Intelligent data handling by S. Murdoch and L. Johnson, Chapman and Hall, London, 1990, pp 177, £25.

"Intelligent data handling" is a very general title. I had expected to find a discussion of what "intelligent" handling of data means and how to achieve it, the integration of knowledge-based systems and database systems, knowledge base management systems, and the application of AI techniques such as machine learning and truth maintenance to information management. A discussion of issues such as why is intelligent handling of data needed, who needs it, and what are the likely business opportunities for the technology would have made for a well-rounded treatment of the subject.

I was misled by the title. This book has the appearance of a mid-1980s PhD thesis belatedly published without substantial updating. This seems like a harsh judgement, but one in which we see the origin of most of the book's shortcomings: its content, in parts, is now rather dated; for example, all but one of the references date from 1986 or earlier, mostly the early 1980s; the main substance of the book is a discussion of a specific piece of work on the distributed architecture for expert systems, preceded by a detailed account of an earlier expert system, MDX/PATREC/RADEX (Ohio State University, 1983), that unwittingly displays the proposed architectural features. As a result, this book is not a general overview of the current state of intelligent data handling.

The book argues that first-generation expert systems were *performance expert systems*, in other words, their metric for success was the percentage of "right" answers they obtained. Second-generation expert systems, we learn, should be *competent expert systems*; they should attempt to capture the underlying model of competence of the expert. The competence failings of the first-generation can be attributed to a "single root cause", which has two parts: some of the knowledge essential for displaying competent behaviour is not explicitly represented in the knowledge base; those expert systems do not follow the same line of reasoning or use the same knowledge structures as the human expert. It is proposed that the "distributed architecture" addresses these issues, and hence provides a basis for constructing competent expert systems. The architecture imposes a regime under which two types of knowledge necessary to perform an expert task are separated: the knowledge specific to a particular case, and the generic expert knowledge.

Chapter 1 discusses the need for competence, and then introduces MDX as the only previously reported expert system built with the "distributed architecture". MDX's structure had resulted from making an analogy with the way the medical community organizes itself. It is not clear why the term *distributed* is used, when what is really meant is *partitioned*. The authors are referring to a logical partitioning of the knowledge base not a physical distribution. As early as page 14 the authors' habit of writing-by-quoting-chunks-of-other-peoples'-text had become noticeable – that page alone has five chunks (this habit excels itself on pages 81 and 95 by repeating the same five-line quote).

Chapter 2 is one place where the age of the material really stands out. The world of expert systems coupled to databases, knowledge bases and very large databases has moved on a long way since this text was written. We learn that "... expert system research and database system research could probably both benefit from cooperation". The fact is that they have cooperated a great deal during the 1980s, so that there are now journals and large conferences dedicated to this subject (for example, the Expert Database Systems Conferences).

Chapter 3 talks about the need for competence in expert systems. This requires a clearer statement of what is expected of a competent expert system. The authors are right in saying it is a common belief that once knowledge has been represented in a knowledge-based system for one task, then it is adequate for other tasks. The explicit representation of knowledge within a system is an enabler for intelligent handling, not a guarantee. Tasks such as teaching, explanation and justification, which are aspects of *competent* behaviour, are achieved through manipulation of this explicit knowledge. If a system can be said to display some "intelligence", surely that is due to the power of the meta-level in handling the *object knowledge* – a discussion of this would have made for a good book.

MDX is discussed in detail in Chapter 4, and is projected onto the "distributed architecture". A much clearer statement of the architecture and its components is needed by this point, together with a discussion of how it differs in substance from other expert system architectures.

Chapter 5 looks at the PATREC component of MDX, and finds it lacking in many respects, in particular that its knowledge is encoded as LISP functions ("an arcane representation of knowledge"). To a certain extent the discussion here of data abstraction is a distraction. As long as the meta-level can manipulate the knowledge as required, it is unimportant how it is represented.

Chapters 6 to 8 cover the authors' own research in the area of competent expert systems. The principal reason for adopting the "distributed architecture" was to enable *competent* expert systems to be built, and so *competence* should have played a much greater part in these chapters. Chapter 6 is a detailed case study of FIBERS, which is a shell for the management of "findings base" (one of the components of the "distributed architecture"). The following chapter illustrates how to integrate a FIBERS findings base with a task-structured hypothesis engine (Appendix B gives an example of the system in action). The transition from Chapter 7 to Chapter 8 is a little disorientating, as we are suddenly thrown into the world of knowledge engineering environments with a "sales-pitch" for KEE.

The final chapter concludes that "To corroborate the arguments presented here, one (or more) full distributed expert system needs to be developed in real-world domains of expertise". The book would have benefited here from giving a present-day view – has a "full distributed expert system" been built, does it satisfy the requirement of *competence*, and how does this work now fit into a broader picture of expert systems development?

The untimeliness of this book is exacerbated by the realization that there are only about 120 pages of real text, to which is added about 30 pages of code listings. At £25 that does not seem like very good value for money.

Reviewed by John Manley, Hewlett-Packard Laboratories, Filton Road, Stoke Gifford, Bristol BS12 6QZ.

An introduction to text processing by Peter D. Smith, The MIT Press, Cambridge, MA, 1990, pp 300.

In the blurb on the back cover, the publisher claims that this is "the first broad, complete and balanced survey of this burgeoning applications area, and the theories that underlie it". In fact, the book is not the first one in text processing, and it is neither complete nor balanced. It offers instead a practical, system-oriented treatment of text-manipulation packages, concentrated very largely in the areas normally known as word-processing, that is, text editing, document formatting, text transformations, and related applications. The author is obviously an expert in the use and evaluation of many of the existing text processing packages, with emphasis especially on text manipulation in the UNIX processing environment. In consequence, the two thirds of the book devoted to word processing facilities will be useful to the practitioner interested in the capabilities of available text handling systems. Repeatedly, the author includes specific program segments, and subsequently demonstrates their effect on particular types of input. This approach is appealing, and it represents by far the most notable feature in this volume.

Beyond this, I found the text largely disappointing. There is first the most unfortunate subdivision of the topic into areas labelled character-by-character handling, symbol-by-symbol handling, word-by-word handling, and natural language processing. These are differences without a distinction that have little intellectual merit. Most text handling applications are solved by approaches that include features from all these categories – for example, text retrieval may be based on certain character handling programs (used, for example, in text scanning systems); in addition, word fragment operations are required to remove suffixes from full words to construct