References

Levesque, HL, 1984, "Foundations of a functional approach to knowledge representation" *Artificial Intelligence* **23** 155–212.
McCarthy, J, 1980, "Circumscription—a form of non-monotonic reasoning" *Artificial Intelligence* **13** 27–39.
Reiter, R. 1980, "A logic for default reasoning" *Artificial Intelligence* **13** 81–132.

Handbook of genetic algorithms by Lawrence Davis (Ed.), Chapman & Hall, London, 1991, pp 385, £32.50.

Reviewed by: GV Conroy, Computation Department, UMIST, Manchester, UK.

This text is the second edited by Lawrence Davis dealing with the topic of genetic algorithms. The first *Genetic Algorithms and Simulated Annealing* (Pitman, 1987), was probably the first to introduce the subject matter to the general public, or rather to the general academic. The papers presented in the early text were very much concerned with the theory of the subject, and were largely academic in nature with practical applications only remotely hinted at. With the latest text one can say that the subject has "come of age". A minor point is that the subject of the text is strictly genetic algorithms applied to optimization problems; however, the editor points this out very clearly early in his exposition, and since the problem domain is a real and difficult one this is not intended to be a criticism of the book.

Since to date the subject matter has remained rather firmly within academic circles it is worthwhile outlining briefly just what the genetic algorithm is all about. John Holland, the inventor of the subject, was intrigued with the way that natural selection amongst biological species caused such species to evolve in a relatively short time to complex beings well adapted to their environment. He therefore carried out a program of research into how one could mimic the evolutionary behaviour of natural species with a formal computer algorithm to solve difficult problems. Simplistically speaking, evolution is a process operating on the chromosomes of living beings. The chromosomes are biological mechanisms for encoding the structure of those beings, and are taken to consist of chains of genes which provide the inherited characteristics of living organisms. Successful members of their species are believed to reproduce more often than the weaker members, thus spreading the advantageous genes throughout the population. It is within the reproductive process that evolution is believed to take place, whereby the chromosomes of the parent organisms exchange genetic material. The new chromosomes may be more successful than the parent organisms, will therefore take part in reproduction more often, and eventually dominate the population they belong to. Since the genetic material may possibly converge to a fixed set of genes in some positions in the chromosomal chain, a process of mutation is also believed to operate, albeit rarely, thus allowing the introduction of new genetic material into a stagnant population.

The genetic algorithm mimics this process in the manner described below. Firstly, an attempt at a solution to a problem in the relevant domain has to be encoded as a bit-string vector of features. Secondly, operators have to be found to manipulate these bit-strings in a manner similar to biological reproduction, that is, operators have to be found that will exchange bits between parent vectors in a meaningful manner for the domain at hand, to produce child vectors. Without going into details, two operators are typically used here, crossover and inversion. Thirdly, mutation is introduced whereby bits may be randomly altered to values not possessed by either parent. The children are then assessed for fitness as solution vectors to the particular problem being considered, and the fittest are then allowed to reproduce more than the weakest. The hope is that the population of vectors will eventually converge to a good solution to the problem under consideration.

Each of the above stages in the genetic algorithm process presents its own difficulties. The first stage, that of finding a suitable bit-string vector representation of a possible solution vector, is not a trivial task for demanding problem domains. The choice of the population size to be used is a trade off between the convergence rate of the population and the loss of good genetic material if kept too

low. During the second stage it is not simply a matter of exchanging bits in the bit-string vectors to mimic reproduction, it is also the case that the operators used must have meaning in the relevant domain under consideration. If the vector encodes a sequence, or is order dependent, then the reproductive operators to be used must respect such an ordering. During the third stage, that of mutation, there exists the problem of mutating an acceptable vector into an illegal one. This can either be disallowed or can be left to the fitness evaluation to reject it, although this then wastes a child of good parentage. For a particular domain the choice of fitness, i.e. how good a vector is as a solution, is dependent on the domain. All the stages involve some choice of parameters controlling the various steps in the process, and it is not necessarily the case that such a choice is easy to make. In fact, the use of a genetic algorithm to arrive at suitable parameter values is a recommended approach for obtaining these.

Lest the reader be misled into thinking that the subject area is now fully understood, with all design decisions simply a matter of consulting an appropriate text, I will now refer the reader to the text under review.

