exchange and multi-agent planning. The need for communication and interaction in cooperative multi-agent communities is identified. Two examples of industrial applications are then introduced for use throughout the book: particle acceleration and electricity transportation.

Chapter 2 describes an initial agent architecture called GRATE, in which generic layers of knowledge for cooperation, situation assessment and control are incorporated. GRATE is interfaced to domain-specific systems via the control module. The situation assessment module decides whether activities are local or social, whether to honour a requested social activity and how to respond to new information. The cooperation module is only responsible for managing the agent's social activities and interacting with other agents. The GRATE architecture is then applied to the two industrial applications given in Chapter 1. The particle accelerator example is particularly interesting in that two real, stand alone expert systems are transformed into cooperating agents under the control of GRATE with few modifications to the expert systems and none to GRATE itself. Simulated domain systems are used in the electricity transportation application. Improvements in various aspects are reported in both cases.

Chapter 3 explains a theoretical model for joint social actions, called the Joint Responsibility model. A novel mechanism for coordination is first introduced based on the notions of commitments and conventions. Conventions are used for monitoring agent commitments, which may be withdrawn in a dynamic environment. These conventions define the conditions under which comments may be dropped and what actions to take in such circumstances. A formal language based on modal and dynamic logic is then described for the formal representation of beliefs, goals, actions and commitments from both individual and social points of view. A mental state of Joint Responsibility is finally proposed for coordinated actions together with an associated social convention, known as the Joint Responsibility Social Convention. The use of the Joint Responsibility model is demonstrated in the electricity transportation example.

Chapter 4 presents a computational architecture for the Joint Responsibility model. A specific realisation of the architecture is devised in a refined version of GRATE, called GRATE*. GRATE* overcomes the shortcomings of GRATE in that it can deal with unpredictable and dynamic changes in an agent community since it incorporates the Joint Responsibility Social Convention. GRATE* is finally evaluated against the original GRATE in terms of coherence, computational cost and communication cost.

Chapter 5 gives a summery of the book's contributions to the DAI community, discusses desirable enhancements to the GRATE systems and the Joint Responsibility model, and expresses a vision that the next generation of DAI systems shall have an explicit knowledge level for cooperation.

The book is extremely well-written in its clarity and structure. It offers a substantial amount of theoretically and practically valuable ideas, concepts and techniques for use in multi-agent systems. In addition, it covers general DAI material very well. Therefore the book is highly recommendable to DAI researchers, practitioners, as well as readers who have a general interest in multi-agent systems.

Reviewed by Dr Jun Huang, University of Central Lancashire, Preston, UK

Constraint-based reasoning by E. C. Freuder and A. K. Mackworth (Eds), MIT Press, Cambridge, MA, 1994, pp 403, £31.50/$47.25, ISBN 0-262-56075-5.

With the current explosion in the number of books being published it is important to be very selective about what you read. This book is a reprint of the 1992 issue of *Artificial Intelligence* (Vol. 58, Nos. 1–3). If you have not seen that issue, then this book is well worth reading.

The issue includes several very important papers, of which I will pick out three. "Partial Constraint Satisfaction" by Freuder and Wallace (no relation to myself!) is a thorough study of how to apply constraint satisfaction techniques embedded in a branch and bound algorithm to problems with "soft" constraints—constraints that should preferably, but not necessarily, be satisfied. "Constraint Satisfaction Using Constraint Logic Programming" by Van Hentenryck, Simonis and Dincbas is a review of the highly influential CHIP constraint logic programming system. Pascal Van Hentenryck decided to effectively reconstruct CHIP as an instance of the concurrent constraints scheme, and it works! "Minimizing Conflicts: a heuristic repair method for constraint satisfaction and scheduling problems", by Minton, Johnston, Philips and Laird, is a fine presentation of an exciting result which emerged while writing programs to schedule the Hubble telescope. The paper takes an extra step beyond the experimental evidence that the repair technique works well: it examines reasons for its success and analyses conditions under which it works well and not so well.

The papers in this collection are of a consistently high standard, and whilst the three landmark papers above fully justify reading the book, there is plenty more to learn from the others. Nevertheless, you can not help asking why do these papers fit under one title?

The two editor's papers focus on constraint satisfaction techniques, but such techniques are at best peripheral in the learning approach of Zweben *et al.*, nor are they used at all in the repair technique of Minton *et al.* In the conclusion of his paper on geometric constraints, Kramer makes a brave attempt to relate his "degrees of freedom analysis" to constraint satisfaction techniques, but a few pages cannot suffice to make this relationship clear. What emerges from this volume is that there is, as yet, no clear framework for constraint-based reasoning. This book, then, is a collection of high-quality samples extracted from a rich terrain of research. The fascinating question remains: what are constraints and how can they be used in producing solutions for complex problems?

The shallow answer, that constraints are (logical) properties of (good) solutions, is too broad in that it fails to distinguish constraint-based from any other form of declarative programming. Constraint-based reasoning typically applies to the class of NP hard problems, where it is very much easier to check if a given answer is correct (i.e., satisfies the constraints) than it is to generate a correct answer. By contrast with traditional forms of declarative programming, constraint-based programming makes a distinction between search (the attempt to generate a correct answer) and constraint satisfaction. Search control is, in fact, an important aspect of constraint-based reasoning, and in this respect it tends to escape from the declarative programming paradigm.

The weakest form of constraint-based reasoning is simply to check whether a given solution satisfies the constraints. However, the power of constraint based reasoning comes from a more active use of constraints to limit and guide the search. This book describes a variety of ways of doing just that.

We can distinguish two fundamentally different ways of searching. One approach is to construct solutions bit by bit applying constraint-based reasoning at each stage to the partially constructed solution. The other way is to start with a complete solution, that may not necessarily satisfy all the constraints, and then to use constraint-based reasoning to repair it, or improve on it, by appropriately modifying relevant parts of the solution.

The incremental search has given rise to the constraint-based reasoning techniques described in most of the papers in this collection, those by Mackworth, Freuder *et al.*, Hyvonen, Cooper *et al.*, Kramer and van Beek. By contrast, Minton *et al.* and Yang focus on repair techniques. The paper by Zweben *et al.* on "Learning to improve constraint-based scheduling" is the only one that clearly transcends this division by working at the meta-level: reasoning about constraint-based reasoning.

We still await a more abstract view of constraint-based reasoning, which can relate constraints and constraint behaviours applied to both forms of search: incremental and repair-based. Such a view must be clarified before we can take the next major step in constraint-based problem solving, the integration of incremental and repair-based search into a single scheme. It is this integration which defines the current frontier of debate and argument for many researchers in constraint-based reasoning today.

Reviewed by Mark Wallace, Department of Computing, Imperial College, London, UK