to existing empirical formalisms from psychological research. It describes the various concept categories with which humans reason, and the attributes of these categories. The attributes are later used in Wrobel's system to identify object sets that are to become concepts. The constraints imposed on concept formation processes determine which concepts may be formed. For the development of computer models for concept formation certain constraints are used to create particular representations. Wrobel points out the importance of goal oriented constraints in problem-solving contexts and later adopts this as the *demand driven* approach in the knowledge revision tasks of his system.

The theoretical assumptions of Chapter 2 are then translated into the system model in Chapters 3, 4 and 5, each dealing with the logical representation, knowledge revision and concept formation, respectively. The logical representation featured in Chapter 3 describes the language used for knowledge representation in the computational concept formation technique presented in Chapters 4 and 5.

The representation is presented in a notation similar to Prolog, though MOBAL uses a slightly different syntax. Here, as in the rest of the book, differences and similarities to the MOBAL system are shown clearly. The formal properties and inference rules of the language are detailed in this chapter. Concept definition is also shown, and the representation is measured up against the psychological categories of Chapter 2.

The fourth chapter forms the main substance of the book, the system and probably the essence of the research. It defines the knowledge revision problem and the resulting nonmonotonicity that hounds the knowledge bases of all incremental learning systems. The knowledge revision activities of the problem-solving system are identified and discussed. They are collectively seen as a process of eliminating nonmonotonicity from knowledge bases when they produce unwanted derivations, in a minimally destructive way. In particular, concept formation is seen as an integral part of knowledge revision. The requirements of a minimal revision operator are developed and algorithms for computing minimal revisions and multiple revisions are defined. For cases when specialisation is sought, or when it is necessary to prevent long exception lists, reformulation operators and a search strategy for this specialisation process are defined. The final section of this chapter is devoted to showing how these results have been incorporated into MOBAL's knowledge revision system. Its interactive use is also described.

The fifth chapter brings together the issues of the preceding three chapters, namely the psychological results for concept formation, the logical representation language and the knowledge revision activities, and shows how they interact to form an incremental learning system. The theoretical and empirical properties of the system are related to the psychological requirements for building concepts. The system is also evaluated, both as a computational tool (in terms of structure, accuracy and complexity of a computational system) and with respect to related work.

The sixth chapter presents arguments against claims that symbolic approaches to machine learning are unable to create, or correctly represent, new concepts. In a general assessment, Wrobel also discusses other approaches, including connectionist networks. He concludes that the context of concept formation should be given more attention in ways such as applying constraints that determine which concepts are formed—as in Wrobel's knowledge acquisition system.

Wrobel sums up the contributions of his work to the field of machine learning in the final, concluding chapter, together with suggestions for future research.

Concept Formation and Knowledge Revision provides a complete, evolutionary perspective for a knowledge acquisition system from psychological foundations to the realisation of the technical model. It portrays a splice through the plethora of machine learning formalisms and does so in a informative and fluent manner. The author is sympathetic towards readers not totally familiar with

the area and all opportunities for reminders to related or earlier work are used, even at the cost of repetition.

The insights and connections offered by this book are especially valuable since the system it refers to, MOBAL, is a real working system not just a theoretical possibility. In addition to detailing the development of a specific system, the book attempts to address a wide set of issues in the construction of knowledge acquisition system and will prove useful to all modelling such systems.

Reviewed by Rashmi Pandya, Department of Computer Science, University of Exeter, UK

Algebraic specification techniques and tools for software development—the ACT approach by I. ClaBen, H. Ehrig and D. Wolz, AMAST Series in Computing, Vol 1. World Scientific Publishing Co., USA, 1993, pp 237, £27.00, ISBN 981-02-1227-5.

The authors present a formal algebraic approach (namely ACT) that aims to support "the entire software development process from semi-formal requirements, via functional requirement specification, design specification and executable specification (or prototype) up to the efficient software system version or product." The method is rooted in universal algebra and category theory, and so readers without some grounding in theoretical computer science beware.

In algebraic specification approaches one defines data types with a number of operations on them. The operations are defined in terms of declarative equations relating constructors (primitive objects) and other operations. The advantage of the approach is that it allows a formal semantics to be assigned to specifications, proofs of key properties are possible and mechanisable by term rewriting, and certain classes of specification can be executed, again by term rewriting. Despite this little use has been made of such techniques in industry; this book is trying to change that situation.

In chapter 1 there is an introduction to the use of algebraic specification methods and the theory underlying them. It is a strange mix covering life-cycle issues, underlying theory and tool support. I found this chapter rather intimidating as an introduction because of the passing references to results of some fairly heavy theoretical computer science. Still, this should not deter the uninitiated as most of the rest of the book concentrates on the application of the method. It is also a little simplistic in its treatment of life-cycle issues, and talks about formal refinement in a very matter of fact way, however the real difficulties of doing this for large, complex systems is glossed over.

The ACT ONE language is presented in Chapter 2. The two main building blocks of the language are type definitions and parameter specifications. It is possible to build parameterised specifications and combine definitions with extension, union, renaming and actualisation mechanisms. We are gradually taken through the features of the language with increasingly hard examples of mini-specifications for natural numbers, lists, stacks, sets and maps, culminating with a symbol table for a block structured language, which is built from a stack of maps. We are also given an overview of the language syntax and semantics before a 39 page case study development of a syntax directed editor. The authors are very honest about limitations of the language, and point out the features where the language might change in the future. At this point I was seduced by the generality and power of the approach, in particular how specifications can be built by combining smaller units. But I was also left wishing I could experiment with the ACT ONE system. My previous experience of algebraic specifications was with OBJ. I remember the initial frustration with the interface and the strictness of the system, followed (much later) by the jubilation of a specification that could have proofs performed about it.

In Chapter 3 we learn how to develop modular systems using the ACT approach. A module consists of four parts: an export interface, import interface, parameter part and body part. The export interface is sufficient for other modules to use the module. The import interface contains a number of formal parameters that can be matched to the exports of other modules. The parameter part allows modules to specified generically and actualised on use. The body defines the operations