Implementing Typed Feature Structure Grammars

Ann Copestake
(University of Cambridge)

Stanford, CA: CSLI Publications (CSLI
lecture notes, number 110), 2002,
xi+223 pp; distributed by the
University of Chicago Press;
hardbound, ISBN 1-57586-261-1, $62.00;
paperbound, ISBN 1-57586-260-3, $22.00

Reviewed by
Gerald Penn
University of Toronto

Typed feature logic had been promising to deliver large-scale, semantically rich grammars ever since Pollard and Sag's (1987) seminal book on head-driven phrase structure grammar (HPSG) appeared. Several private research labs had been writing them, we were told, but they were so good and so valuable that no, sorry, you can't take a peek at them. It was not until the Linguistic Grammars Online (LinGO) consortium, of which Copestake is a member, dedicated themselves to the task of writing a publicly available English grammar, now known as the *English Resource Grammar* or *ERG* (Flickinger 1999), that typed feature logic finally made good on that promise. There is now at least one large-scale grammar based on typed feature structures.

The consortium extended an existing system for developing lexical knowledge bases, the LKB, to the task of developing such a grammar. The LKB is not the first grammar development system for typed feature logic, nor the fastest, nor the most expressive (all three of which are rumored to exist within the walls of the aforementioned research labs), but it, too, is publicly available. The intention of Copestake's book, about half of which is a user's manual for the LKB, is to make it even more accessible, particularly to those, as the author puts it, "who do not like equations, mathematical symbols, Greek letters and so on" (p. 4). That is indeed a tall order, and skeptics could certainly argue that this is not an audience from whom we would generally expect precise, large-scale grammars anyway. Copestake does a very respectable job of bringing the necessary discrete mathematics to them, however, mainly by illustrating concepts through many well-chosen examples and exercises throughout the presentation. To that extent, this book will serve as a very welcome introduction to the topic for novice grammar writers.

The book also contains a number of optional sections that present the definitions more formally, for those who wish to delve further into the logic itself. Even in these there are some ambiguities, though, as well as some mistakes and parochial choices of terminology. The distinction between *specifications* of mathematical structures and the structures themselves is often blurred, for example. The phrase *constraint on a type* is used sometimes to refer to a specific statement of entailment that a grammar writer provides (e.g., p. 68), but elsewhere to refer to the most general satisfier of a type relative to the entire constraint system, which is computed under multiple inheritance from those specifications (p. 75). *Well-formedness*, which is normally used to refer to the syntactic correctness of feature structures, is instead used here to refer to well-typing, which indicates an adherence to the semantic type system of the grammar. The

syntax of descriptions is presented in Backus-Naur form, but descriptions themselves are not abstractly defined (as distinct from the feature structures they denote). In an intensional description logic such as typed feature logic, this distinction is important, and one that beginning students often miss. These ambiguities, combined with a very sparse collection of bibliographic citations, may make it difficult for novices to find clarification elsewhere.

A greater concern is that, among the expositions of syntax and common pitfalls that constitute grammar training, there is very little to be found here in the way of grammar guidance. What do not just functioning but *well-written* grammars look like? What properties should that compliment—that a grammar is well written—imply? If some connection to the theory of programming languages can be drawn, a grammar that is well written is a grammar that is efficient, but it is also a grammar that is documented and testable or otherwise verifiable, a grammar that is reusable and adaptable, and to that end, a grammar expressed in terms that are appropriately modular and encapsulated. These topics are only briefly discussed in the penultimate chapter, "Advanced Features," and even then only with an acknowledgment (p. 192) that they are indeed important, and that semantic typing helps.

Of course, only time will tell whether, from among the several generations of LKB users that will undoubtedly issue from this book, truly exceptional HPSG writers will emerge. Historically, at least, the proposals for capturing the functionality that large, well-written grammars require have encompassed a rather wide range of grammatical constructs and their combinations. The LKB, which essentially consists of only phrase structure rules and a large partial ordering of types, falls very close to one end of that spectrum. Although the observation that a grammar of the size and coverage of the English Resource Grammar should have first been developed at that end certainly has not been lost on those of us who have proposed alternatives, even a casual glance at the now complete ERG must cause one to speculate whether every grammar realized within the restricted functionality of the LKB is automatically a well-written one. More attention to principles of grammar implementation was therefore probably warranted here.

In simultaneously designing the English Resource Grammar and the LKB system, the LinGO consortium had the rare opportunity not only to experiment with other answers to this question, but to set a higher standard among grammar development systems in the functionality required to do a proper job of large-scale grammar design. My own recollection of the consortium's reasoning, however, is that, while this design decision was motivated by the requirements of the English Resource Grammar, those requirements in turn were motivated by the intention of making the grammar as portable to other grammar development systems as possible. This resulted in a least upper bound (pun intended) of the functionality commonly available in these systems in the early 1990s, which in retrospect may have been an unfortunate choice. This is clearly an indictment of a much broader research program than what Copestake's book represents, but to a great extent the book and the system it describes are very much reflections of both the positive and negative outcomes of that program. In that respect, the book is indeed a very faithful record and makes these outcomes very accessible for a wide audience to judge for themselves.

References

Flickinger, Dan. 1999. The LinGO English Resource Grammar. Available at ⟨http://lingo.stanford.edu⟩.

Pollard, Carl and Ivan Sag. 1987. *Information-Based Syntax and Semantics. Volume 1: Fundamentals*. Stanford, CA: Center for the Study of Language and Information.

Gerald Penn is the co-designer of the Attribute Logic Engine (ALE). He has conducted research in typed feature logics and HPSG for over 10 years. Penn's address is Department of Computer Science, University of Toronto, 10 King's College Road, Toronto M5S 3G4, Canada; e-mail: gpenn@cs.toronto.edu; URL: www.cs.toronto.edu/∼gpenn.