I consider this text to be an excellent one, and that from several points of view. The book itself is divided into two parts, one part being a tutorial on genetic algorithms and their application, the other being case studies of applications to real world problems. The tutorial component is excellent: the reader is left with the impression that this is an exciting field to be working in, with the excitement also being conveyed to the reader. Each of the stages in the development of a genetic algorithm is dealt with comprehensively, with many useful illustrations. The text certainly gave me the inspiration to try out these ideas on problems of my own. The material is well written, conveying the key ideas clearly, and should present no difficulty for anyone wishing to study alone. As a text for the educational world I find it very suitable for final year undergraduates in any numerate discipline, since genetic algorithms can be applied to probably any problem area, so there is no particular restriction on the academic discipline involved. I believe that computing students would find the subject matter particularly appropriate, since they are used to algorithmic approaches to problems; in fact, it is my intention to use the material in a course I teach to third year computing undergraduates on knowledge engineering and expert systems. It is also suitable as a topic on an advanced MSc computing course that I am involved with. The usefulness of the text for academics does not stop at the lecturing level, however, since the editor also includes many references to unsolved research topics in the field. Suggestions for further research are sprinkled throughout the material, and I would expect the text to generate many PhD theses in the future. As a research guide the text is also excellent in providing a comprehensive and up to date bibliography of the subject matter; on scanning through three or four chapter reference sections at random, some 40–50% of these references were dated 1989 or 1990.

For the practitioner and the industrialist wondering what there is in the text for them, I am happy to say that there is much food for thought in part two of the book dealing with practical applications. It is probably true to say that for some of the case studies, unless one is actually practising in the field, the exposition is somewhat abstruse. This is possibly a consequence of the fact that the applications are real world ones with their attendant difficulties and complexity. Most of the case studies are, however, very readable, and provide insight into their problem domains. One of the chapters, that on schedule optimization, in fact provides good tutorial material in its own right, as well as dealing with a particular industrial problem.

Available by post is some genetic algorithm software, costing $60 outside the USA. This software has been developed by Lawrence Davis and John Grefenstette, one of the contributors to the text. A knowledge of C or Common Lisp is needed to allow the purchaser to modify the software for experimentation or further development. I would have preferred to see the software before writing this review, but in the time available it was not possible: I have no reason to doubt that, being written in part by the editor, it should be good.

I do recommend this text to all those wanting an introduction to the subject of genetic algorithms at whatever level. For the undergraduate the price of £32.50 will probably be a turn off, since the material would be unlikely to be for a full course. However, most academic libraries should

certainly purchase the text. For practitioners and those academics wanting to get into the field, the book is well worth the money.

The integration of expert systems into mainstream software by Alan C Gillies, Chapman and Hall, London, 1991, pp 246, £19.95.

Reviewed by: Professor R Rada, Department of Computer Science, The University of Liverpool, Liverpool, UK.

This book fills an empty niche in the marketplace. The author notes that he is targeting upper level students in software engineering curricula and software engineers at companies. This seems a reasonable target for the material. Many universities around the world now offer courses on expert systems, and when the course is to emphasize the practical, engineering aspects of building expert systems this book would be helpful.

There are many older books in the area which are well-respected, such as *Building Expert Systems* edited by Hayes-Roth, Waterman and Lenat from 1983, and *Programming Expert Systems in OPS5* by Brownston, *et al.* from 1985. Books appeared in 1990 like *Introduction to Expert Systems* by Peter Jackson, and *Knowledge Systems and IBM Prolog* by Walker, *et al.*, but Gillie's book targets a more specific niche than these 1990 books, and is obviously more recent than the 1983 and 1985 books.

The early chapters of the book address various basic principles, such as the human-computer interface and the assessment of quality. The section on interfaces notes some interesting concepts about the relationship between the normal human–computer interface issues and what are more appropriate decompositions of the problem for expert systems work. My negative comment is that some chunks from these sections on principles could come from a book on another topic as they are not always closely integrated to the specific topic of expert systems.

The book contains three case studies, each occupying its own chapter. The topics are: using the joint knowledge-based system approach to combine human, algorithmic, and knowledge-based approaches to a pattern analysis problem, building an American military decision aid according to classic software engineering rules, and diagnosing tax problems with a system written in Prolog and integrated via C with the relational database management system INGRES. These case studies show the application of the principles described earlier and are crucial to the book's success.

About the style, my positive comments include that the writing style is conducive to relaxed, entertaining reading, and the illustrations are plentiful and helpful. There is a good mix of sentence lengths and use of metaphor. Nevertheless, there are a few places where the jump between a low-level and a high-level explanation seems too large.

The currentness of the topic is complemented by the constructive nature of it. I agree 100% with the author of the book that the major challenge to expert systems is to integrate them successfully in the workplace. This requires both integrating expert systems with the social and psychological needs of people and integrating expert systems with the other computer systems which people use. This book should help those in the commercial audience who want guidelines on how to integrate expert systems into the workplace.

Knowledge-based approaches for structural design by D. Sriram, Computational Mechanics Pubs, USA, 1990, pp 159, £29.00.

Reviewed by: Bimal Kumar*, Design Computing Unit, University of Sydney, Sydney NSW 2006, Australia (*On leave from Department of Civil Engineering, Strathclyde University, Glasgow, UK).

Around the early 1980s there was a sudden upsurge of activity in applications of artificial intelligence (AI) tools and techniques to just about anything. Structural engineers obviously did not want to be left behind, and numerous projects were started in different parts of the world on applying AI techniques to structural engineering. One of the first persons to recognize the