Book Reviews


Many readers of this journal know Patrick Blackburn and Johan Bos (B & B) to be excellent writers, and this book is in many respects a stylistically elegant and remarkably pedagogical introduction to computational semantics. Only a few parts seem a bit forced. On the content side, one may, as always when it comes to textbooks, quarrel about priorities, what is left out and what is included, but at the end of the day, such priorities seem to be mostly a matter of taste. The book contains many excellent chapters and paragraphs that fit many different curricula, not only courses in computational semantics.

In the introduction, B & B urge us to think of the book as a toolkit rather than a conventional book. Such use of the book is supported by an accompanying website, from which, among other things, files mentioned in the book can be downloaded. One trouble with the book, as I see it, shows up already at this point. If a book is intended as a toolkit, quick reference should be easy. Unfortunately, the book makes little use of clearly marked or boxed definitions, principles or figures. Many formal definitions are omitted that could serve as ‘packed’ knowledge for students. Let us backtrack a little:

Computational semantics is about composing meaning representations from strings of natural language and to reason about the meanings in context, also on the authors’ definition. Chapters 2 and 3 are thus devoted to semantic composition, while the next two ones are devoted to inference. Chapter 1 introduces first order logic and, briefly, the inference tasks, and Chapter 6 discusses system designs for computational semantics. Each chapter is equipped with instructive notes, where inquisitive students and teachers can find helpful surveys of related literature.

The first chapter is a nice introduction to first order logic and its use in computational semantics. It guides the student in all the right directions: It tells her what to think, intuitively, about various technical notions, and what to worry about (satisfiability rather than truth, for instance); it also bridges the water between model checking and Prolog nicely. Some knowledge of Prolog is required. On the authors’ view, inference is about querying, consistency checking and informativity (model checking, satisfiability and validity in disguise), and each of these tasks are properly defined. There were just a couple of things that bothered me about this chapter. The heavy use of parentheses, for instance, which in a couple of places makes things almost unreadable, and the very abstract motivation of the inference tasks. The motivation for caring about informativity (validity) is that “like inconsistency, uninformativity can be a sign that something is going wrong with the communicative process” (p26). It would have been easy at this point to include examples from the realm of natural language processing where informativity can be put to use (disambiguation, style checking, machine translation, etc.), but such examples are
not discussed. The bridge from abstract computational semantics to applications is missing. Of course this is again merely a matter of taste.

I also found the last part of the chapter where the choice of using first order logic is motivated, a bit quirky. The authors (rightly) say that this logic is an obvious choice in the context of computational semantics, and their motivation can be summarized as: (i) It is "straightforward to incorporate background knowledge into the inference process" when first order logic is adopted (p45), and (ii) first order logic "has enough expressive power to handle a significant portion of natural language semantics" (p46). The second claim (ii) is supposed to single out first order logic in favor of modal logics and related ones. While the authors are right that few reliable computational tools for modal logics are available, it is not obvious that natural language semantics can be better captured in first order logic than in decidable extensions of modal logic. Some stubborn students will hopefully pursue this line of research anyway. The only help they get from B & B is a brief survey of related work in the notes. On the other hand, B & B show how flexible first order logic is, how it can come to grips with second order entities, etc. In the semantics community, some of this is quite innovative (while logicians may find it trivial), and it is surely suggestive to students who wish to do computational semantics in less expressive logics.

Chapter 2 introduces the λ-calculus by trial and error, it is said. This style of representation is clearly intended to be pedagogical, but personally, I found it a bit forced. Two preliminary experiments are described for a small fragment of English, one in which lexical semantics is trivial (but natural), and the rules are used to explicitly link arguments, and another in which named variables are used to keep track of the syntactic elements. Both experiments fail to capture generalizations, and the first is even inadequate. The failures were illuminative, but it wasn’t clear to me how, exactly, one error lead to another trial.

The introduction of the λ-calculus itself is excellent and intelligible for students at all levels. Some parts are very pedagogical (the authors spend one and a half page on saying that bound variables are only placeholders, and that accidental bindings are to be avoided), but everything serves a purpose. The authors then go on to show how the λ-calculus can be implemented in Prolog. This part of the chapter too is excellent. The last part of the chapter introduces a larger fragment of English.

The notes at the end of Chapter 2 are again excellent. Obvious alternatives to the λ-calculus are mentioned, the history of and the various perspectives on the λ-calculus are introduced, and the question of typed versus untyped λ-calculus is discussed in the context of computational semantics.

Chapter 3 is about scope ambiguities and motivates nicely the enterprise of syntactic derivation of such ambiguities. It takes seriously the view that one could instead compute the weakest reading of the sentence in question and then rely on pragmatics to crystallize the exact meaning, but the view is refuted by considering more complex examples where readings are logically independent. Even if such a view is rarely defended in computational linguists, a textbook ought to take it seriously and tell the student why it is not the received view. Thumbs up!