Book Reviews

Functional Grammar in Prolog: An Integrated Implementation for English, French, and Dutch

Simon C. Dik
(University of Amsterdam)

Berlin: Mouton de Gruyter (Natural Language Processing 2), 1992, x + 264 pp.
Hardbound, ISBN 3-11-012979-5,
DM 118.00

Reviewed by
Patrick Saint-Dizier
IRIT-CNRS

This book is a detailed description of a program, called ProfGlot, that implements in Prolog the components of the theory of Functional Grammar, as described by Dik (1989). The system presented in this book deals with English, Dutch, and French; it can parse or produce linguistic expressions in these languages; it can also translate one language into another.

ProfGlot can a priori parse or generate a great variety of grammatical construction types. It is composed of data and rules that together define well-formed constructions and permissible actions and inferences on these constructions. Great care has been taken to define separate modules that describe language-independent principles and rules, on the one hand, from language-specific rules and data, on the other hand. The author concludes that the modules for English, Dutch, and French are comparatively small and that the greater part of the program consists of language-independent rules and principles that could also be used for other languages.

After a very short introduction to Prolog (14 pages) and to Functional Grammar (12 pages), where general principles are introduced, the author presents the overall structure of the ProfGlot program. A relatively ambitious linguistic coverage and a relatively accurate form of semantic representation, dealing with logically complex phenomena, are introduced, e.g., attributive term modifiers, term modality, subordinate clauses of reason, condition, concession, etc. As a result, the reader then gets the feeling that something great is going to follow.

But here disappointment begins. First, the form: the remainder of the book (i.e., from page 45 to 234) is rather a well-annotated listing of a program than a book that presents the principles, the foundations, and the computer formalisms used to implement in an adequate, modular, declarative, and generic way the principles of Functional Grammar. Reading this book and understanding the portions of programs is a difficult exercise, even for a well-trained Prolog programmer (though the reader of this review may think that going into a Prolog program—and probably also into other languages—is almost never an easy task). In spite of a good hierarchical organization of clauses and of their reasonable length, these clauses are extremely difficult to get into. Variable identifiers, in particular, have little mnemonic content. Consider as an

example a clause taken almost at random in the book:

```
eq_term(L1, L2, [OPS,[R1,R2,R3,R4], [OPS,[R11,R22,R33,R44]])
:-
eq_restr1(L1, L2, R1, R11),
eq_restr2(L1, L2, R2, R22),
etc.
```

The difficulty of keeping in mind the different data structures used to encode lexical and grammatical data also obscures the reading of the program.

Second, the content: considering the considerable number of linguistic criteria and the complexity of associated descriptive systems such as feature-value systems that we find in most linguistic systems (Systemic Grammars are a good example, with which Functional Grammars could have some similarities in their general principles), the linguistic elements presented in this book lead me to think that the program probably generates or parses ill-formed sentences. Consider, for example, the lexical entry for *tall* (notice also the number of embedded lists):

```
bpreda(eng,[[tall],[grad],[[[vert],t,[zero]]]]).
```

This entry says that *tall* is an adjective that refers to a gradable property of the noun it modifies, and it is related to a vertical dimension; `t` and `zero` remain enigmatic to me. This description doesn't suffice to indicate which objects in the world *tall* can modify, does it? Next, some programs, for example, describing the functions of the universal generator, look very simple: one clause to treat anaphora (this treatment is said to be rudimentary by the author), one clause also for reflexive arguments, etc.

Then, I must admit, I got lost.

This book should be viewed as a technical annex (but not as a user manual) for the more general book on Functional Grammar (Dik 1989). It does not have any general conclusion and the bibliography is short. It has, however, a quite exhaustive 18-page list of the Prolog predicates defined in the book.

I personally have a neutral position with respect to Functional Grammars, and I would say that, a priori, any real effort to model aspects of language comprehension or processing should be strongly supported by the whole community. But I do wonder what the goal and the use of this book is. It should, however, be noted that books of this form, which give a comprehensive description of the implementation of a real natural language processing system, either theoretical or practical, would be extremely useful to many people. They are also certainly extremely difficult to organize and to write in an accessible and stimulating way.

Reference

Dik, Simon C. (1989). *The Theory of Functional Grammar, Part I: The Structure of the Clause.* Dordrecht: Foris Publications.

Patrick Saint-Dizier is the leader of a research group in Natural Language Processing and Logic Programming at IRIT. His research interests include advanced logic programming formalisms (constraints, types) for natural language parsing and generation, syntactic modeling, and lexical semantics. Saint-Dizier's address is IRIT-CNRS, Université Paul Sabatier, 118 route de Narbonne, 31062 Toulouse Cedex, France; e-mail: stdizier@lexique.irit.fr.